ANL/DIS-07-2

Argonne NATIONAL LABORATORY

NORTHWESTERN UNIVERSITY

Proceedings of the Agent 2007 Conference *on* Complex Interaction and Social Emergence

Co-hosted by

Argonne National Laboratory
Northwestern University

In association with
North American Association for Computational Social and Organizational Sciences

Northwestern University

Norris Center
1999 Campus Drive
Evanston, Illinois

November 15-17, 2007

ISBN 0-9679168-8-7

UChicago Argonne LLC

A U.S. Department of Energy laboratory managed by UChicago Argonne, LLC

Suggested Citations:

Proceedings: North, M.J., C.M. Macal, and D.L. Sallach (editors), 2007, *Proceedings of the Agent 2007 Conference on Complex Interaction and Social Emergence,* ANL/DIS-07-2, ISBN 0-9679168-8-7, co-sponsored by Argonne National Laboratory and Northwestern University, November 15-17.

Paper: Jones, A., and B.C. Smith, 2007, "Title of Paper," in *Proceedings of the Agent 2007 Conference on Complex Interaction and Social Emergence,* ANL/DIS-07-2, ISBN 0-9679168-8-7, M.J. North, C.M. Macal, and D.L. Sallach (editors), co-sponsored by Argonne National Laboratory and Northwestern University, November 15-17.

About Argonne National Laboratory

Argonne is a U.S. Department of Energy laboratory managed by UChicago Argonne, LLC under contract DE-AC02-06CH11357. The Laboratory's main facility is outside Chicago, at 9700 South Cass Avenue, Argonne, Illinois 60439. For information about Argonne, see www.anl.gov.

Availability of This Report

Decision and Information Sciences Division Office
Argonne National Laboratory
9700 South Cass Avenue
Argonne, Illinois 60439-4832
Telephone: (630) 252-5464
Home page: http://www.dis.anl.gov/
Agent home page: http://www.agent2007.anl.gov

Reproduced from best available originals.

ANL/DIS-07-2

Argonne NATIONAL LABORATORY

NORTHWESTERN UNIVERSITY

Proceedings of the

Agent 2007 Conference

on Complex Interaction and Social Emergence

Co-hosted by

Argonne National Laboratory
Northwestern University

In association with
North American Association for Computational Social and Organizational Sciences

Northwestern University

Norris Center
1999 Campus Drive
Evanston, Illinois

November 15-17, 2007

UChicago Argonne LLC

A U.S. Department of Energy laboratory
managed by UChicago Argonne, LLC

CONTENTS

THURSDAY, NOVEMBER 15, 2007

Methods and Techniques — Parallel Track I

Multiplatform Methods

Measurement and Validation Methods

Evolutionary Methods

Toolkits Track — Parallel Track II

Toolkit Surveys

Emerging Toolkits

NetLogo Toolkit Developments

Repast Toolkit Developments

FRIDAY, NOVEMBER 16, 2007

Computational Social Theory

Orientation and Action

Network Dynamics

Organizational Theory and Practice

SATURDAY, NOVEMBER 17, 2007

Social Simulation Applications — Parallel Track I

Social Interaction and Cognition

Banking, Finance, Business, and Economics

Social Simulation Applications — Parallel Track II

Health Care and Epidemics

Socio-Technical Systems

Social Simulation Applications — Combined Track

Spatial Agents

FOREWORD

Welcome to Agent 2007, co-hosted by Argonne National Laboratory and Northwestern University, in association with the North American Association for Computational Social and Organizational Science (NAACSOS). This is the eighth year of the Agent conference series. As at previous meetings, this year's conference maintains a three-theme organization: (1) methods, toolkits, and techniques; (2) computational social theory; and (3) social simulation applications.

The broader theme of the 2007 conference is *Complex Interaction and Social Emergence*.
Agent modeling has transformed scientific methodology across a number of disciplines. One of the strengths of agent simulation is its ability to represent fine-grain and dynamic interactions among diverse types of actors. Because rich interactions are a frequent source of emergent complexities — including social norms, institutions, and transformations — this focus has significant theoretical potential and, thus, implications for computational techniques and various types of applications, as well. Agent 2007 encourages researchers to explore and report on their modeling of complex interactions and on the diverse forms of emergence that arise from their work.

Our invited speakers — Ian Foster, Rosaria Conte, and Leigh Tesfatsion — are leaders in their fields, with contributions in diverse areas of agent-based modeling. *Ian Foster* is the Arthur Holly Compton Distinguished Service Professor of Computer Science at The University of Chicago and Director of the Computation Institute, a joint project between The University of Chicago and Argonne National Laboratory. At Argonne, he leads computer science projects aimed at developing advanced distributed computing ("Grid") technologies.

Rosaria Conte is a cognitive and social scientist and head of the Laboratory of Agent-based Social Simulation at the Institute for Cognitive Science and Technology at Italy's National Research Council. She also teaches Social Psychology at the University of Siena. Her research fields of interest range from agent theory and architecture to multi-agent systems, and from game-theory to cultural evolution and social simulation.

Leigh Tesfatsion is a Professor of Economics at Iowa State University whose current research focuses on agent-based computational economics (ACE), the computational study of economic processes modeled as dynamic systems of interacting agents. Her particular interest is the development of empirically based ACE frameworks for the study of restructured electricity markets.

The combination of conference presentations will help us to explore the present results and future prospects of agent-based modeling. We hope that you will find the conference to be both educational and stimulating. We appreciate your participation and look forward to your future contributions.

The Center for Complex Adaptive Agent Systems Simulation
Decision and Information Sciences Division
Argonne National Laboratory

Charles Macal
Michael North
David Sallach

Northwestern Institute on Complex Systems
Center for Connected Learning and Computer-Based Modeling
Northwestern University

Uri Wilensky
Lynne Kiesling
William Rand

AGENT CONFERENCE INVITED SPEAKERS AND PRESENTATIONS FROM 1999–2007

Agent 1999

Robert Axtell: Why Agents? On the Varied Motivations for Agent Computing in the Social Sciences

Agent 2000

Kathleen Carley: Computational Social Science: Agents, Interaction, and Dynamics
H. Peyton Young: Conventional Contracts

2001 (Agent conference not held due to conflict with National Academy of Science-sponsored Sackler Colloquium)

Agent 2002

Nigel Gilbert: Varieties of Emergence
Kathleen Carley: The Tension between Transparency and Veridicality
Lars-Erik Cederman: Levels of Complexity: Endogenizing Agent-based Modeling
Scott Page: The Interplay of Differences

Agent 2003

Steve Bankes: Next Steps for Social Simulation: Increasing the Utility, Improving the Rigor
R. Keith Sawyer: Assessing Agent Communication Languages
Lars-Erik Cederman: Explaining State Size: A Geopolitical Model

Agent 2004

Roger Burkhart (Methods): Standardizing an Agent Life-cycle Model
Michael Macy (Theory): Social Life in Silico: From Factors to Actors in the New Sociology
Peter Hedstrom (Theory): Social Mechanisms and Social Dynamics

Agent 2005

Steve Bankes (Methods) **–** Supporting the Modeling Life Cycle
Joshua Epstein (Applications) **–** Generative Social Science: Applications of Agent-Based Modeling
Lars-Erik Cederman (Theory) – Growing Sovereignty: Organizational Shifts in State Systems

Agent 2006

Uri Wilenski (Methods) **–** Designing Agent-based Modeling Environments to Promote Restructuring of Scientific Representation and Education
Scott Page (Theory) **–** It's as Simple as ABC: Agent-Based Culture
Noshir Contractor (Applications) – From Disasters to WoW: Enabling Communities with Cyberinfrastructure

Agent 2007

Ian Foster (Methods) **–** Agents in an Exponential World
Rosaria Conte (Theory) **–** Agent Theory: A Missing Requirement of Generative Social Science
Leigh Tesfatsion (Applications) **–** Agent-based Testbeds for Social Science Research, Teaching, and Training

ACKNOWLEDGMENTS

We acknowledge the support of many people in organizing Agent 2007, particularly Kathy Ruffato of Argonne's Decision and Information Sciences (DIS) Division, who – with assistance from Dee Albarado – handled registration, administration, and logistics, and DIS's Margaret Clemmons, who contributed to planning and publication efforts. Argonne's Technical Services Division (TSD) prepared the Program Book and the Proceedings. Mary Fitzpatrick served as the project manager, with editorial support from Andrea Manning and document processing support from Louise Kickels and Lorenza Salinas. Design services were provided by TSD's Michelle Nelson.

ORGANIZING COMMITTEE

Charles M. Macal, Argonne National Laboratory and The University of Chicago
Michael J. North, Argonne National Laboratory and The University of Chicago
David L. Sallach, Argonne National Laboratory and The University of Chicago
Lynne Kiesling, Northwestern University

Thursday, November 15, 2007

Methods and Techniques

Parallel Track I

Multiplatform Methods

THE IMPORTANCE OF BEING DOCKED

S. K. JOHNSON, The MITRE Corporation, McLean VA
M. T. K. KOEHLER*, The MITRE Corporation, McLean VA
D. QUINN, The MITRE Corporation, McLean VA

ABSTRACT

The concept of docking agent-based models has been stressed in many venues for a number of years. In the decision-support context docking can take on a very important role and we have found docking to be an important exercise not when moving from one framework to another but rather when moving from one version of a model to another. In the decision-support context a great deal of time and energy is spent in the Validation, Verification, and Accreditation cycle so a model may be trusted for a particular use. This paper describes the docking exercise we undertook in order to move confidently from a validated modeling framework to one that had not been face validated. The methodology includes taking very large sample runs from each version of the framework. Output analysis included standard descriptive statistics and non-parametric sample comparisons. Of particular note is that at some levels of aggregation the two models appear to behave very similarly. As one disaggregates groups of agents and looks more closely at the results, however, one begins to find differences. This highlights one of the most important lessons of our docking exercise: just like agent-based modeling, docking must be done at an appropriate level of abstraction for the questions at hand. One must understand the context in which the model(s) are to be used in order to understand what differences are of practical significance and what differences can be tolerated. Our output analysis techniques and results will be discussed in detail.

INTRODUCTION

"The purpose of computing is insight, not numbers."
~R. W. Hamming

In general, there are a number of reasons why one would want to compare, or dock (Axtell 1996), models. First of all, one simply may be curious about the similarities and differences between two models. These models may seem quite similar or quite different at first glance and a deeper understanding is desirable. Second, one may wish to understand the theoretic difference between models or to understand which model did a better job of representing a common theoretic foundation. Third, one may wish to utilize a model as a decision-support tool and, therefore, must know which model is best and in which contexts. Fourth, one may use the comparison to understand the significance of similarities or differences found in the output of the models. This by no means exhausts the reasons one might wish to compare models but it highlights some of the major ones.

Things change a bit as we move from the academic field to a decision-support context. Here model comparisons become more pragmatic. In many cases the comparisons are done with an eye towards which model is best and in which contexts. This type of comparison can extend

* Corresponding author address: Matthew Koehler, The MITRE Corporation, 7515 Colshire Dr., MailStop H305, McLean, VA 22015, e-mail: mkoehler@mitre.org

to different versions of the *same* model. Furthermore, in the decision-support context it is likely that a model has undergone some sort of verification, validation and accreditation (VV&A). See Koehler, et al. 2006 for a more through discussion of VV&A in this context. Loosely speaking, Verification is determining whether or not you built the model correctly, Validation is determining whether or not you built the correct model, and Accreditation is determining if the model is good enough to use for its intended purpose (Hartley 1997). VV&A can be very time consuming, difficult, and costly. Of course effort put into VV&A is a function of the role the model will play. Models that will be relied upon heavily will under go more rigorous VV&A than will models that are used for thought experiments or to gain rough order of magnitude insights. When moving from one model to another, especially when it is different versions of the same model, any aspects of the original VV&A that can be carried over would be very beneficial. In this particular case we discuss moving from MANA (McIntosh 2007) version 3.0.39 (MANA3) to MANA version 4.00.2 (MANA4). Here docking is used as evidence for the claim that the VV&A done for one model may be valid for another model. We have spent a great deal of time face validating MANA3. Now that we are moving to MANA4 we would like to make the argument that the face validation from MANA3 can carry over (at least to some extent) to MANA4. This may prove difficult, however, as there have been a number of changes made to MANA4. Table 1 highlights some of the more important changes.

Table 1: Major changes made in MANA4.

New in MANA4
Agents have orientation and bearing
Agents can move in formations
Agents have sensor and weapon orientation
Agents can have multiple sensors
Sensors can be of multiple types
Sensors and weapons have look angles
Sensors and weapons have slew rates

The evidence we will use for this argument will come from a docking experiment we undertook between MANA3 and MANA4. As discussed by Robert Axtell (Axtell 1996) docking can be achieved in three basic categories. Essentially, docking is the alignment of two different models to understand if one model can subsume another. In the case at hand we wish to say that the two models (MANA3 and MANA4) are not different. Axtell proposes three levels of docking: identity, where two models produce identical results; distributional, where the two models produce statistically indistinguishable results; and relational, where the two models produce output that “behaves” in the same way, meaning that similar changes in inputs cause similar changes in outputs but the distributions are statistically distinct. The necessary level of docking is a function of the empirical relevance of the models, which, in turn, is a function of how the models are relied upon and is the topic of the next section.

Specifying the Relevance of an ABM and Level for Docking

Axtell’s Framework of Empirical Relevance (FER) relates a model to its input data and output (Axtell 2005). The relationship is, generally, input data of various types are necessary to create an ABM; once the model is run it will create output data that will relate to real-world phenomena in some particular way. There are four levels to Axtell's FER. Level 0 is, essentially, a well functioning program that is bug free. Level 0 models have qualitative correspondence at the agent level. This means that the agents behave in a manner that is logically consistent with the subject being modeled. Level 1 is the next level of Axtell's FER. Level 1 is macro-level qualitative correspondence to the dynamics of interest. In this level the agent activity generates dynamics, as a whole, that relate to the phenomena being modeled. For example, a group of agents trading with each other may produce a clearing price for the artificial market. This clearing price may not relate to a real-world clearing price but one was found. Axtell's levels continue with Level 2. Models that fall into this category have macro-level quantitative correspondence with the real-world phenomena being modeled. These models produce the correct distributions within their output. For example, Axtell's model of firm size produces a power-law distribution that is the same as real-world data on the distribution of firm sizes in the US (Axtell 2001). The final level of this framework is that of Level 3. In this level the model not only has macro-level quantitative correspondence but also micro-level correspondence. In general, very few ABMs achieve this level of empirical relevance. This is because it is difficult to specify such a model and even more difficult to obtain the data necessary to estimate such a specified model.

In the decision-support context the weight put upon model output and the importance of decisions based upon said output will necessitate that the model achieve a particular level of the FER. Here MANA is being used in a decision-support context to aid subject matter experts (SME). MANA is only one part of the whole decision-support infrastructure. Furthermore, MANA is to be very fast turnaround. Therefore, MANA is understood to provide rough order of magnitude answers to what-if analyses. This places MANA squarely on Level 1 of the FER scale. MANA needs to be in qualitative agreement at the macro-level and have reasonable micro-level behaviors. This implies that identity between MANA3 and MANA4 is not necessary to conclude that they are equivalent. Therefore, distributional equivalence is adequate to conclude that these models are equivalent, and relational equivalence may be adequate.

RESULTS

As we have already stated, we are interested in showing distributional equivalence of the two versions of the simulation, rather than identity. In our docking experiment, only the versions of MANA were different. We used the same scenarios with the same random seeds and ran the same number of sample runs through the two versions of the simulation.

We tested several types of scenarios, with variable levels of complexity in terms of terrain, agent behaviors, agent interactions with each other and their environments, communications, and weapon systems. Our objective was to test a set of scenarios with low, medium, and high complexity in both versions. We hypothesized that our results would show statistically indistinguishable results between versions since we did nothing to the scenarios other than port them into each model and run them a large number of times. However, if significant differences occurred, would they happen across all of the scenarios, to include the simplest set of scenarios, or would they happen in just the more complicated cases?

The low level complexity scenarios simply exercised important features of the models in the simplest manner possible. The medium level complexity scenario added slightly more complicated terrain, weapons, communications, agent and squad behaviors, and contained a few more squads than the low level complexity scenarios. The high level complexity scenarios have the same terrain as the medium level complexity scenario; however they have a much more sophisticated communications structure, more types of agents, a much larger variety of more complicated weapons, as well as more complicated agent interactions. We setup our experiment to test a variety of scenarios to determine if VV&A should be done on each scenario moved from MANA3 to MANA4 or if we could assume based upon this docking experiment that VV&A is independent of the MANA version and would transfer from MANA3 to MANA4 with the scenario.

Though MANA produces a reasonable set of output data, for the sake of simplicity we chose to use the most basic MANA output statistic: casualties. First, we did a crude comparison of raw casualty numbers by scenario, by replicate, and by side between the two versions of the simulation and found that identity did not exist for any of the scenarios. However, since we do not require identity, the next set of tests explored whether or not the differences between the versions of MANA were statistically significant or if we could claim distributional equivalence of the versions.

We treated the data as a paired sample for our experiment because we used the same scenario with the same random seeds with the before and after elements represented by the two versions of the simulation. We ran the parametric Paired Sample t-Test and the nonparametric Wilcoxon Signed Rank Test and Sign Test as our test procedures. The Paired Sample t-Test assumes the data comes from a normal distribution. The Wilcoxon Signed Rank Test does not assume the distributions of data are normal, however it does assume the distributions are symmetric. The Sign Test does not assume normality, nor does it assume symmetry, but it has less power to detect significance if there is symmetry. Review of the Q-Q plots for each of the datasets indicated that most of the distributions had heavy tails, and therefore appeared to deviate from a normal distribution. Evaluation of the measurement for skew, which is one indicator of how far from symmetric distributions are, clustered the majority of values for each of the paired differences in the scenarios tested within the -0.1 to +0.1 range. Since most of the data sets appeared inconsistent with a normal distribution and since their values for skew did not tend to be too extreme, we are reporting the Wilcoxon Signed Rank Test results in this paper. Of note however, in almost every case, the three tests produce the same results with respect to statistical significance (Bhattacharyya 1977; Gibbons 2003).

Table 2: Wilcoxon Signed Rank Test results between MANA3 and MANA4 for the highest level of aggregation for each scenario complexity level.

Complexity Level - Scenario Name	Pairs	Wilcoxon Signed Rank Test
Low - Four Groups	Blue Casualties MANA V3 to MANA V4	0.279
	Red Casualties MANA V3 to MANA V4	0.141
Low - Two Groups	Blue Casualties MANA V3 to MANA V4	0.143
	Red Casualties MANA V3 to MANA V4	0.311
Low - Two Groups with Communications	Blue Casualties MANA V3 to MANA V4	0.138
	Red Casualties MANA V3 to MANA V4	0.980
Medium - Scenario 1	Blue Casualties MANA V3 to MANA V4	0.797

	Red Casualties MANA V3 to MANA V4	0.000
	Civilian Casualties MANA V3 to MANA V4	0.259
High - Scenario 1	Blue Casualties MANA V3 to MANA V4	0.000
	Red Casualties MANA V3 to MANA V4	0.000
High - Scenario 2	Blue Casualties MANA V3 to MANA V4	0.000
	Red Casualties MANA V3 to MANA V4	0.000

In addition to testing several levels of scenario complexity, our experiment also consisted of testing the same metric at various levels of data aggregation for the low, medium, and high complexity sets of scenarios. The highest level of aggregation is represented by the Blue, Red, and Civilian casualty pairs. Table 2 shows the p-values for each of the pairs tested for each scenario between the two versions at the highest level of aggregation. None of the casualty pairs tested for the low level complexity scenarios had significant results (all p-values > 0.1), whereas all of the casualty pairs tested for the high level complexity scenarios had statistically significant results (all p-values < 0.0). The medium level complexity scenario returned significance only for the Red casualty pairs between the two versions (Red p-value < 0.0), but the Blue and Civilian comparisons were not significant (Blue p-value = 0.797; Civilian p-value = 0.259). At this level, what these results imply is that we cannot claim distributional equivalence across the board, but we can make the claim for distributional equivalence in some instances. For our experiment, this means that the simple scenarios are distributional equivalent. However, this does not hold once we increase the level of complexity. Unfortunately, distributional equivalence appears to be scenario dependent.

However, is the presence or absence of distributional equivalence only scenario dependent or does it also depend on the level of data aggregation? Do the pairs that exhibit non-significance at the highest level of aggregation also show non-significance for each of their respective sub-categories when the datasets are disaggregated? The next step to the experiment was to break the data into sub-categories and run the same statistical tests using the same metric, total casualties by sub-category by side. This process was repeated for each of the scenarios. We did this to see if we could determine the largest contributors to the differences. For space reasons only the data for the medium level complexity scenario is reported.

We hypothesized that we would see the same outcomes when we disaggregated the data into sub-categories as we did for the highest level of aggregation. For the low complexity scenarios, all of the sub-categories by pair had non-significant results. As expected, in the high complexity scenario nearly all of the Blue and Red pairs returned significant results. Interestingly, however, in the medium complexity scenario we found that we did not have distributional equivalence when comparing the sub-categories. Tables 3 and 4 display the p-value results for the medium complexity scenario disaggregated into killer and victim squad categories respectively. The data in these tables appear to support the idea that distributional equivalence is dependent not only on the scenario and the complexity of the scenario, but also on the level of data aggregation.

In Table 2, for the medium complexity scenario we saw that the Blue casualty and Civilian casualty pairs did not have statistically significant results between the two model versions, whereas the Red casualty pairs were significant. Tables 3 and 4 show mixed results for the individual squad pairs for both Blue and Red. In Table 3, half of the Blue squads had significant results and half did not and almost all of the Red pairs returned non-significant results. This data indicates that between the two versions of MANA the Red killer squads were the same and had about the same number of casualties attributed to them; however, the Blue killer squads were split and did not necessarily have the same number of casualties. If we look at Table 4, we see that for the majority of the Blue and Red pairs, we also get significant results, which suggests that the victim squads were not necessarily the same between the two versions. The results for Red are reasonable because the results at the highest aggregation level for Red were significant. However, the results for Blue in Table 4 seem counterintuitive. At the highest level of aggregation, the results for the Blue casualty pairs were not significant between the two versions, but as we disaggregated from the highest level, the results became significant. These outcomes indicate that the victim squads for Blue were not necessarily the same between the two versions, even though the overall number of casualties for Blue was not significantly different.

Table 3: Wilcoxon Signed Rank Test results between MANA3 and MANA4 by killer squad category for medium complexity level scenario.

Complexity Level - Scenario Name	Pairs	Wilcoxon Signed Rank Test
Medium - Scenario 1	Blue Kills on Red	.
	Blue Advance Guard	0.000
	Blue Cargo Truck	0.360
	Blue Convoy Commander	0.027
	Blue Forward Security	0.525
	Blue Fwd Security Command	0.004
	Blue Gun Truck	0.837
	Blue Rear Security Command	0.000
	Blue Rear Security	0.734
	Red Kills on Blue	
	Red IED 1	0.259
	Red IED 2	0.285
	Red Attack Vehicle	0.000
	Red IED 3	0.157
	Red RPG	0.250
	Blue Kills on Civilians	
	Blue Advance Guard	0.518
	Blue Cargo Truck	0.892
	Blue Convoy Commander	0.063
	Blue Forward Security	0.039
	Blue Fwd Security Command	0.152
	Blue Gun Truck	0.148
	Red Kills on Civilians	
	Red RPG	0.004

Table 4: Wilcoxon Signed Rank Test results between MANA V3 and MANA V4 by victim squad category for medium complexity level scenario.

Complexity Level - Scenario Name	Pairs	Wilcoxon Signed Rank Test
Medium - Scenario 1	Blue Victims	.
	Blue Advance Guard	0.285
	Blue Cargo Truck	0.000
	Blue Convoy Commander	0.000
	Blue Forward Security	0.000
	Blue Forward Security Command	0.000
	Blue Gun Truck	0.000
	Red Victims	
	Red Pickup Truck	0.000
	Red IED 2	0.157
	Red Attack Vehicle	0.000
	Red Rifle Squad	0.000
	Red RPG	0.006
	Red Sniper	0.227
	Red Observer	0.044
	Civilian Victims	
	Civilians	0.259

CONCLUSION

As highlighted above docking can be a difficult undertaking even when comparing seemingly equivalent models. We set out to determine if MANA3 was distributional equivalent to MANA4. Clearly, except in trivially simple cases we cannot make that claim. Where does that leave us? Ultimately it is up to the decision-maker to decide if any of the statistically significant results have practical significance within the decision-support context in question. If the models function in a sound manner, and if the distributions of data between versions, even though statistically significant, differ by a small amount, then the decision-maker may still accept the models as equivalent. This is the case because the statistical significance may have little or no practical significance. However, one cannot, *prima impressionis*, claim that a VV&A assessment will move from MANA3 to MANA4.

It should be noted, however, that this comes as no surprise given the numerous changes highlighted in Table 1. The most significant of these changes include agent orientation and the behavior of sensors. In MANA3 agents and sensors had no orientation; there was no difference between front and back. In MANA4 agents and sensors have, *inter alia*, an orientation and a speed associated with changing orientation. Agent success and failure is premised highly upon an awareness of the environment. Therefore, changes to the behavior of sensors and the way agents "look around" the environment will impact scenario results. MANA4's inclusion of orientation for both agents and sensors significantly increases the verisimilitude of the framework. Finally, as we wanted to make as direct a comparison as possible between the two versions of MANA *no attempt* was made to change default settings in MANA4 to better emulate behaviors of MANA3. Given the higher verisimilitude of MANA4 one, in fact, may not want MANA4 to exactly match MANA3.

Table 5: Medium complexity scenario descriptive statistics for MANA3 (Old) compared to MANA4 (New)

	N	Mean	Std. Deviation	Minimum	Maximum	Percentiles 25th	Percentiles 50th (Median)	Percentiles 75th
BlueOld	500	4.016	1.68089107	1	15	3	4	5
BlueNew	500	3.972	1.71262435	2	14	3	4	4
RedOld	500	16.796	2.264650367	7	22	15	17	19
RedNew	500	16.16	2.17896668	9	21	15	17	18
CivilianOld	500	10.404	3.25052547	2	19	8	11	13
CivilianNew	500	10.162	3.066359588	1	20	8	10	12

Table 6: High complexity scenario descriptive statistics for MANA3 (Old) compared to MANA4 (New)

	N	Mean	Std. Deviation	Minimum	Maximum	Percentiles 25th	Percentiles 50th (Median)	Percentiles 75th
BlueOld	394	138.8654822	13.48921637	57	180	134	140	146
BlueNew	394	128.1395939	3.685237151	116	138	126	128	131
RedOld	394	259.1142132	3.213307415	248	267	257	259	262
RedNew	394	270.5025381	2.999786872	261	279	268.75	271	272

This, again, highlights the role of the subject matter expert and decision-maker. For example, if we recall from Table 2, the results for Red were statistically significant. But if we look at Table 5 the Red means between the two versions (MANA3 = Old; MANA4 = New) do not differ by a large factor and the distributions of data between the two versions are similar. In this instance, a decision-maker may conclude that the results are not practically significant and that the models are equivalent. However, this may not be the case for every scenario. Recall that in Table 2, the results were statistically significant across the board for the high complexity scenarios. The descriptive statistics in Table 6 for one of the high complexity scenarios indicate that the distributions of data for the Blue pairs differ by a much larger factor. In this case, the decision-maker may deem the results practically significant, because in reality, this may mean the difference in whether or not an operation is halted or continued. In this case, carrying over the VV&A of the scenario may not be possible; thus, necessitating a new VV&A cycle for MANA4.

REFERENCES

Axtell R., 2005, "Three Distinct Kinds of Empirically-Relevant Agent-Based Models," Brookings Institution Center on Social and Economic Dynamics Working Papers.

Axtell R., 2001, "Zipf Distribution of U.S. Firm Sizes," *Science*, 293(5536), 1818-20.

Axtell R, R. Axelrod, J. Epstein, and M. Cohen, 1996, "Aligning Simulation Models: A Case Study and Results," Computational and Mathematical Organization Theory, 1:123-141.

Bhattacharyya, G. and R. Johnson, 1977, *Statistical Concepts and Methods*, John Wiley & Sons, Inc.

Gibbons, J., and S. Chakraborti, 2003, *Nonparametric Statistical Inference*, Marcel Dekker, Inc.

Hartley, D., 1997, "Verification & Validation in Military Simulations," in *Proceeding of the 1997 Winter Simulation Conference*, K. Healy, D. Withers, and B. Nelson, editors.

Koehler, M, P. Barry, and T. Meyer, 2006, "Sending Agents to War," in *Proceedings of the Agent 2006 Conference*. Chicago, Il: Argonne National Lab.

McIntosh, G., D. Galligan, M. Anderson, and M. Lauren, 2007, "Recent Developments in the MANA Agent-based Model," in *The Scythe*, issue 1, The Naval Postgraduate School, Monterey, CA.

Measurement and Validation Methods

INTEGRATING ABM & GIS TO MODEL TYPOLOGIES OF PLAYGROUP DYNAMICS IN PRESCHOOL CHILDREN

W.A. GRIFFIN,* S.K. SCHMIDT, A. NARA, P.M. TORRENS, J.H. FEWELL, and C.M. SECHLER, Center for Social Dynamics and Complexity, Arizona State University

ABSTRACT

We illustrate an objective, non-intrusive method that tracks the behavioral, temporal, and spatial data characterizing evolving group processes in children. This work establishes a methodology combining behavioral observational data, GIS, and agent-based modeling as an aggregate tool to give researchers the ability to establish group typologies according to the behavioral and geospatial distributions of its constituents. The proposed integration of behavioral coding with GIS, and the subsequent attempt to reproduce this aggregation with computational simulation has not been attempted before. As such, this work establishes an integrative protocol for measuring peer-to-peer processes and will serve to modify the research criteria in scientific fields using behavioral observation of humans.

*Corresponding author address: William A. Griffin, ISTB-1 530 East Orange Street, PO Box 874804, Arizona State University, Tempe AZ 85287-4804; email: william.griffin@asu.edu . This material is based upon work supported by the National Science Foundation under Grant Nos.: 0339096, 0338864, & 0324208. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

Integrating ABM & GIS to Model Typologies of Playgroup Dynamics in Preschool Children

William A. Griffin, Shana K. Schmidt, Atsushi Nara, Paul M. Torrens, Jennifer H. Fewell, & Casey M. Sechler

The current study is an attempt to further the amalgamation of a multi-disciplinary team that integrates human development, computer simulation, biology, and geography. We rely on emerging technologies and methods in agent-based modeling, social network analysis, and geographical information science to address questions of current interest to scientists studying the typology, ontology, and morphology of group dynamics. Our model systems consists of young children, with each involved discipline contributing towards answering a critical societal question, namely, how do children form relationships in the context of transitions and change? We propose that children's play partners are multiply determined by the combinatorial dynamics generated by a child's own characteristics with those of his or her peers and the geo-spatial characteristics unique to the environment. More specifically, we are proposing that four discrete, yet related, interpersonal dynamics underlie the formation and maintenance of group formation in preschool children. These four indices of behavioral and affective patterning are the foundation of our ability to track groups as they form. Each index provides unique, yet tractable, information about the groups as they arise, disband, or maintain levels of stability. In aggregate, these indices provide a quantitatively robust dataset that captures complex evolving processes. This aggregate -- a compilation of behavior, affect, and geo-spatial location residing in time -- is the basis for determining the validity of our computer simulation model. Our objective is to reproduce the observed pattern of grouping behavior.

Playgroup Morphology and Ontology: Interaction, Process, and Critical Components

Affect Tone. Children’s affective expression can be viewed as a series of affect epochs. From these epochs, two aspects of each participant exchange are generated: (1) affect valence and (2) matching rate. That is, for each child, any social action with another child, provides an opportunity to generate moment statistics (i.e., mean) of affective valence (i.e., positive, neutral, negative), either for a specific episodic exchange or over an extended period of observation. In addition, by gathering affect on each play partner, it is possible to compare affective states between any two individuals at any given time during the observation period; in effect, this permits an estimate of affect matching rates within a dyad or group. Note that affect matching, a group level characteristic, provides very different information about emotion than does the individual’s general propensity to be in a particular affective state. In combination these two features represent intra-individual and inter-individual affect signatures embedded in time, space, and context.

Bid Ratio. Within group (and within some general exchanges) behavior is generated by bid exchanges among its members. At the simplest level, what members do (i.e., the type of behavior (e.g., swing, play in the dirt)) is secondary to how a call to action is made .that is, the bid. Group cohesion is generated, not from the activity generated by the bid, but by the successes of the bidding process. Group maintenance is generated by successful bids; conversely, a series of unsuccessful bids jeopardizes the group. Obviously, it is not the ratio of successful to unsuccessful bids that inherently destabilizes a group but rather the inability of the group members to accurately gauge the social situation exhibited through this ratio .that reflects poor judgment, inadequate social skills, and so on. Moreover, within our construction and coding of the bid process, we will examine the intra-group bid structure. Specifically, as noted above, a bid can receive one of four responses: (1) accepted, (2) ignored, (3) rejected, or (4) counter-bid by other members of the group. Because of this potential response set, we expect each group and each member of the group to display an unique distribution of responses that have evolved from individual propensities combined with group level reinforcement histories.

Intra-Episode Variability. After forming a group, the characteristics of the group (i.e., group phenotype; see Fewell (2003) for discussion about group level phenotypes) are evident by their affective (e.g., affect matching) and behavioral probabilistic structure, or more precisely, the consistency of this structure across episodes of play. This consistency, however, does not imply the lack of either variability or drift. We should, for example, assume some adaptive variability over time; any complex, evolving system typically evidences moderate variability in response to fluctuations of endogenous components and exogenous influences (Auyang, 1998). For playgroups, such fluctuations would come from changes in the environment and the continuous entering and exiting of other children in the group. Furthermore, we expect to see a drift in the structure as the group matures. Again, this would be expected as group members learn to modify their intra-group behavior as a function of history; they would or should be able to telegraph bids (e.g., subtle behavioral cues .idiographic to the group .are enough to initiate or terminate an action or play sequence). We propose that this third feature can be captured using available mathematical tools that: (1) adequately capture and describe the relevant socio-affective and behavioral characteristics of the groups (see e.g., Griffin, 2000); and (2) parsimoniously elucidates intra-group, inter-episode changes .either by estimating changes in probabilistic structure (e.g., sequential analysis) or covariance change (e.g., Price's Equation).

Time-Space. The set of factors that influence play likelihoods can be conceptualized as occupying an n-dimensional space along 4 primary axes: Affect Matching, Bid Ratios, Inter-Episode Variability, and time-space. Whereas the first three reflect intra-group behavior, the fourth dimension - time/space - represents the milieu of these groups. As noted above, play propensity between two or more children may be a function of who is available and where they are physically located relative to some feature of the playground (e.g., swing set); of course, we assume that these two aspects of play behavior are not independent. Time, within the conceptualization of play presented herein, has multiple facets. First, there is chronological time (e.g., 11:00 am). Second, there is calendar time (e.g., October). Third, there is episode frequency (e.g., 3rd time a particular group is seen playing together). And finally, there is episode duration. Groups are followed for an hour in the proposed index coding system. Although there is the possibly of left censoring, and to some extent, right censoring, the average duration of a play epoch(s) is within an hour. We think that the Time-Space axis can be incorporated into our conceptualization of playgroup dynamics via the GIS methodology described above.

These four indices of behavioral and affective patterning are the foundation of our ability to track groups as they form. Each index provides unique, yet tractable, information about the groups as they arise, disband, or maintain levels of stability. In aggregate, these indices provide a quantitatively robust dataset that captures complex evolving processes. This aggregate - a compilation of behavior, affect, and geo-spatial location residing in time - is the basis for determining the validity of our computer simulation model. Our objective is to reproduce the observed pattern of grouping behavior.

Observational Data Collection

Throughout the fall and spring, children's naturally-occurring free-play interactions are recorded. Observations are collected for 5 hours/day each weekday for the academic school year. The observations commence on the first day of classes, and each class has 3-4 coders collecting data each shift. Coders rotate throughout the classroom, remaining unobtrusive and uninvolved in children's activities. They record data using handheld computers, with the data automatically inserted into a database. Data from the handheld computers are downloaded into a desktop computer and converted into files that can be read directly into data management software. The advantage of using the handheld computers is that time-stamped data can be collected efficiently, entered quickly, and recorded with minimal error.

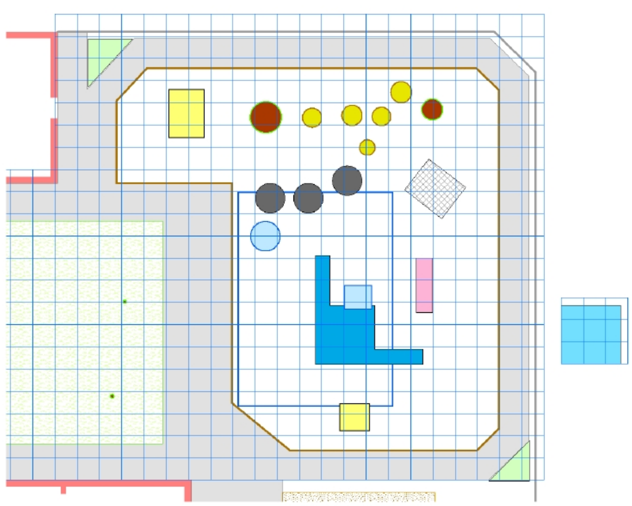


Figure 1. Map of the study site (e.g., outside area with a slide, climbers, playhouse, trees)

Interval Coding. Using a GUI interface, observers identify the first child in a randomized list and briefly (for 10 seconds) observe the child, record data, and then repeat the procedure for the next child on the random list. During the 10-second period, the observer codes several dimensions of the child and his/her context. For example, coders record whether the child is alone, with a teacher, or with other children. For solitary, teacher, and peer codes, the target is observed for activity (e.g., riding a bike, reading books, etc.), affect (i.e., positive, negative, neutral) and the presence of social peers (i.e., peers involved in direct interaction) and area peers (i.e., peers in the physical vicinity but not interacting with the target child). On a fine-grained grid that is digitized to a spatial location on the tablet PC screen, the start point (X,Y), stop point (X,Y), and farthest distance traveled (X,Y) are recorded (see e.g., Figure 1). Additionally, when a target is observed with a peer, we code who the child is playing with, the activity, the affective exchange between the group peers, and the physical location of the group. Such data are used to determine if the specific type of activity, affective proclivities, and physical location influences the degree to which children interact with others (e.g., we can compute separate models for distinct combinations of the three factors).

Group Coding. Each week, the scan data are analyzed to determine cohesive groups. Once a group is identified, a separate coder is assigned to follow each child within the group. Each day, four one hour blocks (2 in the am and 2 in the pm) are allotted for the group procedure. The coders first identify the location of the group members. In a calibrated database, each coder begins recording data into the tablet PCs on their respective child. In repeated 10 second intervals (for 30 minutes), the observer records the context of the event, who is present in the episode, the various affect and behavioral codes (e.g., bids, referencing, attending), and the physical location of the interval. Additionally, each child (whether group members or not) is randomly selected for 30 minute individual increments. The procedure used for the individual index coding is identical to the group observations; this method allows us to make comparisons using similar observational methods for children who form groups vs. those who do not.

Geographic Information Science and Tracking Playgroups

Once the field data are collected, they are transferred to a workstation GIS, where they are organized into a rich longitudinal database of children's movement behavior. These data are then coupled to the behavioral observations and aggregated and reconfigured as necessary to tease-out group movement, clustering, spatial segregation, and spatial polarization. This can be done on a one-to-one, one-to-many, and many-to-many basis for children. Additionally, it can be expressed geographically relative to notable features in the play environment: adjacency to sandboxes, distance from teachers, proximity to the outer limits of the play space, etc.

Applying this methodology has the added benefit of allowing us to query the database by spatial analysis and geovisualization. For example, using spatial analysis, we can run a suite of spatial statistics over the data to look for the formation of statistically-significant clusters of activity or conditions in the model. We can also test for the tendency of certain behaviors to co-locate in space, or identify group dynamics associated with patterns of spatial segregation. Using geovisualization, we can also build-up instance-level and aggregate surfaces of e.g., cooperation or disruptive behavior, and look at these clusters relative to the features of the playground. For example, we can visualize hotspots of collaborative activity, or coldspots where children's play tends to be isolated. The formation of databases of this form has the added benefit of providing a seed data-set for our agent-based model, as well as acting as a calibration and validation resource for our simulation.

We have developed a system for building time geography relationships that captures events in space and time in a robust GIS framework. This allows us to construct space-time paths and space-time prisms for individual, dyad, and group behavior (examples using synthetic data for two children are shown in Figure 2; this is an accurately scaled representation of the school). Doing so further allows us to build a map of activities in time and space, e.g., in what places do young children tend to spend the majority of their play time, how might this differ from other children, how does this vary by time-of-day, how does this alter when polarizing influences are absent, etc. These spatially explicit aspects provide a critical component to the scenario building implemented in our ABM.

Simulating Playgroups: PlayMate

Using dynamic child behaviors to modify the likelihood of interacting with another child, PlayMate provides a representation of postulated developmental shifts in playgroup formation for children ages three to five years. Framed around a state transition model, each child, represented as an agent, can be in one of four states: (1) playing with another child; (2) playing with an adult (a teacher); (3) playing alone after playing with another child; or (4) playing alone after playing with an adult. Play likelihood across the four states is modified through Play Propensity and Arousal (i.e., proxies of internal configurations), with accumulating values in each of the four states for each child (see Griffin et al., 2004, for a review).

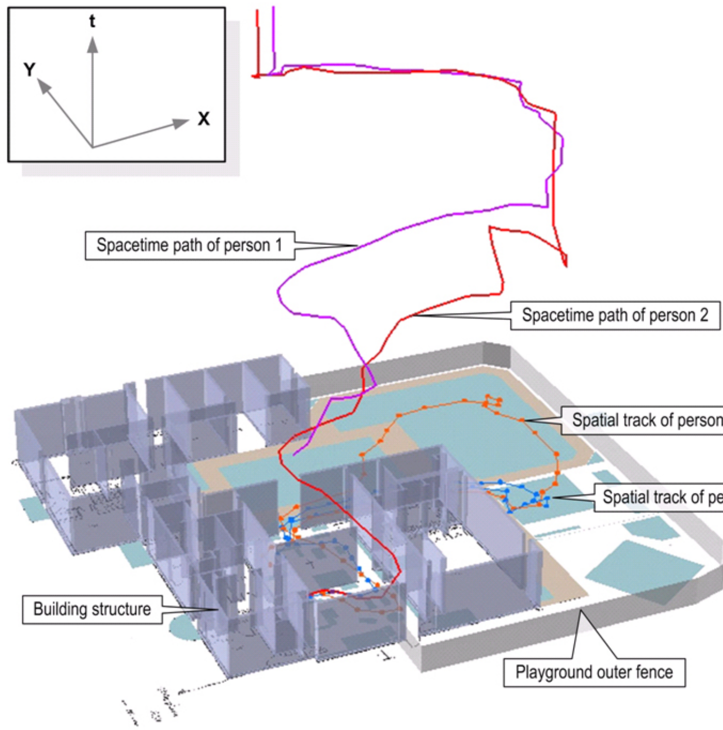


Figure 2. Illustrating the spatial (X,Y) and spacetime (X,Y,t) paths of two children.

To implement the simulation, a child is selected in round robin fashion to play with another child from the available pool (one is randomly removed to simulate a sick-day), and upon pairing, child i assesses child j on several dimensions determined by the investigator; minimally, these include gender and one relevant attribute (e.g., bidding behavior, affect, or a composite of both) being examined. The greater the homophily, as assessed by closeness on the variables in the model, the less likely the child is to exit the child-playing state and to continue playing with other children. Transition rules condition arousal level updates, and behavioral and affective exchanges as well as memory are updated through a summative value after each play episode. The summative values are entered into a tally matrix that, in turn, is converted to a child-to-child probability matrix. The tally and probability matrixes are then compared to similar matrixes extracted from the actual data. For model validation, PlayMate generates numerous quantitative indicators of the structure and composition differences between the simulated and real data; these include difference measures of Euclidian distance, Mean cell values, Entropy, Uncertainty reduction (a measure of mutual information), Solitary play, and row (i.e., child) signal-to-noise ratios. Each measure is assumed to provide slightly different information about the characteristics of the matrix structure. Our existing work validating PlayMate centered on simulating and replicating the peer interaction patterns obtained from coding individuals, with the incorporation of GIS and the Index coding procedure, we will be developing new validation indices. These will necessarily be complex, reflecting aggregate individual, group, and GIS data.

Data Simulation. Prior to running the simulations, each child receives a score based on
the three factors of gender, attribute level, and memory. For gender, each child receives a binary number (e.g., 0, 1), and rank orders for attribute scores are given based a predefined hypothesis (e.g., similarity in affect across all domains drives propensity for play). Finally, integers for memory rankings are based on a list of recent play pairings, with a current capacity of five possible pairings. Simulation runs typically consist of each child in the class playing 50 rounds in the round-robin fashion. Subsequently performing the routine 50 times allowed us to obtain approximately 120-200 play episodes, characteristic of the numbers obtained for each child in the real data within each time frame. State shift and play partner propensities are influenced by the three factors, with each variable weighted according to the theoretical justification that displayed affect and bidding behaviors are the strongest predictors of peer selection. Essentially, increased peer preferences are determined by the aggregate of the three factors, with attribute level difference modifying the likelihood of being in a child play state.

References

Auyang, S. Y. (1998). Foundations of complex-system theories: in economics, evolutionary biology, and statistical physics. New York: Cambridge University Press.

Griffin, W. A. (2002). Affect Pattern Recognition: Using Discrete Hidden Markov Models To Discriminate Distressed from Nondistressed Couples. Marriage and Family Review, 34, 139-163.

Griffin, W. A., Hanish, L. D., Martin, C. L., & Fabes, R. A. (2004). Modeling playgroups in children: Determining validity and veridicality. In D. L. Sallach, C. M. Macal, & M. J. North (Eds.), Agent 2003: Challenges in Social Simulation (pp. 93-113). Chicago: University of Chicago & Argonne National Laboratory.

Schmidt, S. K., Griffin, W. A., Hanish, L. D., Martin, C. L., & Fabes, R. A., Barcelo, H. Greenwood, P. (2005). Playmate: New Data, New Rules, and Model Validity. In D. L. Sallach, C. M. Macal, & M. J. North (Eds.), Agent 2004: Social Dynamics: Interaction, Reflexivity and Emergence (pp. 339-353). Chicago: University of Chicago & Argonne National Laboratory.

AN INFERENTIAL APPROACH FOR VALIDATING AGENT SIMULATIONS

A. YAHJA*, National Center for Supercomputing Center, Urbana, IL
K. M. CARLEY, Carnegie Mellon University, Pittsburgh, PA

ABSTRACT

As the size and complexity of the agent-based simulation models increases so does the time and resources needed to validate the model. Validation is critical for replication of simulation results, which is a basis for scientific advance. Automated and semi-automated tools are needed to support validation activities and so reduce the time and number of personnel needed.

A tool called WIZER (What-If Analyzer) which embodies our inferential approach is implemented. WIZER consists of four parts: an Alert WIZER, an Inference Engine, a Simulation Knowledge Space module, and an Empirical/Domain Knowledge Space module. The Alert WIZER characterizes simulation data with assistance from the statistical tools it semantically controls, compares simulation data to empirical data, and produces semantic descriptions of both the data and the comparison. The Inference Engine performs both causal and "if-then" rule inferences.

WIZER is run on a simulator called BioWar which models disease spread in a demographically-representative city population. The results show that WIZER validates in a clear and automated manner the simulation models for the relative timing of peaks of influenza incidence and school absenteeism. They indicate that the inferential approach underlying WIZER can increase the transparency and reduce the time for model validation.

Keywords: simulation validation, semantics, knowledge systems, causality, virtual experiments

---

* *Corresponding author address*: Alex Yahja, 4038 NCSA Building, National Center for Supercomputing Applications, 1205 West Clark St., Urbana, IL 61801; e-mail: alexy@uiuc.edu

INTRODUCTION

Computer modeling and simulation provides a means of understanding and predicting the behavior of real-world systems based on knowledge of basic laws, empirical findings, and assumptions. It complements theory and experimentation/observation as the third pillar of science. Simulations can be viewed as virtual experiments. Computing advances mean better simulation models can be built. Typically, however, simulation results are designed solely for human analysis and validation is provided by subject matter experts judging that the model "feels right," possibly after preprocessing the results using statistical tools. This process is time-consuming and deficient in clarity, transparency and objectivity. The remedy is usually prescribed in the form of a methodological approach (Yilmaz 2006), of which Verification, Validation and Accreditation (VV&A) process is one, but validation remains a cumbersome process.

NASA lost the Mars Climate Orbiter spacecraft on September 23, 1999. Mission specifications called for using metric units, but the Lockheed Martin group sent navigation information in English units. The mix-up meant that Lockheed Martin engineers modeled navigation with pounds force (the English unit for measuring thruster impulse) while JPL did their calculations in newtons (the metric measurement). One pound force is equivalent to 4.45 newtons. The software for the spacecraft thrusters uses the wrong unit. While management failure played a role, this would never have happened if an automated validation process existed – one as simple as tying-up each number with its semantics. The error would have been caught early if a continuously validated spacecraft and orbital simulations existed. Similar problems and misunderstandings happen in the modeling and simulation world where researchers rarely are able to replicate others' simulations quickly, precisely and reliably.

Computational modeling and analysis focuses on employing computers to build model specifications, verify code, and execute simulation. Indeed, the notion of computational modeling and analysis usually means a quantitative run completed by computers and inference/analysis on the results of the computer run completed by human experts. Computers are not employed to help automate inference, validation, model improvement, or experimental design. Figure 1 depicts this imbalance in automation. Recent advances in data mining made automated analysis more common, but data mining deals only with empirical data, not with automatic building, validating, and improving models. Machine learning approaches can be applied to learning logical, mathematical, and statistical models from data, but they have not been extended to automatic construction, validation, and improvement of simulation models.

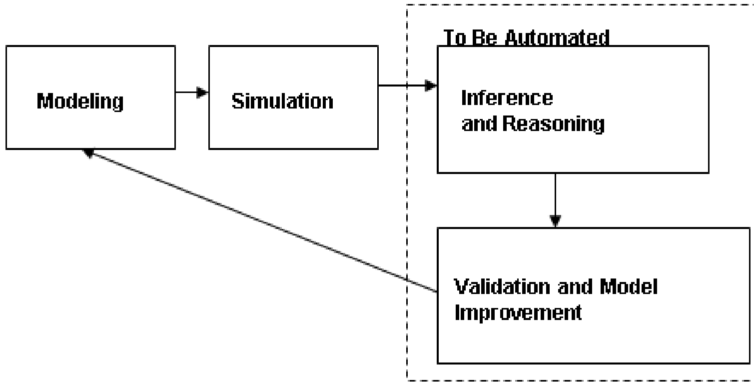


FIGURE 1 Automation of Inference, Validation and Model Improvement

We take the road less traveled to automate validation: an inferential approach uniting simulation with ontological, causal, and knowledge-based reasoning. A tool is implemented based on the approach. This tool is applied to a simulation testbed called BioWar (Carley et al. 2003) which emulates how a city's population reacts to influenza outbreaks.

OUR INFERENTIAL APPROACH

Our inferential approach for validation consists of causal reasoning, knowledge-based reasoning, ontological reasoning, and the scientific method. We call the tool WIZER for What-If Analyzer. WIZER is a knowledge-based tool; the importance of knowledge – and the reasoning based on that knowledge – is emphasized. While WIZER uses statistical tools, they are used in the context of knowledge bases and inferences. Simulation and its outputs are described based on knowledge. Inference rules and descriptions of statistical tools are encoded semantically. WIZER consists of an Alert module, an Inference module, and two knowledge space modules. Figure 2 below shows the diagram of WIZER.

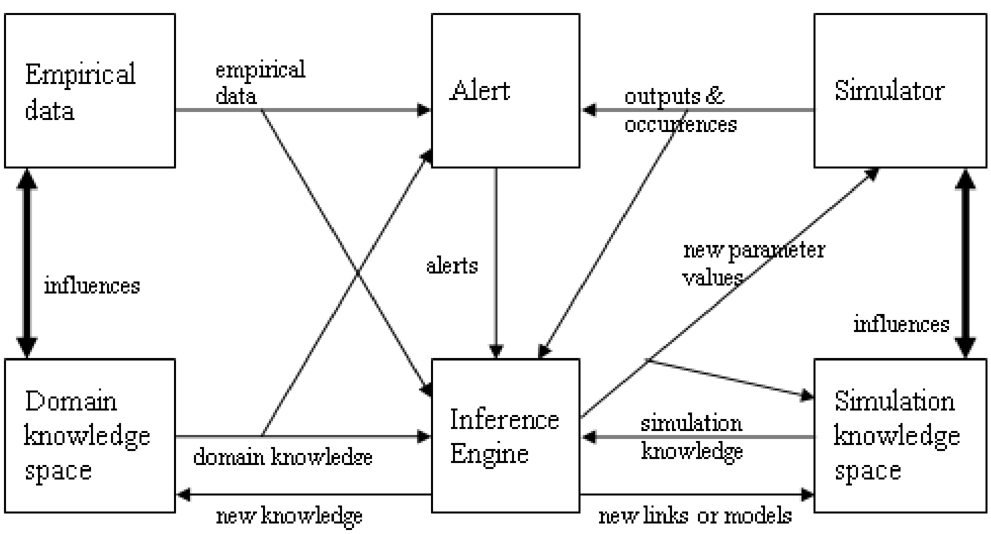


Figure 2 WIZER Diagram

The Alert module does two tasks: (1) describing data using statistical and pattern classification tools, (2) matching a data description with empirical data, producing semantic alerts. Alerts here are defined as semantic characterizations of numerical data (not just alerts in the sense of imminent danger). For example, the Alert module can semantically describe the ups-and-downs of a school absenteeism curve taking into account other semantic or contextual information such as holidays and weather-incurred closings. While not depicted in the figure, the Alert module can also semantically categorize input data and empirical data.

The Inference Engine takes in the outputs from the Alert module, the simulator's causal diagram, a meta-model of the simulation's knowledge space, combined with empirical data, domain knowledge, and parameter constraints (of the domain knowledge space), to determine which parameters, causal links (Pearl 2000, Pearl 2003), and model elements to change – or not to change – and how. The Inference Engine calculates the minimal number of perturbations to the simulation model to best fit the outputs. The model (including the causal diagram) and any potential alternate models are coded into ontologies and rules. Perturbations are implemented as the effects of ontological and rule-based reasoning. An inference produces new parameters for the next simulation. This cycle repeats until a user-defined validity level (which can be defined semantically) is achieved. The user interface module is not shown in the figure for clarity.

The Domain knowledge space module provides domain knowledge to the Inference Engine. Empirical data can change domain knowledge and domain knowledge can ascertain what empirical data are relevant. This depends on the strength of evidence supporting the knowledge and the data. The Simulation knowledge space module provides the simulator with knowledge such as the causal network of the simulation model. The Inference Engine produces new parameter values and possibly new links for the Simulation knowledge space module. The simulator influences and is influenced by the Simulation knowledge space module. The parameter data is empirical, but this empirical data is used in the simulator. Because the empirical data used in the simulator is not the same as the data used for validation, the delineation is clear. Both domain and simulator knowledge spaces are represented by a graph. We use an RDF-based semantic representation. This semantic representation describes and facilitates control of simulation models, knowledge spaces, results, inferences, and statistical tools. In the N3 notation, the basic syntax for RDF is a simple one: *<variable1>* *<relationship>* *<variable2>*, where *variable1* represents a subject, *relationship* a verb, and *variable2* an object part of an English sentence. In our implementation, the verb "causes" can specify empirical relationships. In a conventional ontology and semantics, an ontological and semantic relationship is defined conceptually and logically – based on description logics – and not empirically. Scientific method is employed to get empirical causal relations.

RESULTS OF WIZER RUNS ON A SIMULATION TESTBED

Here we present the results of WIZER runs on the BioWar simulation testbed. BioWar (Carley et al. 2003) is a city-wide simulation model of weaponized biological and chemical attacks on a demographically-realistic population with a background of naturally-occurring diseases.

We describe below the results for one validation scenario that examines the relative timing of the peaks of the children's absenteeism curve and the incidence curve. The empirical data for this scenario is gathered from the National Institute of Allergy and Infectious Disease (NIAID).

Validation Scenario: Absenteeism Curves

The variables and output values for this scenario are as follows.

- Outputs for empirical matching: we choose the simulated actual incidence and school absenteeism drug purchase curves.
- Variables: because the onset of absenteeism is influenced by symptom onset and symptom severity, these two factors are important model variables.

The knowledge base consists of causal rules and "IF-THEN" rules related to the causal ones. The causal conceptual diagram is as follows:

(causes symptom-onset absenteeism-onset)
(causes symptom-severity absenteeism-onset)
(convertible infection-rate incidence-rate); *computable from each other*.

Onsets are computed with reference to the time of infection. The rules related to the causal relations are as follows:

(if-then (toosoon absenteeism-onset) (op-lengthen symptom-onset))
(if-then (toolate absenteeism-onset) (op-shorten symptom-onset))
(if-then (toosoon absenteeism-onset) (op-lower symptom-severity))
(if-then (toolate absenteeism-onset) (op-higher symptom-severity))
(if-then (tooshort absenteeism-vs-actual-incidence)
    (op-toosoon absenteeism-onset))
(if-then (toolong absenteeism-vs-actual-incidence)
    (op-toolate absenteeism-onset))

The simulation instantiations of variables are as follows:

(setvalue symptom-onset 2)
(setpriority symptom-onset 3); *priority for conflict resolution*
(setvalue symptom-severity 3)
(setpriority symptom-severity 1)

The simulation instantiations of outputs are as follows. One BioWar simulation of Hampton city (population 142,561 persons) with 100% scale is run. The Alert WIZER computes the peaks of the actual-incidence and school absenteeism curves. It outputs the relative timing of the peaks. The following figure shows the actual-incidence curve.

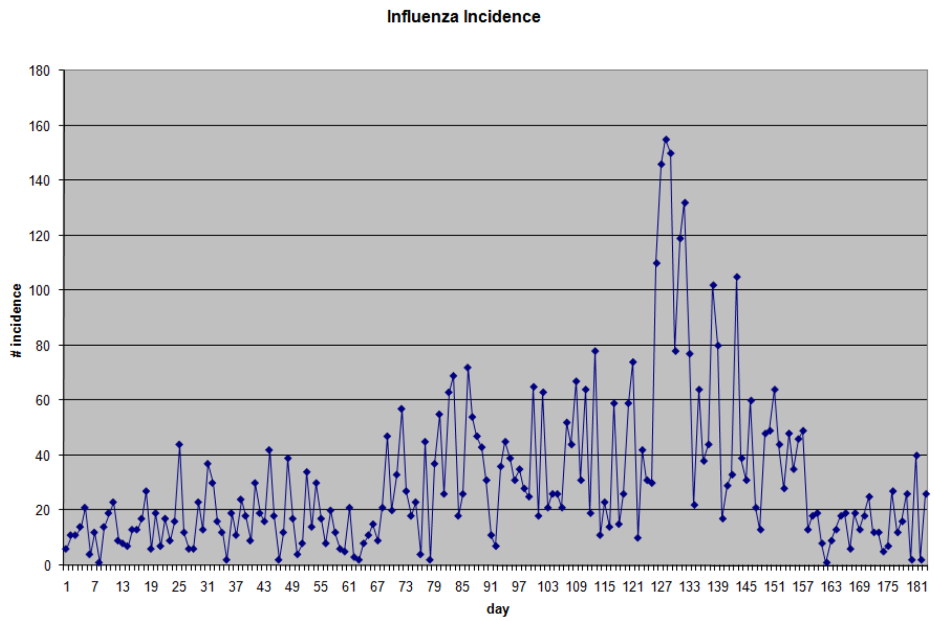


Figure 3 The Peak of Incidence Occurs on Day 128

As shown, the peak of incidence occurs on Day 128. Day 1 is the start of the simulation, corresponding to September 1, 2002. In the simulation trial, the relative time difference between simulated absenteeism and simulated actual-incidence peaks is 10 days.

(setvalue absenteeism-vs-actual-incidence 10)

The empirical data gives 1-4 days as the incubation period for influenza. Absenteeism occurs a day after the end of incubation. Thus, the empirical data are as follows:

(setvalue emp-absenteeism-vs-actual-incidence-lowval 2)

(setvalue emp-absenteeism-vs-actual-incidence-highval 5)

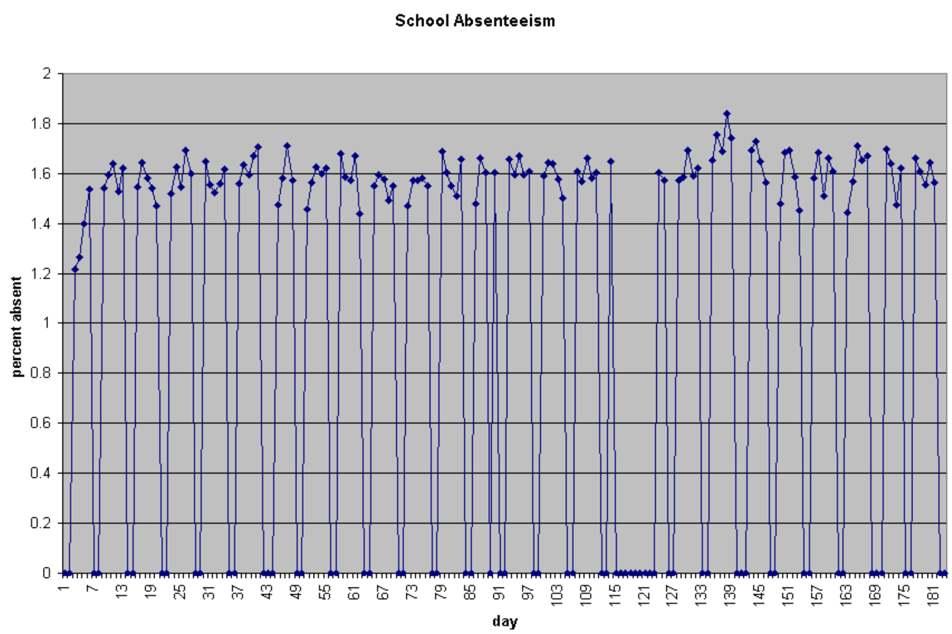


Figure 4 The Peak of School Absenteeism Occurs on D

As shown, the peak of school absenteeism occurs on Day 138. The curve is discontinuous on Saturdays and Sundays because schools are closed. Days 115-121 are holidays.

The Inference Engine compares the relative timing of absenteeism and incidence peaks with the empirical relative timing. After conflict resolution based on the priority value (here other weighting factors are not considered), it produces the following inference

(toolong absenteeism-vs-actual-incidence)

(op-higher symptom-severity)

because the absenteeism peak lags 10 days behind the incidence peak; twice as long as the empirical maximum of 5 days.

The inference is that the symptom-severity (the relative magnitude of manifested symptoms) should be increased. For the next cycle of the simulation, symptom severity is increased by 100% using an encoded rule about critical point heuristics. BioWar is re-run and then WIZER is re-run. The following figure shows the resulting school absenteeism curve.

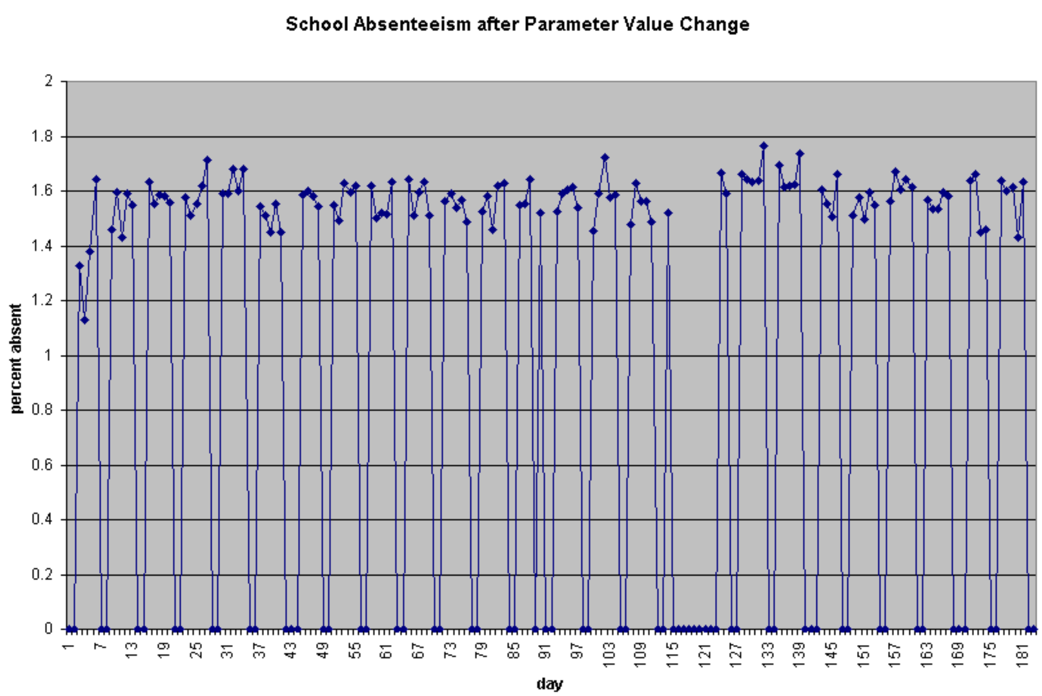


Figure 5 The Peak of School Absenteeism after Change Occurs on Day 132

As shown, the peak of school absenteeism now occurs on Day 132. The Inference Engine compares the relative timing of absenteeism and incidence peaks with the maximum empirical relative timing. After conflict resolutions are performed, it now produces the inference of:

(within-range absenteeism-vs-actual-incidence)

(op-valid)

The relative time difference between absenteeism and actual-incidence peaks is now 4 days, less than the previous cycle's relative time difference of 10 days, and now one day shorter than the maximum empirical time difference. Thus, the peak of school absenteeism has moved into the valid, empirically bounded range of 2-5 days. The Inference Engine announces that the simulated absenteeism curve peak is now valid.

The following figure shows the school absenteeism curves before and after the parameter value change.

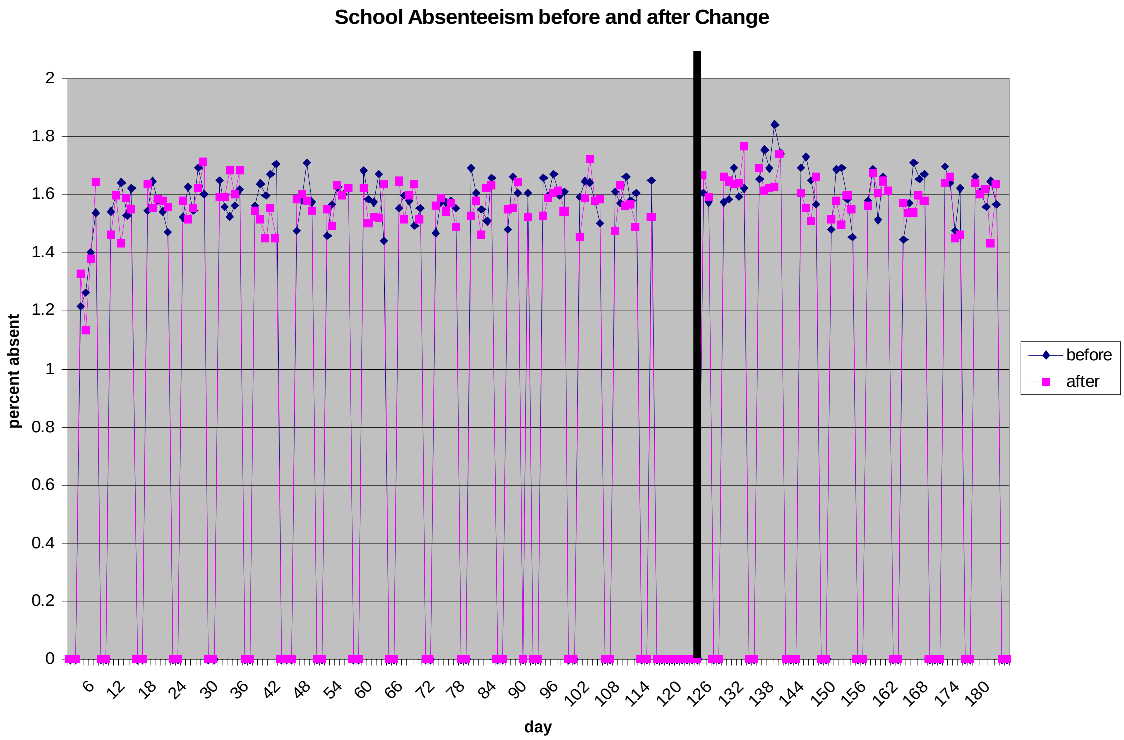


Figure 6 School Absenteeism Curves before and after Parameter Value Change

As shown, after changing the parameter value, the absenteeism peak moves closer to the incidence peak (as shown by the black vertical line).

Validation Measures

Validation is measured based on a piece of knowledge that corresponds to a data stream. For the results on school absenteeism above: initially, the simulated school absenteeism peak occurs later than it should be. Thus this data stream has zero validity, strictly speaking. After parameter values were changed by WIZER, the simulated absenteeism peak moved to within the empirical range, achieving validity.

WIZER versus Response Surface Methodology

BioWar has hundreds of parameters. The resulting parameter space is gigantic. Suppose that the Response Surface Methodology or RSM (Myers and Montgomery 2002, Carley et al. 2004) is used to completely characterize BioWar for validation. Let us assume that that BioWar has 200 parameters (a conservative number) and that each parameter can have 3 different values (3 levels), the parameter space is *3^200* cells, unmanageable with any current technology. As BioWar is stochastic, each cell requires multiple virtual experiments (assume here 40 virtual experiments) to achieve statistically significant results.

Experimenters, of course, can divide the system into modules and validate module by module, assuming all other modules have reasonable parameter values and the existence of some modularity in the system. If this is done for BioWar, experimenters can probe the relationships between incidence rate and infection factors such as *ailment effective radius* (initial infection radius), *ailment exchange proximity threshold* (person-to-person transmission of bioagents for contagious diseases) and *base rate* (initial infection rate). Assuming each of these factors has 3 levels (3 possible values) the following table shows the number of cells required.

TABLE 1 Cell number for incidence factors validation

Parameter	Categories	Size
Ailment effective radius	500, 1000, 1500 meters	3
Ailment exchange proximity threshold	500, 1000, 1500 meters	3
Base rate	10%, 30%, 50%, 70%	4

As shown, the total number of cells required is *3 x 3 x 4 = 36* for the non-stochastic case. Being stochastic, BioWar requires at least *36 x 40 = 1,440* virtual experiments.

WIZER enhances the way experimenters decide which parameters and what parameter levels to choose by codifying the knowledge in a form that is clear, explicit, and operable by computers. With its inference engine, WIZER can reason about parameters and simulation results, producing new inferences. Furthermore, utilizing its knowledge inference, WIZER can reduce the number of virtual experiments needed. The above number of virtual experiments for RSM of *1,440* is the upper limit of what WIZER needs. Typically, WIZER needs fewer due to its inferences about simulation results after each simulation cycle. The better the inferences and the knowledge bases, the fewer the number of required simulation runs.

The following table shows what WIZER gains when used for BioWar. The gain is compared against what normally transpires when humans do the validation. The numbers are first-order estimates. The time it takes for WIZER depends on computer speed, memory, and storage capacity. In addition to BioWar, we have validate a simulation model of socio-cognitive co-evolution called CONSTRUCT (Carley 1991) using WIZER with comparable performance gains.

TABLE 2 Advantages of WIZER versus manual validation for BioWar

Aspect of validation	Manual validation	WIZER
Time to generate input data	Days if not weeks, due to the data access rights, usage policy, non-disclosure rules, privacy concerns, data ownership rights, and other problems.	Days if not weeks, and longer than what it takes if done by human, as the data needs to be formatted and prepared for computer processing
Number of points in response surface that can be estimated	1 per 10 minutes	20 per 10 minutes
Number of data streams	One data stream examination per 15 minute	Many more data stream examinations (>15) per 15 minutes, limited only by computer speed
Knowledge management	Difficult	Facilitated
Number of rules processed	One per 5 minutes	300 per 5 minutes
Number of causal relations considered	One per 5 minutes	300 per 5 minutes
Selection of experimental variables	Implicit but good, depending on experience	Explicit and computer operable
Use of statistical tools	Depends on experience	Encoded in the inference
Documentation of inference and experiment steps	Need extract work	Included in the inference trace
Ability to explain simulation results	Depending on experience	Part of inference trace
Enforced precision	No	Yes
Enforced clarity	No	Yes
Man-hours	Large	Medium-to-Large
Retention of knowledge	Depends on personnel	Facilitated
Large problem solving	Possible, e.g., by careful analysis	Facilitated

ON THE CONSTRUCTION OF KNOWLEDGE SYSTEMS

While the automation of simulation validation brings efficiency, there is an upfront investment in the construction of knowledge bases and inference rules. There is also a research question of how one validates the knowledge bases and inference rules.

Our perspective on the above issues is multi-faceted. The knowledge codification in a form that is clear, explicit and operable by computers facilitates replication of simulations and their results. Replication of results is critical for scientific progress. Current practices of simulation validation (which is usually done by people who construct the simulation) leave this critical issue of replication as an afterthought. As a result, it slows down the scientific advance of modeling and simulation. The codification of knowledge is necessary to do sound engineering and science. Current codification focuses on model specifications and usually has a form of formalized English language. (Codification for code verification can be done using formal methods.) It is straightforward to require that the codification be done not just for model specifications but also for validation specifications and that it be done in a form that is operable by computers. It is a simple extension of existing activities to cover broader scope. The fact that now the codification allows computer automation will recoup some of the time and resources investment spent on codification. Needless to say, this is similar to building houses by designing detailed engineering models beforehand. People can build houses without detailed designs but this often results in a quality-problem and delays in construction. Indeed, quality does matter, not just quantity. Codification also facilitates collaboration. To ameliorate the startup investment cost, we are implementing simulation infrastructure to help modeling and simulation practitioners encode their model and validation specifications in a form operable by computers.

An investment in clear, explicit and computer-operable representation of knowledge for specification and validation is also useful because this higher-level of representation can help reduce errors in the specification and validation process. This is analogous to the fact that high-level languages such as Java helps reduce programming errors and increase programmer productivity as contrasted to the low-level assembly language or machine-level machine-code. An upfront cost here is the compiler and, in our case, the knowledge bases and inference rules. In fact, modeling and simulation itself is an investment vis-à-vis construction without models. Boeing is successful in using computer modeling and simulation in lieu of physical prototyping in the construction of its latest airplanes.

Avoiding a conflict of interests, it is a good practice to separate people or institutions who build a simulation system with those who validate it. Thus, the validation people will build their own knowledge bases for validation. Current validation process has already put validation specifications on paper. It is an extension to current activities to put those specifications in a form that is clear, explicit and operable by computers. We will have knowledge bases from validation stakeholders and from model builders. Comparing these is one way to validate the knowledge bases. The fact that we structure our knowledge bases according to causality simplifies the validation of the knowledge bases. The issue of knowledge bases validation as a whole, however, is a subject for another paper.

Another aspect of validation of large and complex simulations is the need to facilitate collaborations among diverse experts located at various locations. For this, an explicit and clear specification of simulation models and results facilitates collaborations. WIZER and cyberinfrastructure components can help researchers setup, run and replicate simulations with precision and speed.

The conventional knowledge systems have a weakness of being brittle, which means that the inferences will go awry if they are employed outside the specified application domain. It is also hard to ensure the correctness of knowledge and inferences when new rules are added. We address these issues by restructuring rules in knowledge bases using causality and by using knowledge systems strictly within their application domains. For large simulations such as BioWar, we have multiple knowledge modules representing different domains. This is similar to what happens in human problem solving: epidemiologists deal with diseases and symptoms, city health officials deal with quarantine and other response policies, police deal with how to maintain security and order, first-responders deal with how to give first-aid quickly, etc. As causality is empirical, the inferences are grounded on empirical knowledge and data. Structuring knowledge bases along causality is one way to partition the knowledge bases into smaller, more coherent and more manageable knowledge bases. A related work on scaling up knowledge bases is structure-based partition (Amir and McIlraith 2005, Ramachandran and Amir 2005). As WIZER is a knowledge-based causal system, it can scale well given appropriate knowledge including statistical knowledge. BioWar itself is a sufficiently complex model to test validation approaches: it can represent a demographically-realistic, spatiotemporally-realistic, and features-rich city with millions of people. In the real-world, statistics is used to scale economic models and market indicators are used to scale the model measurements. If we would like to have precise world-scale validated economic models/simulations, we will encounter the challenge of getting the proper data before the challenge of inference. As more and better economic data become available, WIZER can scale with the data and help build better economic models.

We have argued above that it is critical and necessary to invest in a clear, explicit and automated representation and validation of simulations because we need to have replications to do good science. It is also a good engineering practice to have design schematics covering the entirety of the system (not just model specifications but also model behaviors and results) that can be automatically checked and executed by computers. While it takes time and resources to construct knowledge bases and inference rules, Table 2 indicates that once this is done we can recoup the investment and get the dividends. The needed investment is also reduced as the construction of knowledge bases and inference rules can piggy-back the model specification activities.

DISCUSSION

WIZER is unique in that it pioneers ontological and knowledge-based inference for simulation validation and model-improvement. WIZER is a causal and logical reasoning, hypothesis building & testing, and simulation control engine with statistical and pattern recognition capabilities. It strives to employ deep and structural knowledge by employing causal and ontological reasoning. WIZER seeks to emulate scientists doing experiments and analyses via the scientific method, rather than providing another methodological approach or programming environment. The causal reasoning component provides link to empirical data and knowledge of scientific experiments.

While other toolkits such as Swarm (http://wiki.swarm.org), TAEMS (O'Hare and Jennings 1995, Lesser et al. 2004), and Repast (http://repast.sourceforge.net) provide programming environments for agent-based simulation systems, WIZER is designed to help with scientific experimentation, validation, scenario analysis, and model improvement. WIZER is able to run on top of any simulation system, including those constructed using Swarm and Repast toolkits.

The following table compares WIZER and other tools:

TABLE 3 Feature comparisons between WIZER and other techniques

	WIZER	Swarm/TAEMS/Repast	Evolutionary Strategies	Data Farming
Programming environment?	No	Yes	No	No
Unit of inference	Rule and causation	None	Evolutionary and genetic operators	Data growing heuristics
Object of operation	Simulation, data and knowledge	Code	Simulation and data	Data
Experimentation?	Yes, automated	Yes, human operated	Yes, automated (fitness)	No
Knowledge operation?	Yes	No	No	No

WIZER differs from evolutionary programming (Fogel 1999), evolutionary strategies, and genetic algorithms in that it does not need a population of mutation/crossover candidates nor does it need mutation, crossover, or other evolutionary and genetic constructs. Instead, WIZER applies knowledge inference to simulations to determine the next simulation run. If the result of inferences mandates a radical change, a revolution will occur. From the historical point of view, evolution took millions of years to affect change, while the application of the scientific method after the Renaissance advanced science and affected changes on the Earth's surface in only a few hundred years.

Our approach facilitates the integration of simulation and knowledge inference. As a simulation runs, producing perhaps emergent behaviors, simulation-based knowledge is automatically captured and analyzed. As knowledge changes, the simulation can be changed.

For social sciences, the inferential approach allows investigation of the foundations of social networks, first by the validation of agent-based systems and in future by the validation of more realistic systems (e.g., physical models). Unlike the Exponential Random Graph Model (ERGM) or p* (Robins et al. 2006) which attempts to characterize the probability of social network structures in a top-down manner using pure statistics, WIZER can be used to characterize a range of agent behaviors and the resulting emergent social behaviors from agent interactions in a bottom-up fashion and within a proper context of the application and semantics. Our inferential approach indicates a path toward more profound theories for social interactions and group behaviors.

In the scientific community, the explosion of data and the need for collaboration paved the way for cyberinfrastructure, which is an infrastructure for data acquisition, data management, knowledge sharing, visualization, and collaboration over the Internet for scientists and engineers. The linkage between simulations within cyberinfrastructures and knowledge inferences is not yet automated. Our approach suggests one way to automate the linkage and thus provide a simulation infrastructure for scientists and engineers. This may speed up the analysis of massive data sets. Social scientists, artists and humanists in particular need a simulation infrastructure to play out and investigate phenomena that cannot be described by math, logic and statistics alone.

The inferential approach underlying WIZER for simulation validation facilitates more precise research in organization and management sciences. As data become more available, aided by high performance computers, the simulations become more precise enabling more detailed theories to be built and tested.

References

Amir, E. and S. McIlraith, 2005, "Partition-Based Logical Reasoning for First-Order and Propositional Theories," *Artificial Intelligence*, 162 (1-2), pp. 49-88.

Carley, K.M., 1991, A Theory of Group Stability. American Sociological Review, 56(3): 331-354.

Carley, K. M., D. Fridsma, E. Casman, N. Altman, J. Chang, B. Kaminsky, D. Nave, and A. Yahja, 2003, *"BioWar: Scalable Multi-Agent Social and Epidemiological Simulation of Bioterrorism Events,"* Proceedings of North American Association for Computational Social and Organizational Science (NAACSOS) Conference 2004, Pittsburgh, PA, http://www.casos.ece.cmu.edu/casos_working_paper/carley_2003_biowar.pdf.

Carley, K.M, N.Y. Kamneva and J. Reminga, 2004, *Response Surface Methodology*, CASOS Technical Report CMU-ISRI-04-136, Pittsburgh, PA

Fogel, L.J., 1999, *Intelligence Through Simulated Evolution: Forty Years of Evolutionary Programming*, Wiley Series on Intelligent Systems, New York, NY.

Lesser V, K. Decker, T. Wagner, N. Carver, A. Garvey, B. Horling, D. Neiman, R. Podorozhny, M. NagendraPrasad, A. Raja, R. Vincent, P. Xuan, and X.Q. Zhang, 2004, *Evolution of the GPGP/TAEMS Domain-Independent Coordination Framework. Autonomous Agents and Multi-Agent Systems*, Volume 9, Number 1, Kluwer Academic Publishers, pp. 87-143.

Myers, R.H. and D.C. Montgomery, 2002, *Response Surface Methodology: Process and Product Optimization using Designed Experiments, 2nd ed.*, New York, NY: John Wiley.

O'Hare, G. and N. Jennings, 1995, *Foundations of Distributed Artificial Intelligence*, Wiley Inter-Science, pp. 429-448.

Pearl, J., 2000, *Causality: Models, Reasoning, and Inference*. Cambridge, UK: Cambridge University Press.

Pearl, J., 2003, "Statistics and Causal Inference: A Review," *Test Journal* 12 no. 2 (December): 281-345.

Ramachandran, D. and E. Amir, 2005, Compact Propositional Encodings of First-Order Theories, in 20th National Conference on Artificial Intelligence (AAAI'05).

Robins, G., T. Snijders, P. Wang, M. Handcock, and P. Pattison, 2006, "Recent Developments in Exponential Random Graph (p*) Models for Social Networks," *Social Networks, forthcoming*.

Yilmaz, L., 2006, "Validation and Verification of Social Processes with Agent-Based Computational Organizational Models," *Computational & Mathematical Organization Theory*, vol. 12, no. 4, pp. 283-312.

COMPARING AGENT TRAJECTORIES

H.V. PARUNAK,* S. BROPHY, S. BRUECKNER
NewVectors division of TTGSI

ABSTRACT

Sometimes it is desirable to measure the difference between the spatial trajectories of two or more agents. The naïve measure (the sum of Euclidean distances between locations at successive timesteps) increases with the lengths of the trajectories, which is not suitable for some applications. This paper explains the problem that motivates such a comparison, describes the design of the comparison that we are using, and gives an example of its application.

Keywords: trajectories, prediction, comparison

INTRODUCTION

It is often useful to invoke spatial metaphors, such as “location,” “move,” and “trajectory,” in describing agent behaviors.

Like any system, a software agent has a state, the vector of all variables that describe its condition. By analogy with the $<x, y, z>$ vector of physical location, we call the set of all states that the agent can assume its “state space,” and its current state is its “location” in that space (which may be continuous or discrete, and may or may not have a proper metric). For some agents (e.g., robots or routing agents), an important component of their state is their physical location, but it is also useful to think of an agent searching for information as having a location in “semantic space,” or of a planning agent as occupying a location in “task space.”

When agents make decisions, they often change their state, and we say that they “move” in their state space. Similarly, successive decisions constitute a “trajectory.” Again, these terms are understood literally for physically situated agents, but are applicable metaphorically to any agent.

For some applications, an agent’s trajectory is more important than its individual movements, and the set of trajectories of several agents is more important than their individual trajectories. To analyze such systems, we need to compare trajectories and characterize them collectively. This paper offers some tools for this purpose

Section 2 motivates the comparison of agent trajectories in the context of a specific modeling construct, the polyagent. Section 3 describes several measures that can be used to compare trajectories. Section 4 gives an example of using the measure.

* Corresponding author address: H. Van Dyke Parunak, NewVectors, 3520 Green Court, Suite 250, Ann Arbor, MI 48105; e-mail: van.parunak@newvectors.net.

MOTIVATION FOR A MEASURE

Our polyagent technology for predicting the future (Parunak and Brueckner 2006) represents each domain entity by multiple ghost agents, each exploring a different alternative future for the entity. For clarity, we assume that the future under consideration is a possible path through two-dimensional space, though paths through more complex structures (such as semantic networks or hierarchical task networks) can also be explored.

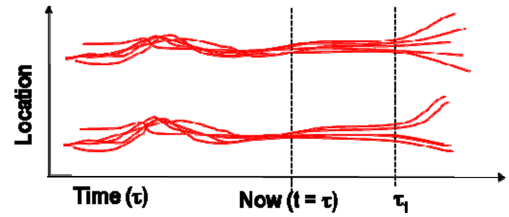


Figure 1 Trajectory analysis: Ghosts run past "Now" into the future to make predictions. The upper bundle diverges at τ_1, while the lower one bifurcates.

We wish to interpret the set of trajectories discovered by the ghosts. In particular, we are interested in characterizing their divergence over time. The probabilistic behavioral models of the ghosts emulate interactions of their entities with one another and with the environment. Since these interactions are highly nonlinear in most domains, they tend to result in phenomena such as divergence and bifurcation, and can also characterize the ghosts' environment.

Figure 1 illustrates divergence and bifurcation. Ghost time is indexed by τ and real-world time by t. Ghost simulation begins in an environment whose state corresponds to a point in the past relative to t. When $\tau = t$, we compare ghosts with the real entities that they represent, and allow the fittest ones to run into the future to form predictions. The upper bundle diverges beyond the "prediction horizon" (Parunak, Belding et al. 2007). Detecting this divergence would enable the system to avoid wasting resources on exploring further. The lower bundle bifurcates. In this case there is still predictive value in running the ghosts ahead, but detecting the branch point is crucial for understanding the system.

The degree of divergence depends heavily on the environment. For example, if ghosts are exploring possible paths for a pedestrian in the middle of an open field, they will diverge more than ghosts exploring paths for the same pedestrian at the bottom of a long, narrow valley. Distinguishing these cases can enable us to make more efficient use of the population of ghosts, and can also provide a useful characterization of the environment in its own right.

For our purposes, a measure of trajectory similarity should meet three requirements.

1. It should be independent of trajectory length, so that we can apply it across trajectory bundles of different lengths, and use it to monitor similarity as a trajectory evolves.

2. It should be tolerant of both temporal and spatial offset. Two trajectories that follow the same path but at slightly different times, or that run parallel to one another but not in exactly the same location, should be considered similar to one another, with the degree of similarity decreasing smoothly as the differences increase.

3. It should be efficient to compute. This requirement is motivated by our desire to use the measure in a real-time feedback loop to modulate the generation of polyagent ghosts.

The gold standard for measuring things is a metric, which is a function *d* from a Cartesian power of a set *X* to the reals that exhibits non-negativity, identity of non-discernibles, symmetry, and subadditivity. In general, our functions do not satisfy all of these conditions, so we call them "measures."

DEVELOPING A MEASURE

Our approach to comparing trajectories has three components: measuring the difference of a pair of paths, extending this measure to a bundle of trajectories, and converting the unbounded measure of difference to a bounded measure of similarity.

Pairwise Comparisons

We will compute our metrics on some experimental paths. Figure 2 shows the first test set: eight trajectories in two bundles, moving from left to right. Half of the trajectories in each bundle zigzag to simulate stochastic variation around the main course of the bundle. We want our distance measure to show that these two bundles separate, then converge.

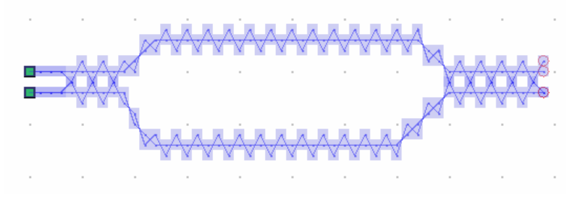


Figure 2 Eight trajectories from left to right. Four take the upper path, four take the lower. On each path, two go straight, while two zig-zag around them.

The naïve starting point for comparing trajectories is the sum of the Euclidean distances between corresponding points in the trajectories. If d_i is the Euclidean distance between the *i*th pair of points in a trajectory of length *N*, the distance is $\sum_{i=1}^{N} d_i$. When we have more than two trajectories, we take the mean of the pairwise distances. This approach is reasonable when

- All trajectories have the same number of steps
- All trajectory steps are of the same time duration
- All trajectories start at the same location

The sum of pointwise Euclidean distances is monotone nondecreasing as the length of the trajectory increases, since each additional step may add more difference. Thus pairs of long trajectories show a larger difference from each other than pairs of short ones, simply because they include more points, violating our first requirement. Some form of normalization is needed.

The obvious normalization is by the length of the trajectory, giving the average separation per step,

$$\frac{1}{N}\sum_{i=1}^{N} d_i$$

If we apply this measure in real-time, the number of items in the sum and thus the normalizing constant increase throughout the run, with undesirable consequences. Figure 3 shows the point-by-point Euclidean distances (the upper zigzag line), and the running average separation (the lower line). In the lower line, while the divergence of the trajectories is clearly marked, their subsequent convergence is much less clear, because the change is diluted by the many differences already included in the average.

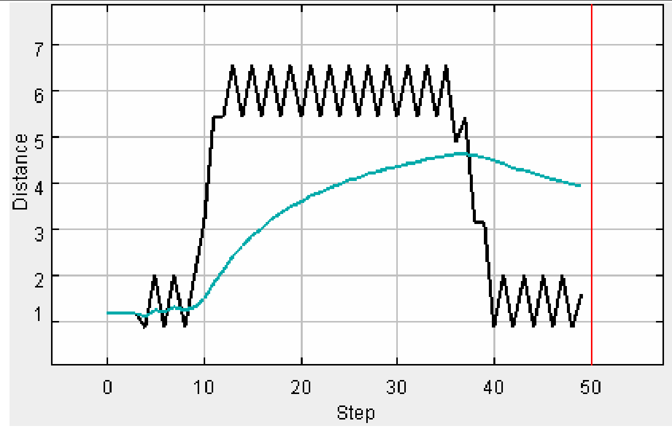


Figure 3 Point-by-Point and Running Average Distances

For monitoring trajectory proximity during execution, a running average of point-wise trajectory separations over a **Scoring Window** is more effective than an overall average**.** In our tests, a scoring window of 4 is long enough to smooth the scores. The wider the scoring window, the longer it takes for the score to reflect a change in path similarity patterns. We use the scoring window to normalize scores for the various refinements discussed below. Figure 4 shows the behavior of a scoring window of width 4 on the trajectories of Figure 2. It smooths out the zigzags and gives a distance profile that corresponds to our intuition about the overall behavior of the bundles, but it lags the actual movement of the trajectories by 2 time steps (half the width of the window).

The Euclidean measure does not recognize path pairs that follow identical routes with a small time lag as being similar, and thus does not satisfy our second requirement. Two alternative mechanisms can accommodate time lapses, step windows and the Laurinen algorithm.

The step window method uses two parameters, the **Past Step Limit** and the **Future Step Limit**, to define a window of comparison around the matching point on the paired path. For each point on one path, the distance is computed to every point on the other path that falls within this window. The shortest such distance is that point's distance from the other trajectory. Then these distances are averaged over the trajectory.

This approach captures the similarity between some lagging paths, but shows discontinuities as paths move within the window, and cannot discriminate between paths that lag at different distances if they all fall within the window. These problems result from the abrupt boundaries and arbitrary length of the window. In addition, of the four conditions for a formal metric, the step window method violates all except nonnegativity. The main culprit is asymmetry: the sum of distances of points in trajectory A to the closest points in trajectory B is not necessarily the same as the sum of distances of points in trajectory B to the closest points in trajectory A.

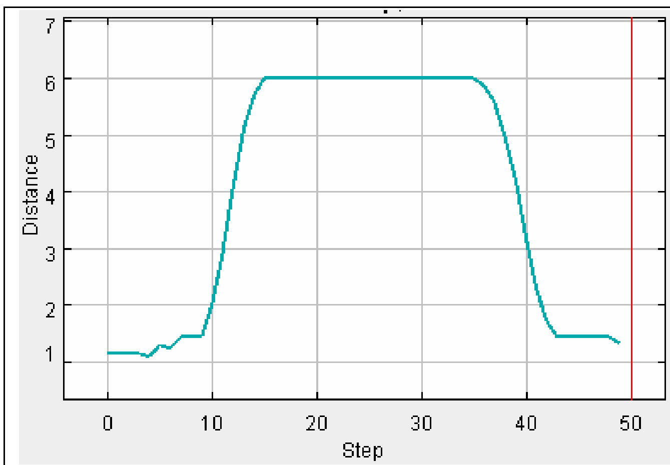


Figure 4 Running average, scoring window = 4

A more general method for aligning paths that are not exactly aligned temporally is Laurinen's algorithm (Laurinen, Siirtola et al. 2006), which explicitly includes temporal distance when measuring the separation between points on two trajectories. Applying this algorithm requires defining a mapping from time to space. We multiply the time-distance between the comparison steps by a **Step Weight Factor** and use the result as a third component in the Euclidean distance computation (along with the x-distance and y-distance components) in selecting the closest matching points between two trajectories. (The Step Weight Factor is analogous to the speed of light in special relativity, in its role of rendering space and time commensurate.) In our polyagent application, agents can move a maximum of five cells at each time step, so we set the step weight to 1/5 = 0.2. This approach allows lagging paths to score as similar, and provides a smoother function than does the step window approach.

By itself, this computation is asymmetrical, and violates the same three metric conditions as the step window method. To ameliorate the problem, Laurinen computes the distance in both directions and chooses the maximum of the two. This approach violates only the triangle inequality. In practice, in spite of this shortcoming, it is serviceable as a well-defined measure of trajectory similarity.

Figure 5 shows the effect of these two adjustments on time-lagged paths. Four trajectories (a straight one and a zigzag one for each of the upper and lower branches) are synchronized with each other. One straight trajectory for each branch is delayed by three time steps, and one zigzag trajectory for each branch is delayed by four time steps. The upper curve uses a scoring window of 4, but makes no correction for lagging, and as a result gives a higher distance (about 7) than the same measure applied to time-synchronized trajectories in Figure 4 (about 6). The lower two curves, nearly superimposed, show the Laurinen measure with step weight 0.2 (slightly higher) and past step limit = future step limit = 5. Both cases greatly reduce the penalty imposed by the time lag.

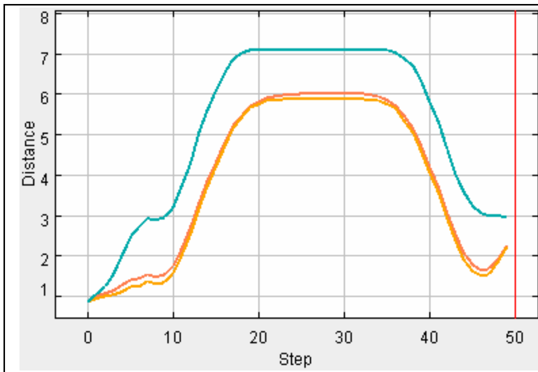


Figure 5 Distances of time-lagged paths. Top: scoring window 4. Middle: Laurinen with step weight 0.2. Bottom: past limit = future limit = 5.

Dealing with Bundles

The methods discussed so far define a similarity between two paths. In some applications, we want to characterize the tightness or looseness of a bundle of trajectories.

The naïve approach (used in the plots so far) is to average the similarity scores of all possible pairs in the bundle, requiring $O(N^2)$ operations. In keeping with our third requirement, we prefer a linear time algorithm to enable the similarity score to be used as a live feedback control. Various **Pairing Strategies** can reduce the computation while maintaining the same scoring pattern. We explored four strategies:

1. PATH_PAIRS computes all path-pair combinations ($2N(N\text{-}1)$ operations).

2. MEAN_PAIRS compares all paths against the bundle mean location ($2N$ operations).

3. INTO_MEAN measures the distance from individual paths to the mean (*N* operations).

4. FROM_MEAN measures the distance from the mean to individual paths (*N* operations).

The MEAN_PAIRS approach results in a score that follows the same trend, but is generally lower than the full PATH_PAIRS score, because the bundle mean is usually closer to a path than the score that path would get when compared to all of the paths individually.

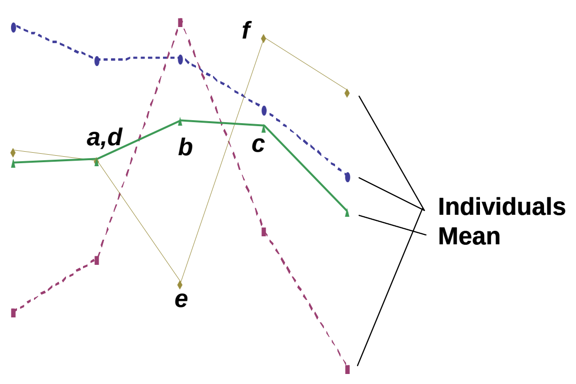


Figure 6 INTO_MEAN vs. FROM_MEAN. Points *a*, *b*, and *c* are on the mean path; points *d*, *e*, and *f* are on one of the individual paths.

Laurinen measures the difference between two paths in both directions and takes the maximum. PATH_PAIRS and MEAN_PAIRS follow this convention (thus the factor of two in the number of operations). Notice the impact of this convention when reasoning with mean paths. INTO_MEAN uses only the components of the score from each individual path to the mean, while FROM_MEAN uses only the components from the mean to the individual paths. The mean path naturally tends to be straighter than the individual paths, resulting in a systematic difference between INTO_MEAN and FROM_MEAN.

Consider comparing the mean path with a path that mostly follows the group, but loops out and then back into the bunch (Figure 6). First, consider the INTO_MEAN score from point *e* on an individual path to the mean path. All of the nearest points (*a*, *b*, or *c*) on the mean path are far away. But in computing the FROM_MEAN score, points *a*, *b*, and *c* will find close points on the individual trajectory (*d*, *d*, and *f*, respectively), and their relatively large distance to point *e* will never enter the computation.

Figure 7 shows all four scores for the trajectories of Figure 2.

This observation enables a further efficiency. Since MEAN_PAIRS uses the larger of the INTO_MEAN and FROM_MEAN scores, and since INTO_MEAN is usually larger than FROM_MEAN, INTO_MEAN is an efficient surrogate for MEAN_PAIRS. However, one may prefer to use FROM_MEAN instead, for the following reason. The mean over a set of trajectories tends to smooth out their individual variations, and so FROM_MEAN automatically smooths without the time lag imposed by a scoring window. Figure 9 compares the FROM_MEAN scores with windows of 1 and 4. As the distance increases relative to the variance, the measure with scoring window of 1 (the left-most curve) becomes almost as smooth as that with a window of 4 (to the right), and without the lag.

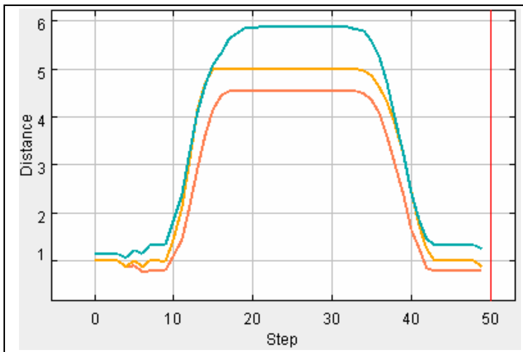


Figure 7 Bundle scores. Top: PATH_PAIRS. Middle: INTO_MEAN = MEAN_PAIRS. Bottom: FROM_MEAN.

In applications, the use of the mean both to compute the baseline trajectory and to combine the differences of individual trajectories from the baseline is sensitive to outliers, and in practice we prefer to use medians for both of these computations.

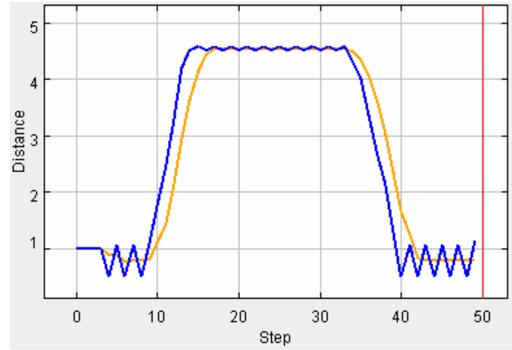


Figure 9 Smoothing effect of FROM_MEAN

Similarity Calculation

Our measures so far are unbounded upward. It is often more convenient to have a measure that is bounded (say, in [0,1]). The naïve transform, to compute the similarity as the inverse of the distance, 1/d, would work if our separations were always > 1. When computing the distance for a bundle, rather than just a pair of paths, or when using a scoring window, the distance can be < 1, resulting in similarity scores > 1. Several approaches are possible.

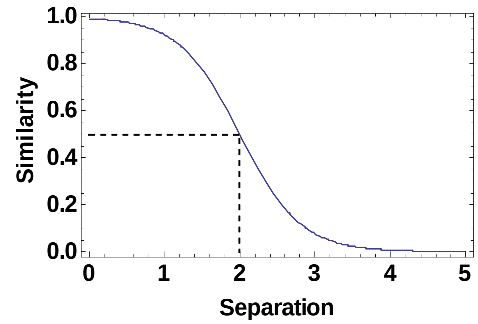


Figure 8 Sigmoid Transformation, for *steep* = 2.5 and *offset* = 2

We could define the similarity to be 1 for any distance < 1. The step function generated by this approach loses information as to whether the computed bundle distance is increasing or decreasing for small separations.

We could scale the similarity as $N/(N+d)$. This transform avoids the step function, and raises the values to use more of the 0 to 1 range. But the shape of the curve still drops off too quickly for distances that should all be close to similar.

The transformation we have found most satisfactory is a sigmoid (Figure 8),

$$similarity = \frac{1}{1+e^{-steep(offset-d)}},$$

Offset determines the distance that is mapped to a similarity of 0.5, and *steep* determines the steepness of the transform at that point. For our test cases, *offset* = 2 and *steep* = 2.5 closely follow the naïve $1/d$ transformation. Figure 10 shows the similarity obtained by this transformation from the INTO_MEAN measure with scoring window of 4.

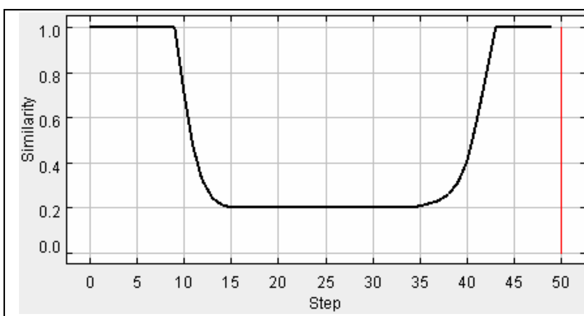


Figure 10 INTO_MEAN distance for Figure 2 transformed to similarity

USING THE MEASURE

This section analyzes some actual ghost trajectories from a military scenario that shows the effect of the environment on their movement. The terrain includes both open areas and roads. When ghosts are on a road, they prefer to follow it, but in open terrain they move more freely. Our plots do not show terrain features explicitly, but we will describe them for the examples we discuss.

In addition to plotting the INTO_MEAN similarity score, we also plot the option set entropy (OSE). Our similarity scores are global measures, appropriate for centralized use in managing a polyagent system, but not accessible to individual agents. An agent can monitor its option set entropy locally. So the relation between these two characteristics is of great interest.

In this application ghosts live on a square lattice, and make their choices stochastically, spinning a roulette wheel with as many segments as they have next possible steps (the "option set") whose segments are weighted in the following fashion. First, the ghost combines a number of environmental signals ("digital pheromones") from each option that it may choose into a single attractiveness score for that option. In our application, the options are the cells to which the ghost may move in the next step. Then, to adjust the degree of determinism in the system, we map the attractiveness to a probability using the Boltzmann distribution,

$$p_i = \frac{e^{w_i/t}}{\sum_i e^{w_i/t}}$$

where w_i is the attractiveness of the ith option, p_i is the probability of moving to that option, and t is the Boltzmann temperature. When t is large compared with w_i, each option has an equal chance of being selected. When t is small, the choice becomes more deterministic in favor of the most attractive option.

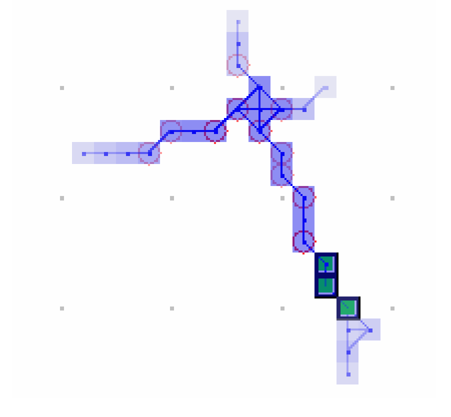


Figure 11 Ghosts on a road system

The entropy over the option set probabilities, normalized by the log of the number of possible steps, reflects how much guidance the ghost has at that step. This option set entropy (OSE) varies from 0 when the ghost is moving deterministically to 1 when it is executing a random walk. OSE is a good summary of how converged an agent system is (Brueckner and Parunak 2005). Might it serve as a local indicator of the convergence of an agent bundle?

Figure 11 shows 19 trajectories that remain on a road system. The trajectories all begin at the dark area toward the lower-right of the figure. Figure 12 shows the similarity[1] and average OSE across all agents for this system.

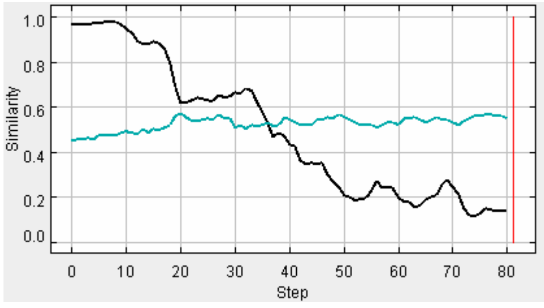


Figure 12 Similarity (descending curve) and OSE (gently rising curve) for Figure 11

[1] INTO_MEAN, Laurinen step weight 0.2, scoring window 4, transformed through sigmoid with offset = 2 and steep = 2.5.

Consider first the similarity. The trajectories diverge initially as the ghosts spread out. Then, between time 20 and 30, they come closer together, before continuing to diverge beyond time 33. At time 20 the ghosts reach the crossroads. Because they have several options available (note the small peak in OSE at time 20), they tend to loiter in the area for a few moments, and their local trajectories converge. Once each ghost converges on a road to follow out of the crossroad, the similarity again falls.

The OSE rises gently until the ghosts reach the crossroad, where it reaches a local maximum, then levels off for the rest of the run. The initial increase reflects the ghosts' initial exploration. The peak reflects their contemplation of the crossroad, and the final level portion corresponds to their constrained exploration of the various roads.

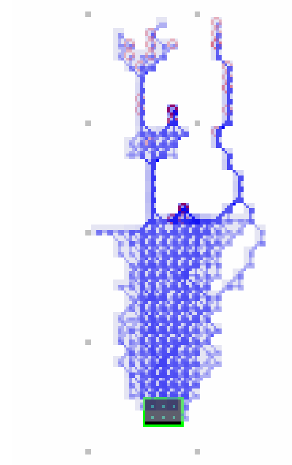


Figure 13 Shift from open terrain to roads

Now consider the 59 trajectories in Figure 13. Figure 14 shows similarity and OSE. The ghosts, again moving from south to north, begin in open terrain where they spread out, reflected in decreasing similarity. The OSE is constant during this time: the environment does not constrain the ghosts, other than a general attraction toward the roads at the north. At time 12, some trajectories discover the road emerging from the right-hand side of the main cluster, and this constraint causes similarity to level off. At time 17, further roads branch out. Because the ghosts have multiple roads from which to choose, similarity begins to drop again, while the additional movement constraints from the roads cause the OSE to drop. The rise in OSE from time 22 to 27 corresponds to the first wide spot on the left-hand road, offering ghosts more options. Because they loiter in this region, the decline of similarity is less pronounced. OSE again decreases as the ghosts follow the roads leading from this wide spot, then increases gently again after time 33, as they discover the wider set of options at the end of the left-hand road.

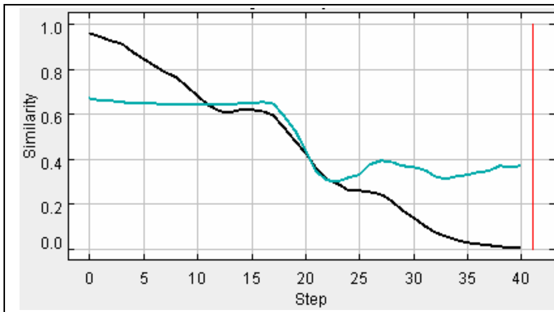


Figure 14 Similarity (steadily decreasing) and OSE (varying) for Figure 13

These examples show that while OSE and similarity are sometimes correlated, they measure different things. OSE reflects how constrained individual ghosts are, while similarity reflects how close they are to one another. All four combinations can occur. Highly constrained ghosts can be close to or far from one another, as can ghosts that experience little constraint. Correlations emerge when ghosts that are generally traveling in the same direction reach a decision point, which increases their OSE and at the same time allows them to catch up with one another, increasing their similarity.

CONCLUSION

It is often desirable to characterize the trajectories exhibited by a set of agents. In our work, these trajectories represent alternative possible futures being generated by a polyagent, and the degree to which they converge is an important index of the quality of the predictions. Such measures may be useful in other applications as well (for example, clustering targets into groups within which the behavior is similar). We seek measures that are independent of the trajectory length (so they can be used for real-time control of the agents), tolerant of both temporal and spatial offset, and efficient to compute. Naïve measures do not satisfy these requirements, but the transforms presented in this paper provide a rich toolbox that we are using in analyzing predictive trajectories.

ACKNOWLEDGMENT

This research is funded by the National Geospatial-intelligence Agency (NGA) under contract GS-35F-4912H. The views expressed in this paper are solely those of the authors, and are not endorsed by the NGA or the US Government.

REFERENCES

Brueckner, S. and H. V. D. Parunak, 2005, *Information-Driven Phase Changes in Multi-Agent Coordination*. Workshop on Engineering Self-Organizing Systems (ESOA, at AAMAS 2005), Utrecht, Netherlands, Springer. 104-119. http://www.newvectors.net/staff/parunakv/AAMAS03InfoPhaseChange.pdf.

Laurinen, P., P. Siirtola, et al., 2006, *Efficient Algorithm for Calculating Similarity between Trajectories Containing an Increasing Dimension*. 24th International IASTED Multi-Conference on Artificial Intelligence and Applications (AIA2006), Innsbruck, Austria. 392-399. http://delivery.acm.org/10.1145/1170000/1166957/p392-laurinen.pdf?key1=1166957&key2=6755105711&coll=&dl=ACM&CFID=15151515&CFTOKEN=6184618.

Parunak, H. V. D., T. C. Belding, et al., 2007, *Prediction Horizons in Polyagent Models*. Sixth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS07), Honolulu, HI. 930-932. www.newvectors.net/staff/parunakv/AAMAS07PH.pdf.

Parunak, H. V. D. and S. Brueckner, 2006, *Polyagents Model Multiple Futures Concurrently*. Social Agents: Results and Prospects (Agent 2006), Chicago, IL, Argonne National Laboratory.

Evolutionary Methods

SUGARSCAPE ON STEROIDS: SIMULATING OVER A MILLION AGENTS AT INTERACTIVE RATES

R. M. D'SOUZA *, Dept. of MEEM, Michigan Tech. University
M. LYSENKO, Dept. of Computer Science, Michigan Tech. University
K. RAHMANI, Dept. of MEEM, Michigan Tech. University

ABSTRACT

In this paper we present a new technique for simulating mega-scale Agent-Based Models (agent population sizes exceeding one million) at interactive rates. We achieve this performance by leveraging the computing power of Graphics Processing Units (GPUs). To test our system, we implemented SugarScape, a simple model with many common ABM features. We are able to achieve over 50 updates per second with agent populations exceeding 2 million on an environment with a resolution of 2560x1024 with visualization.

Keywords: ABM Simulation toolkit, GPGPU, Parallel Computing

INTRODUCTION

Due to the emergent nature of Agent-Based Models (ABMs), it is critical that the population sizes in the simulations match the population sizes of the dynamic systems being modeled [1]. In domains such as social modeling, ecology, and biology, the agent population can exceed several million. However, the performance of current agent simulation frameworks is inadequate to handle such large population sizes. Single core CPU performance has stagnated due to physical limitations. This fundamentally limits the performance of all serial frameworks for ABM simulation. Parallel computing frameworks designed to run on computing clusters suffer due to the bandwidth limitations [2]. Issues such as load balancing and synchronization can severely degrade performance [3]. Moreover, visualization is inefficient in distributed systems because of the amount data that must be communicated to the computer node that handles the display.

In this paper we investigate Graphics Processing Units (GPUs) as an alternative platform for ABM simulations. GPUs are powerful parallel processors designed to perform graphics functions such as rigid body transformations and special effects such as lighting. Driven by the 3D gaming industry, GPU computing power has been growing at a rate far exceeding Moore's law [4]. New generations of hardware have opened up more features for programming allowing GPUs to perform tasks other than graphics computations.

GPUs are appealing for large scale ABM simulations for two reasons: one is the sheer number crunching computational power. An NVIDIA GeForce 8800GTX has an average throughput of 512 GFlops [5], while a top-of-the-line Intel Xenon 2.6 Ghz quadcore processor has a theoretical maximum throughput of only 63 GFlops. More importantly, the memory bandwidth of the GeForce 8800GTX GPU is rated at 820.9 Gbps, while the Xenon is rated only at about 68 Gbps. The second major advantage is cost. The GeForce 8800GTX GPU retails at $600 while the Xenon processor costs over $1000. Additionally, Xenon processor suffers from cost overheads due to custom mother boards. Graphics cards on the other hand can be mounted on much cheaper computer hardware.

Because of the high degree of parallelism, the computational model of GPUs is very different from traditional programming. Taking advantage of the new data-parallel architecture is non-trivial and requires radically new algorithms. In this paper we investigate ABM simulation using the GPU. To the best of our knowledge, this is the first attempt at using GPUs for ABM simulation. The following sections will briefly describe the implementation of SugarScape [6], an ABM model that captures most of the behaviors of social sciences ABMs.

APPROACH

The act of forcing a GPU to perform computational labors beyond computer graphics is known as General Purpose GPU (GPGPU) programming. This technique requires a radical shift from traditional serial programming techniques [7]. While older GPUs only supported a limited set of behaviors, modern graphics cards have rich programmable functionality. By exploiting these capabilities, it is possible to run general numerical computations on specialized GPU hardware. GPUs follow a single instruction multiple data programming model, which does not fit conventional programming methods.

In GPGPU terms, textures (used to store image patterns) act as memory. Color channels in each texel (smallest data element) are used to store values of variables. Updating data values of textures is typically accomplished using shaders. In our implementation, variables for agent states are stored in agent state textures (Figure 1). Variables representing the environment states are stores in environment state textures. At each step, the environment and agent state textures are updated using pixel shaders. To make the updates iteratively continue throughout the simulations, a method called ping-ponging is used [8]. Using framebuffer objects, results of the updates is written back to another texture without communicating with the CPU. Since all computations are done by the GPU, our method is not hampered by the slower bandwidth of the connection (PCI express) between the CPU and GPU. CPU is involved only in processing some user input, and issuing rendering commands to the GPU.

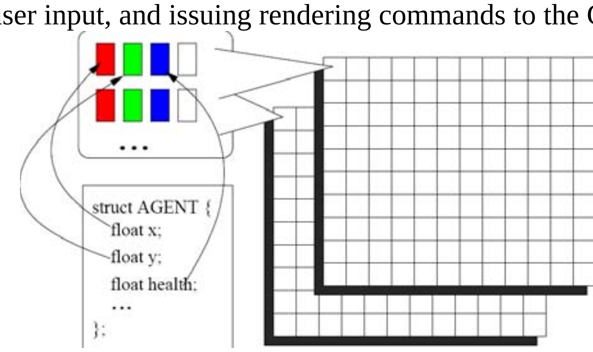


Figure 1 Agent state textures

SUGARSCAPE ON THE GPU

In this paper, we implement SugarScape to show that GPUs can be used to simulate large scale ABMs efficiently. SugarScape is an extremely relevant model since its has most of the important components of social simulation. Agents in SugarScape have a number of attributes such as vision and metabolism, and are capable of adapting to varying environments. We implemented rules G (sugarscape growback), R (agent replacement), M (agent movement), S(agent mating), P (pollution formation), and D (pollution diffusion). In the following sub-sections, we briefly explain the implementation of each of these rules.

Sugarscape Grow Back (G)

Sugarscape grow back is by far the simplest rule to implement on the GPU. To do this, we store the current level of sugar inside one of the color channels within the world texture, and the maximum sugar level within another channel. Re-growing the sugar then becomes an image processing operation, where the sugar-level channel is replenished at a given rate until it reaches saturation. Supposing that the sugar-level is stored in the red channel, while the limit is stored in the green channel, we get the following simple shader written in GLSL shader language:

```
vec4 next_sugar(vec4 prev_sugar, float regrowth)
{
      return vec4(min(prev_sugar.r + regrowth, prev_sugar.g), prev_sugar.gbr);
}
```

Movement (M)

Updating an agent's position can be performed using a pixel shader applied to the agent state texture with the current world state as input. The basic idea for this operation is similar to that used in various GPU particle system [9], with the added twist that the agents are moving according to the state of the sugar-level in the outside world texture. Assuming that the sugar is stored in the red channel of the texture, and that the agent vision range is given by the constant VISION, the following code updates the agent's position.

```
vec2 next_position(vec2 prev_position, sampler2DRect world)
{
     vec2 best_position;
     float best_sugar;

     for(int i=-VISION; i<VISION; i++)
     for(int j=-VISION; j<VISION; j++)
     {
          vec2 p = prev_position + vec2(i, j);
          float s = texture2DRect(world, p)).r;

          if(s > best_sugar)
          {
               best_position = p;
               best_sugar = s;

          }
     }

     return best_position;
}
```

A somewhat more difficult task is the problem of performing environment agent interactions. To do this, we must use a separate rendering pass to perform a scatter operation [10]. The idea is to write the agents into a separate agent collision map using a separate rendering pass [11]. From this collision map we can locate agents directly based on their spatial position, which makes environment and agent-agent interactions possible.

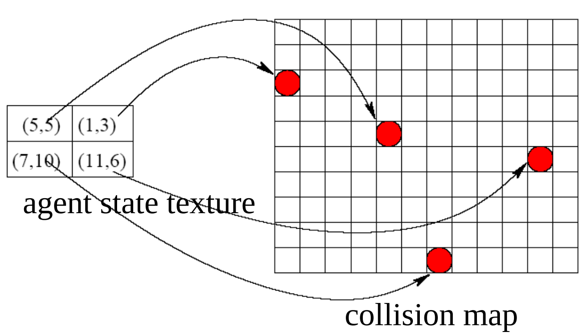


Figure 2 Agent scatter

Scattering the agents is typically performed using a vertex shader. A vertex array of indices with the same dimension as the agent state texture is initialized. Using a series of shaders, this array is then drawn into the collision map to determine the positions of each agent. There are two basic methods for scattering using a vertex shader. The most primitive is to use the render-to-vertex-buffer extension, and directly scatter the agents in such a fashion. A much simpler and faster method is available on the latest GeForce8 cards using vertex-shader-read-from-texture. With this feature, the vertex array is allocated and initialized once, and subsequent scatter passes simply read from the agent state texture as they are scattered. Figure 2 illustrates the scatter operation.

Replacement (R)

Handling agent death is accomplished using a state flag. If set to dead, then the agent is simply not updated, and not scattered during other phases. Using conditional branching, this test can be made extremely efficient. For simple replacement, the position and attributes of the agent can be randomized upon death, rather than killing off the agent. However, this strategy is not compatible with mating. Therefore, we only implement the death aspect of replacement, and allow mating to create a dynamic population.

Mating and Reproduction (M)

Agent replication is one of the most difficult aspects of any agent based model to implement properly. Making it work on the GPU is one of the key difficulties inherent in realizing efficient models. The basic problem is analogous to memory allocation. Given a new agent, we wish to place it within an empty (dead) agent cell within the state texture. A simple sequential algorithm to perform this replication is to traverse the set of all agents until an open space is found, then place the new agent into the first available memory location. Unfortunately, this is not likely to perform well for any realistic models given both the enormous number of agents, together with the enormous amount of replications per update.

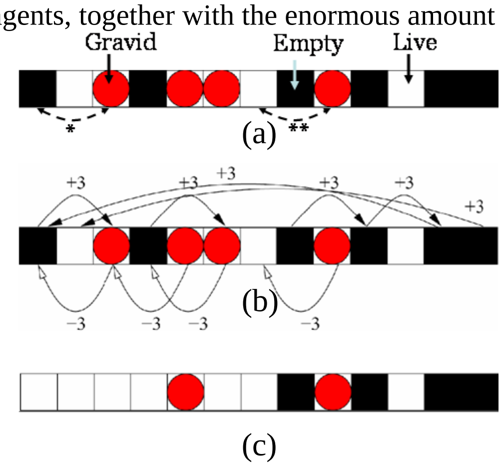


Figure 3 Stochastic memory allocation process (a) initial state (b) mapping (c) state after 1 iteration

Further improvements on the basic sequential allocation technique are possible, using objects such as freelists, but none of these are suitable for parallel allocation. In order to gain the necessary amount of speed, we use a novel stochastic parallel allocation strategy. The key to this approach is to relax the assumption that all allocations must succeed immediately. Agent replication is initiated by setting a flag within the agent state texture that signals that the agent is *gravid* or about to reproduce (Fig. 3(a)). The basic goal of the allocator is to place each newly created agent into one of the empty cells. In other words, it must match each gravid cell to a unique empty cell (Fig. 3(b)). This can be accomplished by defining a random invertible map, from the agent state texture onto itself. For the purposes of the GPU, a linear shift is sufficient. A single iteration of this technique with an offset of 3 is shown in Fig. 3. In this iteration, the map "*" is successful while as the map "**" is unsuccessful (Fig. 3(a), Fig 3(c)). Subsequent iterations with different offsets may solve this. As the number of iterations increases, the probability of success quickly converges to 100%.

Pollution Formation and Diffusion (P &D)

Implementing pollution once again requires use of the collision map. Assuming that the concentration of the pollution is stored within a separate color channel of the world texture, it is possible to blend the collision map together with the information contained in the agent state texture to add more pollution to the environment. Pollution diffusion is handled using finite differences over the background. Both processes can be carried out simultaneously within the world texture, and evaluated at the same time as the environment growback.

RESULTS

In our prototype implementation, we have achieved over 50 updates per second with agent population size exceeding two million on an environment with a resolution of 2560x1024 with visualization. Figure 4 shows a screen shot of the visualization. Figure 5 shows the scalability of our system. We are able to freely interact with the simulation as it runs, and dynamically change model parameters without any perceptible degradation in performance.

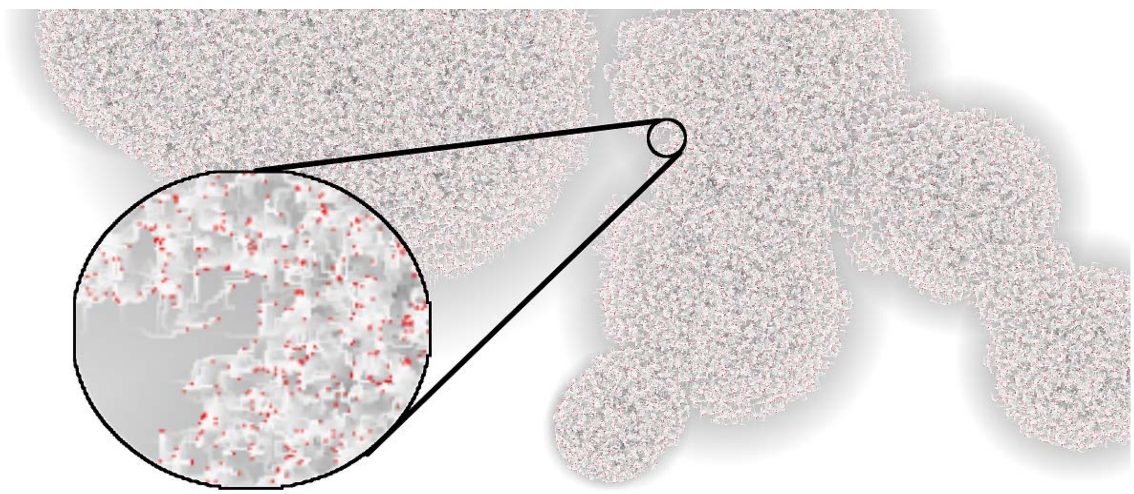


Figure 4 Sugarscape screen shot

All simulations were carried out on a single desktop. The computer hardware consisted of AMD Athlon 64 bit CPU with 1GB of main RAM running Ubuntu7.0 operating system. The graphics card is an NVIDIA GeForce 8800 GTX. The total cost of the system is under $1,400.

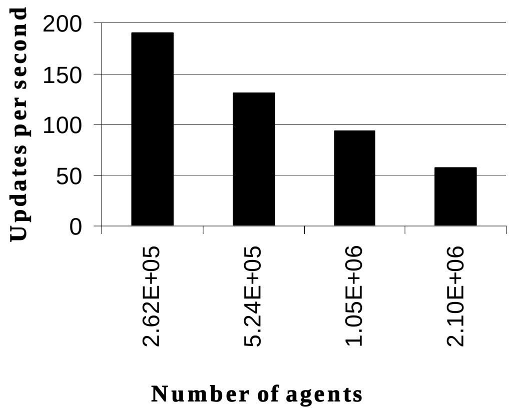


Figure 5 Scalability

CONCLUSIONS

We have successfully implemented an ABM simulation on the GPU. Our simulation runs entirely on the GPU and takes full advantage of the ultra high memory bandwidth and computational power. To the best of our knowledge, there are no single computer ABM frameworks that can deliver the performance of our prototype system. We suspect that our prototype will outperform High Performance Computing (HPC) clusters as well. Currently, statistics calculation is not implemented. However, we are working on an algorithm that is based on image histogram generation, a topic well researched in computer graphics [12]. While the simulation performance of GPUs is phenomenal, programming them is completely counterintuitive. In the future we plan to develop libraries for essential ABM functions to ease deployment of ABM simulations on GPUs.

REFERENCES

[1] Gilbert, N., Bankes, S., 2002, Platforms and Methods for Agent-Based Modeling, *PNAS*, 99(3) :7197–7198.

[2] Quinn, M. J., Metoyer, R., Hunter-Zaworski, K., 2003, Parallel Implementation of the Social Forces Model, in *Proceedings of the Second International Conference in Pedestrian and Evacuation Dynamics* (August 2003), pp. 63-74.

[3] Scheutz, M., Schermerhorn, P., 2006, Adaptive Algorithms for Dynamic Distribution and Parallel Execution of Agent-Based Models, *Journal of Parallel and Distributed Computing*, 66(8):1037-1051.

[4] Pharr, M., Fernando, R., 2006, GPU Gems2: Programming Techniques for High-Performance Graphics and General Purpose Computing, Addison-Wesley Publishing.

[5]] nVidia, (2007), GeForce 8800 Specifications/Performance.

[6] Epstein, J. M., and Axtell, R. L., 1996, Growing Artificial Societies: Social Science From the Bottom Up. MIT Press.

[7] Göddeke, D., 2005 , Gpgpu Tutorials - Basic Math, available at http://www.mathematik.uni-dortmund.de/~goeddeke/gpgpu/tutorial.html

[8] Pharr, M., Fernando, R., 2004, GPU Gem 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation, Addison Wesley.

[9] Kruger, J., Kipfer, P., Kondratieva, P., Westermann, R., 2005, A Particle System for Interactive Visualization of 3D Flows, *IEEE Transactions on Visualization and Computer Graphics*, 11(6):744-756.

[10] Sheuermann, T., and Hensley, J., 2007, Efficient Histogram Generation Using Scattering on GPUs, To appear in proceedings of *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (SI3D '07).*

[11] Millán, E., and Rudomín, I., (2006), Impostors and Pseudo-instancing for GPU Crowd Rendering. In GRAPHITE '06: *Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia*, 49–55, New York, NY, USA, ACM.
[12] Fluck, O., Aharon, S., Cremers, D., and Rousson, R., 2006, GPU Histogram Computation. *In SIGGRAPH '06: 18 ACM SIGGRAPH 2006 Research posters*, page 53, New York, NY, USA, ACM Press.

EVOLUTIONARY MULTI-AGENT TEAMS FOR ADAPTIVE OPTIMIZATION

L. HANNA, Carnegie Mellon University, Pittsburgh, PA.
J. CAGAN,* Carnegie Mellon University, Pittsburgh, PA

ABSTRACT

This paper explores the ability of a team of autonomous software agents to deal with changing optimization environments by evolving to use the most successful algorithms at the points in the optimization process where they will be the most effective. The communal agent team organizational structure employed in this work allows cooperation of agents through the products of their work and creates an ever changing set of individual solutions. An evolutionary approach is used, but evolution occurs at the strategic rather than solution level. As an application of this work, individual solutions will be tours in the familiar combinatorial optimization problem of the traveling salesman. With a constantly changing set of these tours, the team, each agent running a different algorithm, must evolve to apply the solution strategies which are most useful given the set at any point in the process. As a team, the evolutionary agents produced better solutions than any individual algorithm used.

Keywords: Evolutionary agents, adaptive optimization, traveling salesman problem

INTRODUCTION

For many complex optimization problems such as combinatorial optimization problems, exact algorithms and solution strategies for determining the optimal solution often don't exist or are so involved that they are only practical for specific applications under specific conditions. In other words, it is very difficult to determine for each possible starting point in a highly multi-modal design space, what is the best strategy for moving the solution closer to the global optimum. The conditions that motivate using specific solution strategies, if they're even known, may change rapidly as the design space is traversed.

Thus we argue that solution strategies should *evolve* dynamically as conditions change, i.e., as new solution states are discovered during the optimization process, the best strategies may be employed at the correct time to achieve maximum improvement of individual solutions. Evolution is not a new concept, but the use of evolutionary processes on the solution *strategies* is very different from typical genetic algorithms where genetic operators reproduction, mutation, and selection are usually applied to the *solutions*. Here, the solution strategies are recombined, altered, and removed through these genetic operators based on their success in improving solutions.

However, in order to ensure that a globally superior solution is obtained, evolving strategies should also be organized and coordinated in such a way that the design space is explored in as many promising directions as possible when new solutions are presented. The

* Corresponding author address: Jonathan Cagan, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA. 15213; e-mail: cagan@cmu.edu.

idea of cooperation of strategies for design space exploration, in addition to their evolution for maximum effectiveness, has led to the assertion that strategies should be embodied in independent, autonomous software agents, which evolve at a population level to determine the best solution strategy for a given set of solutions but also cooperate to more thoroughly explore the design space.

The evolution of agents, representing solution strategies, at a population level is a rather unique concept. Grefenstette (1992) explored the evolution of solution strategies for predator-prey scenarios, but with the goal of producing a single 'super strategy' from the evolution of a population of strategies (a strategy consisting of a set of decision rules) which could then be applied to the predator-prey scenario (which was the simulated to determine the fitness of individual strategies). Our aim is to evolve an efficient *team* of agents. Because of the constantly changing set of solutions, the presence of agents in the team which run inferior strategies, even in diminished numbers, strengthens the performance of the entire team: again, no one agent can accomplish what the team as a whole is capable of.

TRAVELING SALESMAN PROBLEM

The Traveling Salesman Problem (TSP) was chosen as an application for the proposed framework because it is such a well known and straightforwardly defined problem, though the goal of this work is not to present an algorithm which solves the TSP better than any other algorithm thus far. The objective of the TSP is, given a set of cities and a cost function for each pair of cities, to find the round trip tour with the lowest cost that visits each city once and only once. For the problems we will explore in this paper, the cost, or distance, function between cities is a 'pseudo-euclidean' function described by Padberg and Rinaldi (1987).

Though more successful algorithms have been developed (the reader is referred to Applegate et. al (2006) and Laporte (1992) for descriptions of the best known and most current algorithms), for this work only a few have been chosen in three categories of algorithms, *construction*, *improvement*, and *reduction*. Construction algorithms are so named because they take, as input, an incomplete or partial tour and return either a complete tour or a longer partial tour after adding cities in a predefined manner. For this study three simple and straightforward construction algorithms, nearest insertion, farthest insertion, and arbitrary insertion, were used (Golden et.al. 1985). Improvement algorithms, as their name would suggest, improve an existing partial or complete tour by rearranging the order of the cities in the tour based on different rules. There were three improvement algorithms used in this study, 2-Opt (Bentley, 1990), 3-Opt (Syslo et. al 1983), and a simple mutation. Reduction algorithms break down complete tours into partial tours. In this work, two very basic reduction algorithms are employed. The first of these is random reduction, which involves simply randomly removing a random number of cities in the tour. Best partial reduction, the second reduction algorithm, returns the best partial tour (the tour with the shortest average leg length) containing half of the total number of cities in the same consecutive order as the original tour.

METHODOLOGY

Evolutionary Agents

To perform genetic operations such as crossover and mutation, individuals in an evolving population are most easily represented by binary strings. The binary string defining an individual agent in the evolving team of the proposed framework represents the decisions the agent will make in its lifetime. We argue that decisions should be the primary element of an agent's genetic makeup because autonomy, the ability of an agent to make decisions on its own without being told what to do, is essential to the definition of an agent (Wooldridge et. al. 1995; Sachdev 1998). For the particular application of the TSP, agent decisions were defined as follows:

1. From what memory will a tour be chosen,
2. Which tour from that memory will be worked on,
3. How will the chosen tour be worked on (i.e. which algorithm will be run), and
4. Where (which memory) will the new tour be put once work is completed on it.

Thus, the genetic string of each agent consists of four binary chromosomes identifying these properties (see Figure 1). The choice methods define the characteristics of a tour which an agent will choose, i.e. if the agent will choose the best tour, the worst tour, be biased towards better tours, or be biased towards worse tours. The significance of the memories will be discussed in the next section.

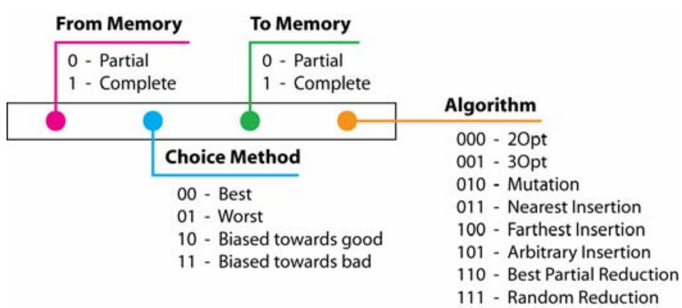


Figure 1 Structure of proposed evolving agent genetic string

Agent Organization: Creating an Evolutionary Multi-Agent Team

The agent system architecture developed is similar to the asynchronous team architecture developed by Talukdar, et. al. (1998) in that it incorporates the idea of shared memories, which allow agents to cooperate indirectly by providing a place for agents to present their work so that it is visible and available to others. However, in those systems the characteristics of each agent and the rules for their relationships to the memories are specified *a priori* (Sachdev 1998; De Souza 1993). In the proposed system the agent-memory cycles are evolved by including input space and output space decisions in the agents' chromosomal representations (toMemory and fromMemory). For the specific application of the Traveling Salesman Problem, only two memories were used: one for partial tours (tours that do not contain all of the cities) and one for complete tours. The tours in these memories evolve over time through the genetically determined actions of the agents, rather than through recombination and mutation within the population of solutions as would occur in a typical genetic algorithm (Grefenstette et. al. 1985; Potvin 1996).

ALGORITHM DESCRIPTION

Our proposed Evolutionary Multi-Agent System (EMAS) algorithm simulates temporal asynchrony by dividing the overall process into discrete iterations. Each iteration, all agents undergo *activation*, at which point they make decisions and perform actions based on their genetic sequence. After all agents have been activated, *reproduction* occurs, in which parents are selected and new agents are created. Reproduction and activation both involve a simple operator for *mutation*. Finally, the agent community undergoes *selection*, where the weakest individuals are removed from the population. In this section, each of these important functions is discussed in detail.

Mutation

In the proposed framework, mutation is used for two purposes. The first purpose, common to most evolutionary and genetic algorithms, is to make the system more stochastic – mutation allows a more thorough exploration of the design space for individuals by introducing randomness into their creation. In the proposed framework individuals are also mutated when they are not being successful. This secondary mutation is a way of allowing individual agents to adapt to an environment by trying new decision methods, achieving diversity by variation. Both types of mutation are random, meaning that a single randomly chosen bit is altered in the binary gene.

Activation

Each iteration, all agents are activated. Activation of an agent consists of verification that it is able to work (some memory-algorithm combinations are incompatible, i.e. construction cannot be performed on a complete tour) and simulation and testing to determine if it will make a positive difference. Simulation is an important step: agents will not place a solution they know will decrease the average solution quality into their destination memory. This keeps the quality of solutions in the memory high (i.e. keeps the average tour length low). As stated earlier, if an agent is unsuccessful, i.e. unable to improve the average solution quality, after three tries, the agent undergoes mutation. If an agent is successful in coming up with a solution that increases the average solution quality, it then inserts the new tour into its destination memory. A flowchart of agent activation is shown in Figure 2.

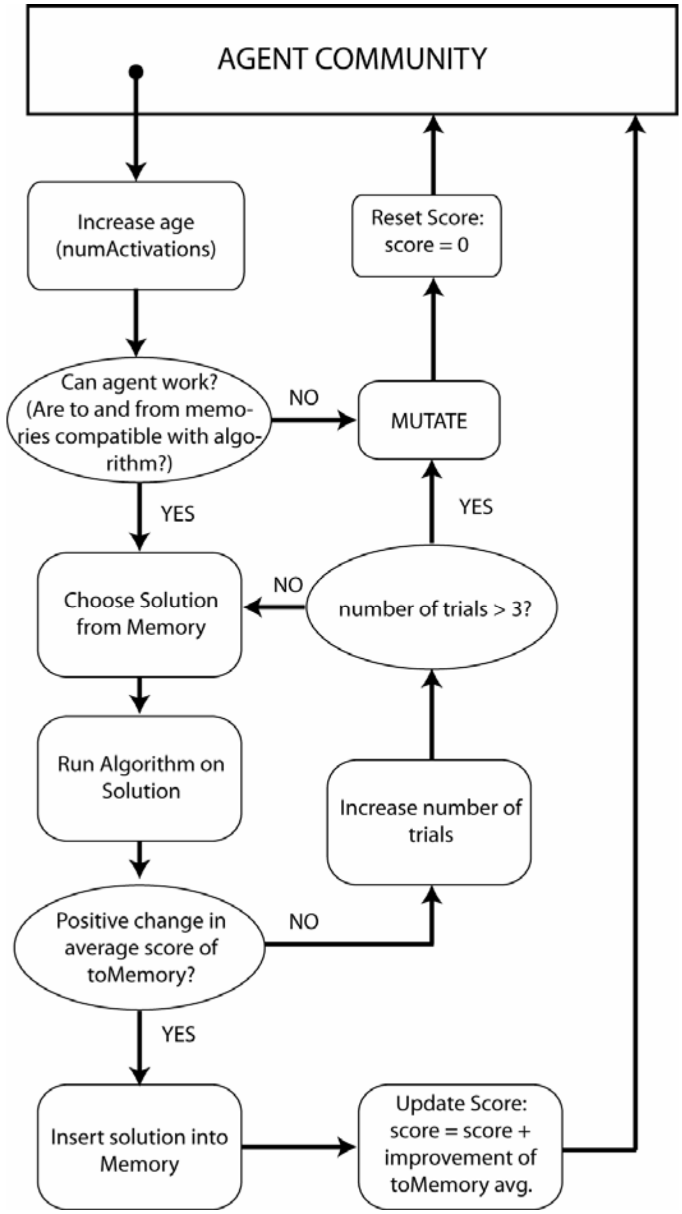


Figure 2 Flowchart of agent activation

Fitness

A key principle in both selection and reproduction is the concept of fitness. It is often difficult to establish a meaningful method for deciding who should live and reproduce and who should not. Thus, before going into detail on the procedure for reproduction and selection, it is important to establish the method of evaluation of individuals. In the proposed framework, the indication of an agent's success is embodied in its score. Score is based both on the amount of improvement made by the agent to the average solution quality in its destination memory and the number of times it has been activated (its 'age'). When agents mutate, their score is reset to zero but their age remains the same.

Reproduction

After activating each agent in an iteration, agents with a score above zero are paired up as parents and allowed to reproduce. Each agent may only reproduce once in an iteration, and during reproduction is subjected to crossover with a randomly assigned partner at a single random crossover point. The resulting two children each have a 50% chance of being mutated. After they are created, the children agents are activated.

Selection

When new agents are added through reproduction, the worst agents are selected from the population to be eliminated, keeping the population size constant. Selection begins by sorting the agents by score from lowest to highest. Agents with the same score are then sorted by age, the oldest on the bottom and the youngest on the top. Once sorting is complete, agents are removed from the bottom of the list until the population is back to its original size.

RESULTS

Though our primary goal in this work was not to develop a method for solving TSP to optimality, the quality of the solutions reached by the evolutionary team of agents proposed in this work was very good compared to the performance of the individual algorithms on their own. The solution quality reached by our Evolutionary Multi-Agent System (EMAS) algorithm were consistently better than those reached by the other base algorithms and hybrid algorithms (*a priori* designated construction algorithm followed by improvement algorithm).

The base construction algorithms nearest and farthest insertion always produce the same final optimization solution for a given starting city, so running these algorithms for each of the starting cities is a good measure of the average effectiveness of each of these base algorithms. Similarly, the same starting tour will always lead to the same final optimization solution after running any of the base improvement algorithms presented. Though the random order of city addition in arbitrary insertion makes the final tour different even for the same starting city, testing each starting city still provides a good estimation for the effectiveness of this algorithm as well. Thus, for the 48-city problem, EMAS was run 48 times (100 iterations each time) and compared to the solutions resulting from running construction algorithms from each starting city and then running improvement algorithms on the resultant tours. Table 1 clearly indicates that, on average, the solution quality produced by the EMAS algorithm is much better than any of these hybrid algorithms. Similarly, the histogram in Figure 3 shows that the majority of solutions reached by the EMAS algorithms were within 1% of optimal, whereas only two of the other hybrid algorithms had any solutions at all in that range.

Table 1 Mean and standard deviation of hybrid algorithms compared to EMAS algorithm for 48-city problem

Algorithm	Mean (% from Optimal)	St. Dev. (%)
3-Opt+NI	3.27	2.67
3-Opt+FI	3.47	1.52
3-Opt+AI	3.08	1.32
2-Opt+NI	9.61	2.32
2-Opt+FI	6.68	1.1
2-Opt+AI	6.09	2.06
EMAS	0.68	0.61

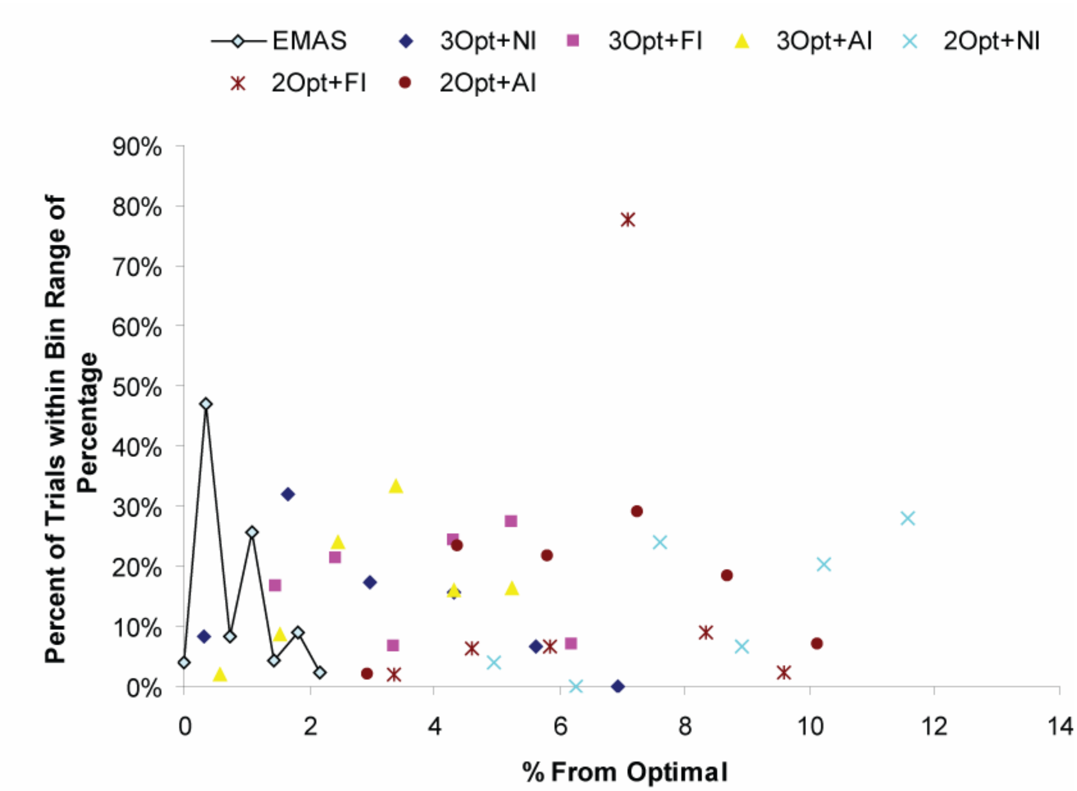


Figure 3 Histogram of 48-city problem comparing best solution consistency of EMAS to that of hybrid algorithms

The consequence of this increased quality of solutions was computation time. Because EMAS involves running several of the base algorithms each iteration, it is expected that the amount of time required to reach the solutions generated is much higher. A single run of 3-Opt on any individual starting tour for ATT48 would take less than a second, whereas a single trial of EMAS run on ATT48 for 100 iterations takes an average of around 8 seconds. As mentioned earlier, however, it doesn't matter how many times this algorithm is run on the same starting tour, it will always produce the same final tour, which as we have just shown for the base algorithms (construction only and improvement only) is usually worse than the result of the EMAS algorithm. We show in Table 3, however, that even running the same number of algorithms as would be run during a single trial of EMAS (10 algorithms * 100 iterations) in random order without employing the evolutionary aspect of the EMAS algorithm will still result in worse solutions.

Table 2 Comparison of EMAS to randomly generated algorithm activation order for ATT48 (Averages of 50 trials)

	Mean (% from Optimal of average tour in Complete memory)	St. Dev. (%)	Avg. Time (sec)
1000 Randomly Ordered Algorithms	4.83	0.354	2.45
EMAS	4.38	1.66	8.33

SUMMARY AND DISCUSSION

The results present a convincing argument for the evolution of agents in a team at the population level. Decisions have likewise proven to be a useful genetic property of agents in such an evolutionary setting. The evolutionary teams evolved to generate better solutions than the base algorithms alone. We have also shown that the strength of the EMAS algorithm lies in its ability to evolve the best team each iteration. Evolution and activation within this team results in solutions that are better than simply running the same number of algorithms randomly on a similar set of solutions. We thus argue that the use of evolutionary agents to determine the best solution strategies dynamically is a strong approach to adaptive optimization.

We have also begun to test this strategy with a much larger, 532 city TSP with even better results in terms of solution quality. In so doing we have identified patterns in how the EMAS algorithm allocates types of agents throughout its run. We hypothesize that we can take advantage of such patterns to improve run time in future work.

Acknowledgements

This research was sponsored by the Air Force Office of Scientific Research, Air Force Material Command, USAF, under grant number FA95500710225. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFOSR or the U.S. Government.

REFERENCES

Applegate, D.L., R.E. Bixby, V. Chvatal, and Cook, W.J. 2006. *The Traveling Salesman Problem*. Princeton University Press, New Jersey, USA.

Bentley, J.L. 1990. "Experiments on Traveling Salesman Heuristics," *Proc. 1st Annual ACM-SIAM Symposium on Discrete Algorithms*, SIAM, Philadelphia, PA. pp. 91-99.

De Souza, P.S. 1993. "Asynchronous Organizations for Multi-Algorithm Problems" Ph.D. Dissertation, Carnegie Mellon University, Department of Electrical and Computer Engineering.

Golden, B.L. and W.R. Stewart, 1985. "Empirical Analysis of Heuristics," in *The Traveling Salesman Problem*, E.L. Lawler, J.K. Lenstra, A.H.G. Rinooy Kan, and D.B. Shmoys, eds. John Wiley.

Grefenstette, J.J., R. Gopal, B.J. Rosmaita, and D. Van Gucht. 1985. "Genetic Algorithms for the Traveling Salesman Problem". *Proc. 1st Int'l Conf. on Genetic Algorithms*. pp. 160-168.

Grefenstette, J.J. 1992. "The Evolution of Strategies for Multi-Agent Environments". *Adaptive Behavior* **1** (1) pp. 65-89.

Laporte, G. 1992. "Traveling Salesman Problem: An Overview of Exact and Approximate Algorithms". *European Journal of Operational Research*. Vol. 59, no. 2, pp. 231-247.

Padberg, M. and G. Rinaldi. 1987. "Optimization of a 532-city symmetric traveling salesman problem by branch and cut." *Operations Research Letters* **6**, 1-7.

Potvin, J. 1996. "Genetic Algorithms for the Traveling Salesman Problem". *Annals of Operations Research*, Vol. 63, no. 3. pp. 337-370.

Sachdev, S. 1998. "Explorations in Asynchronous Teams" Ph.D. Dissertation, Carnegie Mellon University, Department of Electrical and Computer Engineering.

Syslo, M.M., N. Deo, and J.S. Kowalik, 1983. "Discrete Optimization Algorithms with Pascal Programs," Prentice Hall, Englewood Cliffs, NJ.

Talukdar, S., L. Baerentzen, A. Gove, P. De Souza. 1998. "Asynchronous Teams: Cooperation Schemes for Autonomous Agents". *Journal of Heuristics*, **4**. 295-321.

Wooldridge, M. and N.R. Jennings. 1995. "Intelligent Agents: Theory and Practice" *The Knowledge Engineering Review* **10** (2), 115-152.

THE EL FAROL BAR PROBLEM AND COMPUTATIONAL EFFORT: WHY PEOPLE FAIL TO USE BARS EFFICIENTLY

WILLIAM RAND*, Northwestern University, Evanston, IL
FORREST STONEDAHL, Northwestern University, Evanston, IL

ABSTRACT

Does how much an agent thinks about its own actions affect the global properties of a system? We use the El Farol Bar Problem to investigate this question. In this model, the El Farol Bar represents a scarce resource. Does the amount of computational ability that agents possess affect resource utilization? For instance, if agents attend the bar randomly on average 50 people will go to the bar. On the other hand, if agents act as neoclassical economics suggest, its not clear what the average attendance at the bar will be, but in this paper we will argue that it will also be near 50. In Arthur's original model, he showed, using a simulation involving an ecology of strategies, that the average attendance of the bar converged to 60. Fogel et al. gave their agents more computational power and let them use a evolutionary algorithm; they showed that the average attendance at the bar was 56, not 60. If we examine these four results of (1) random agents, (2) perfect agents, (3) Arthur's agents, and (4) Fogel et al.'s agents, we can ask whether there is a relationship between computational effort and attendance at the bar (e.g. the utilization of a public resource). To investigate this question we look at a model where we can control the computational power that each agent has to predict the attendance each week.

Keywords: machine learning, agent-based modeling, El Farol Bar Problem, genetic algorithms

INTRODUCTION

Truly adaptive agents are one of the promises of agent-based modeling, but they are rarely used. This is particularly surprising since adaptation is one of the advantages that many people list as a reason to use ABM instead of other modeling techniques. When Holland (1995) discussed complex adaptive systems (CAS) and their relationship to ABM in *Hidden Order*, he devoted an entire chapter to adaptive agents, and specifically mentioned internal models as one of the mechanisms that define a CAS, and one of the most classic agent-based models, the El Farol Bar Problem, utilized adaptive agents.

In 1994, Arthur posed a problem he called the El Farol Bar Problem. The El Farol Bar is in Santa Fe, New Mexico and on Thursday nights it plays Irish music. There are 100 people in Santa Fe who like Irish music, but if more than 60 of them attend the bar then the bar is too crowded and no one enjoys the bar. If everyone attends the bar randomly, i.e. each Thursday they flip a coin to decide if they should attend, then the bar will be underutilized since on average 50 people will go to the bar. In other words, if the agents spend no computational effort, the bar is underutilized. On the other hand, we could consider what would happen if the agents act as neoclassical economics suggest and each agent does their best to predict the attendance of the bar (i.e. uses an infinite amount of computational effort). In this case, there seem to be two possible results, either (1) each agent predicts exactly the same attendance at the bar, in which case every agent will either go to the bar or stay home, or (2) since there is an infinite number of

* *Corresponding author address*: William Rand, Northwestern Institute on Complex Systems, 600 Foster Street, Evanston, IL 60208-4057; e-mail: wrand@northwestern.edu.

ways to predict the next number in a finite sequence each agent will have a different prediction; assuming that half of these predictions are that the bar will be crowded and half that it will not be crowded, then half of the 'neoclassical' agents will go to the bar, and half will stay home. In either case the average attendance at the bar will be 50, and the bar will be underutilized.

Arthur (1994) suggested a third model. He gave each agent a 'bag of strategies' that they could use to predict the bar attendance and every week the agents used the strategy that would have worked the best had they used it in the previous weeks. Arthur showed, using a simulation, that the average attendance of the bar using this 'ecology' of strategies wound up being 60 and thus under this model the bar was maximally used. At the end of his paper, Arthur speculates that if he had used an evolutionary algorithm instead of the 'bag of strategies' technique that the results would have been similar.

Fogel et al. (1999) decided to take up Arthur's challenge and re-ran the El Farol simulation but allowed each agent to use an evolutionary algorithm with 10 strategies that evolved over 10 generations each week. Fogel et al. showed that in their model the average attendance at the bar was 56, not 60. The agents were doing better than random, but the attendance was not at 60, as Arthur had originally suggested.

If we examine these four results of (1) random agents, (2) perfect agents, (3) Arthur's agents, and (4) Fogel et al.'s agents, we see that there is a different amount of computational power being employed and a different average attendance at the bar. Is there a relationship between computational effort and attendance at the bar (e.g. the utilization of a public resource)? We will investigate this question in this paper. We begin by examining some related background to the question at hand. From there we pose a possible hypothesis and postulate what it would entail. We then build a model to test this hypothesis, where we can control the number of evaluations that each agent carries out each week (i.e., the amount of computational power given to each agent). We explain how this model fits within our general framework for agent-based modeling and machine learning, and we discuss the design of this model. We then present the results of an experiment where we varied the computational effort of each agent. Finally we conclude by discussing these results within the larger context of adaptive agents, and the trade-off between computational effort and resource utilization.

BACKGROUND

Arthur's original paper (1994) was more concerned with critiquing neoclassical economics than it was with investigating the particular properties of the El Farol Bar Problem. However, his research still represents one data point in our investigation into how computational power affects resource utilization. Fogel et al.'s extension (1999) of Arthur's work was more concerned with questioning the stability of the attendance at the bar and how randomness and evolution of strategies affected the long-term dynamics of the El Farol Bar Problem. Still this work has also started to answer the question of how computational ability affects resource utilization. Wolpert et al. (2000) examined how to automatically configure agents to best utilize a bar. Whereas we are interested in how computational capabilities affect the overall performance of the system when the agents do not care about the overall resource utilization, Wolpert et al. take a more engineering approach to the question at hand, and design a system that attempts to optimally utilize the bar. There has been other work on the El Farol Bar Problem (Edmonds, 1999) and its refined version, The Minority Game (Challet et al., 2005), but the relationship between computational power and resource utilization is rarely investigated.

HYPOTHESIS

Our relationship of concern is the correlation between computational power and resource utilization. Though the connection between these two variables may be impossible to discover in general, we can begin by investigating it within the scope of the El Farol Bar Problem. As we laid out in the introduction, there are four data points that we already have to help us investigate this relationship. The first is random agents, which result in an average attendance of 50. The second is neoclassical agents, which we argue will result in an average attendance of 50. The third and fourth data points are Arthur (1994) with an attendance of 60 on average, and Fogel et al. (1999) with an attendance of 56 on average. How exactly to relate these to computational power is difficult, but clearly random agents possess the least computational power, since they do no computation except flipping a coin, and neoclassical agents possess the most computational power, since they are assumed to have infinite resources. The results from Arthur and Fogel et al. both fall somewhere between these two extremes. If we think of computational resources as the number of evaluations that we allow each agent to perform on its pool of strategies with the current history of attendance then we can actually quantify this resource. In Fogel et al.'s case this is easy. Each agent runs an evolutionary algorithm each week in which it evaluates 20 strategies (10 parent strategies and 10 child strategies) for 10 generations, resulting in 200 evaluations. In Arthur's case this is more difficult since he does not specify how many strategies each agent has in his original paper, but he lists as example numbers of strategies, 6, 12, and 23. Since Arthur's model does not create new strategies and just evaluates the extant strategies, then we can use 6, 12, or 23 as an approximation to the number of evaluations Arthur's agents carry out each week. Regardless Arthur's agents use less computational resources each week than Fogel et al.'s but more resources than the random strategy. We can graph these (rough) data points on a figurative plot (see Figure 1).

In Figure 1, there is a line representing a possible relationship between the number of strategy evaluations and the average attendance at the bar. This line is a hypothesis, but it is a reasonable hypothesis. The general reasoning is that if the agents have too little computational power then their behavior is essentially random. However, if the agents have too much computational power then their predictions start to resemble each other. Since the number of data points that the agent is using to evaluate each strategy is small (usually around 10), there is only so much data that the agents possess. As a result, after a certain point additional refinement of strategies does not result in an improved prediction. If agents were able to remember which strategy they had used in the past and make sure and choose a different strategy then the results might be different, but in the current model if the agents have found a strategy which correctly predicts the previous attendance then they will stay with it regardless if the same strategy failed them in the past. As a result, in the end the strategies of all agents will start to look similar. The more similar the agents' strategies look, the more likely all of the agents are to take the same action. In the extreme, if all agents take the same action then each week they will all go to the bar or all stay home, assuming they stay home or go to the bar with a uniform probability, this will result in an average attendance at the bar of 50.

On the other hand, if the agents are boundedly rational, and only possess a limited amount of computational power their strategies will likely be very different from each other. This will result in each agent choosing a strategy that works fairly well, but if also likely to be different than the other agents. This will create the 'ecology' of strategies that Arthur discusses. Given limited computational power it is unlikely that the agents will find an optimal strategy for the past *n* weeks and thus they will all converge to suboptimal solutions. The hypothesis expressed by Figure 1 is that this heterogeneity of solutions in boundedly rational agents will result in some agents attending but not all of them.

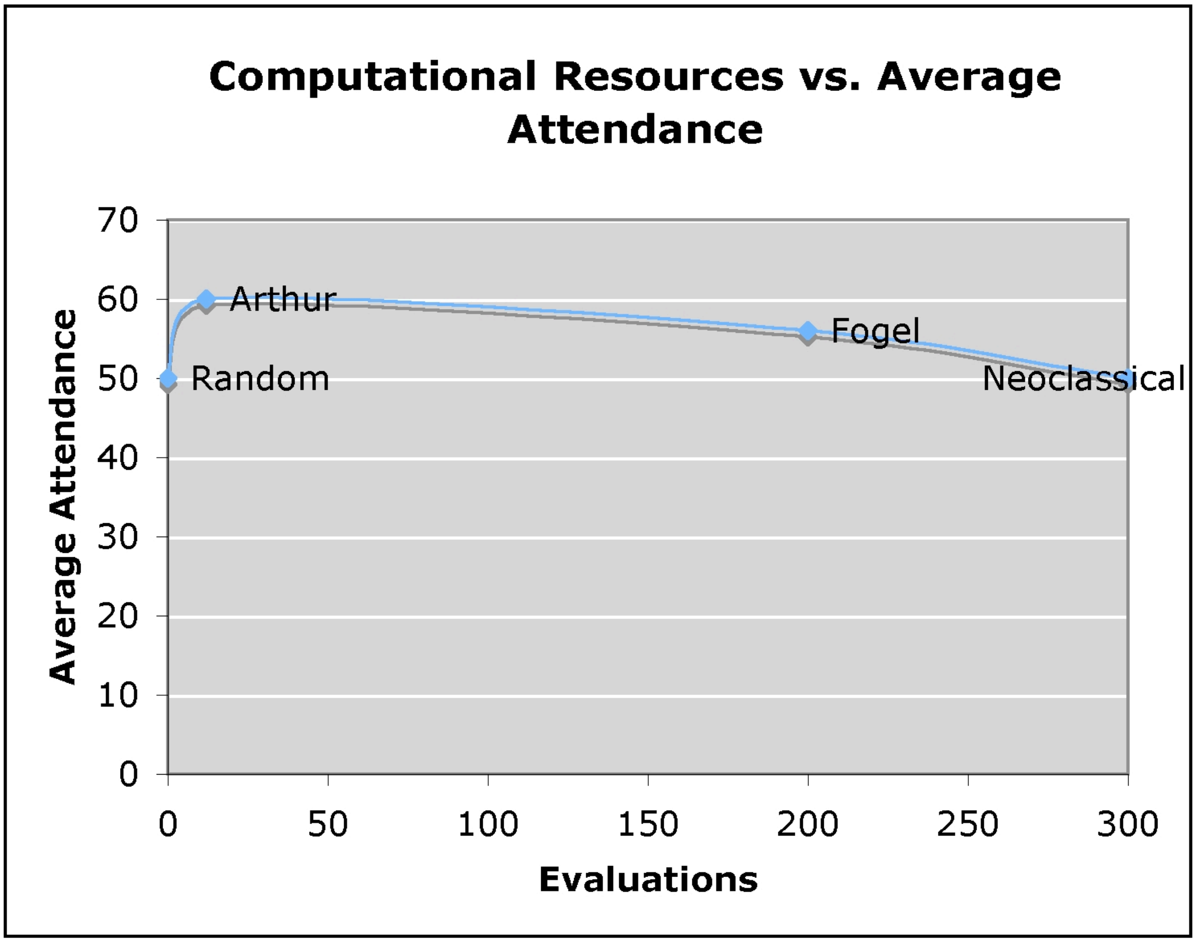


FIGURE 1 Computational Resources vs. Average Attendance.

To investigate this hypothesis we examined the El Farol Bar Problem with a group of agents where we could control the number of evaluations that each agent carried out each week. Before we get to the details of how this experiment was carried out, we will examine how we placed this model within the larger ABM-Machine Learning framework that we previously developed (Rand, 2006).

FRAMEWORK

At a high level, ABM and Machine Learning (ML) (Mitchell, 1997; Hastie et al, 2001) utilize fairly simple algorithmic structures to control their flow of operation. Roughly these algorithms can be described as: initialize the system, observe what is happening, refine the system, take actions, and repeat until time is up. As a result it is easy to examine how these systems can be integrated. Let us use the El Farol Bar Problem as an example.

Arthur's original model included a simple ML technique in it. In Arthur's model all the agents had a group of strategies. They would take this set of strategies and see which strategy would have done the best of predicting the bar attendance if they had used it in the past. Since at each time step a new data point is generated it is possible that the actual strategy from the group of strategies that each agent will use can change at every time. This is a very simple ML technique. Initialize the population of strategies by generating some random strategies, like take last week's attendance double it, subtract the third to last week's attendance from last weeks, or take a running average of the last three weeks attendances. Then at each time step the algorithm observes how the strategies have done on the current set of training data, i.e. the previous bar attendances. After that, the algorithm refines the internal model by selecting the best strategy given the new data. Finally the algorithm acts on the strategy that reflects the refined model and repeats. As Arthur speculated (1994) and Fogel et al. showed (1999), the original El Farol Bar algorithm of a 'bag of strategies' could be replaced by another standard ML technique.

We wanted to make use of a different ML technique than the one Arthur described. There are many different ML techniques and there is no obvious best technique, but partially since it was originally suggested in Arthur's paper, we decided to investigate the use of an evolutionary algorithm, and employed the genetic algorithm (GA) as originally devised by Holland. As we have mentioned, Fogel et al. had previously explored a similar technique within the El Farol Bar Problem. The GA makes sense in this context because it has the ability to create a fairly robust time series predictor (by doing simple regression) and it is similar to Arthur's original technique, in that it considers a population of solutions, evaluates them, and decides which strategy to use. In addition the GA is often described as manipulating *schemata* and thus may be similar to the human process of induction (Holland et al., 1986) which is what Arthur's original model was intended to emulate. As a result of all of these factors we chose the GA.

We then placed the original El Farol Bar Problem and the GA within the context of the Integrated ABM-ML cycle that we had previously described (Rand, 2007). The result is illustrated in Figure 2.

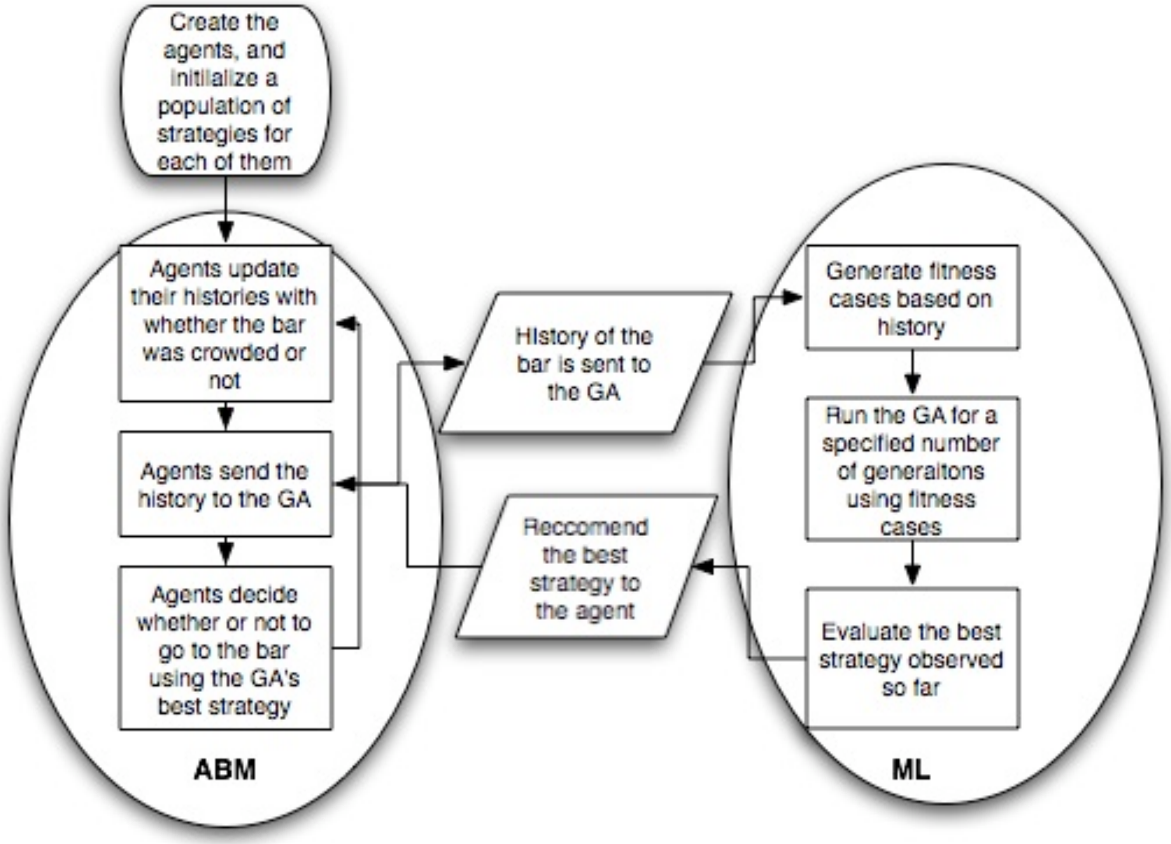


FIGURE 2 The El Farol Bar Problem and a GA within the context of the Integrated ABM-ML cycle.

EXPERIMENT

We used the framework description from Figure 2 to guide the development of an implementation of the El Farol Bar Problem in NetLogo (Wilensky, 1999). Similar to Fogel et al. (1999), we used an auto-regressive (AR) model, whereby a strategy consists of a list of real-valued numbers; these numbers are weights (in a weighted linear combination) that are used in predicting the attendance at the bar, based on attendance in previous weeks. Specifically, for a given strategy S with AR coefficients (w_0, w_1, ... w_L), where L is the number of preceding weeks considered when predicting attendance, we have the following prediction formula:

$$p(S,t) = w_0 + \sum_{i=1}^{L} w_i a(t-i)$$

In this equation, *p(t)* is the prediction for the attendance at week *t*, and *a(t – i)* is the actual recorded attendance at week *(t – i)*. For our experiment, we fixed L at 10 weeks. Each agent had its own population of 10 strategies (initialized with weights drawn uniformly at random between -1.0 and 1.0), which we evolved over time using a real-valued genetic algorithm. Our model differs from Fogel et al. (1999) in several ways. Fogel et al. created offspring solely through asexual reproduction – each of the weights in the parents' strategy was mutated by adding a zero mean Gaussian random variable with standard deviation 0.1. While our simulation also used this method for mutation, mutation was not the primary genetic operator. Each weight was mutated only with probability *1 / (2L)*. Instead, staying closer in form to the simple genetic algorithm (Holland, 1975), our primary genetic operator was crossover, whereby two parent strategies (lists of weights) are split and recombined to form new offspring. Fitness evaluation for a strategy consisted of measuring the sum of the prediction errors, if the strategy had been employed for the last *L* weeks. Each agent uses *2L* weeks worth of attendance history, so that it can perform this evaluation. Specifically, the fitness of strategy *S* at current week *t* is defined as:

$$f(S,t) = \sum_{i=t-L}^{t-1} |p(S,i) - a(i)|$$

In this equation, we employed a linear error function where any error is weighted equally. This differed from Fogel et al., who used squared error for fitness, biasing selection toward strategies that do not make egregious errors. In order to evaluate fitness in early weeks, we provided "false" historical data – that is, *a(-1), a(-2), ... a(-L)* were each initialized to random numbers between 0 and 100. We used a strictly generational GA, where a generation of 10 parents is replaced by 10 children in each generation, whereas Fogel et al.'s method evaluates 10 newly created children and 10 parents from the previous generation and chooses the 10 best strategies to create the next generation. Finally, we used tournament selection with a tournament size of 3.

The primary goal of our initial experiment was to examine the relationship between society-wide resource utilization (formalized as the bar attendance) and the amount of computational effort expended by the agents (formalized as the number of generations of strategy evolution the agents were allowed per week). Whereas Fogel et al. (1999) fixed the number of generations per week at 10, we allowed this parameter to vary from 0 to 20. Note that in the 0 generations per week case, no evolution is occurring at all. In this case, each agent has only a fixed set of 10 strategies to choose from, which is, in some respects, similar to Arthur's model (1994). We ran the model for 500 weeks per run and carried out 30 runs for each parameter setting. Fogel et al. found that the behavior of his system reached a "steady-state" of chaotic oscillation after 100 weeks, so our choice of 500 weeks seemed sufficiently large.

In several details, our experimental set up has deviated from that of Fogel et al., and it is important to note that we are not attempting to exactly replicate their experiment or results. However, by investigating the same problem and using a similar strategy representation, our work is comparable to theirs, and differs mainly in some particular details of the evolutionary algorithm, which Fogel et al. admitted were chosen somewhat arbitrarily. Thus despite these differences, a secondary goal of our experiment was to determine if our results support the general findings of the prior work by Fogel et al (by showing them to be robust despite variations in the general evolutionary algorithm).

RESULTS AND DISCUSSION

For each run, we measured the attendance at the bar in each of the 500 weeks. Since the early weeks could be skewed by the random initial conditions, we decided to concentrate on the attendance behavior during the last 100 weeks of each run. The first metric we examined was the mean attendance, as this was the quantity focused on by previous work (Arthur, 1994, Fogel et al., 1999). This is shown in Figure 3.

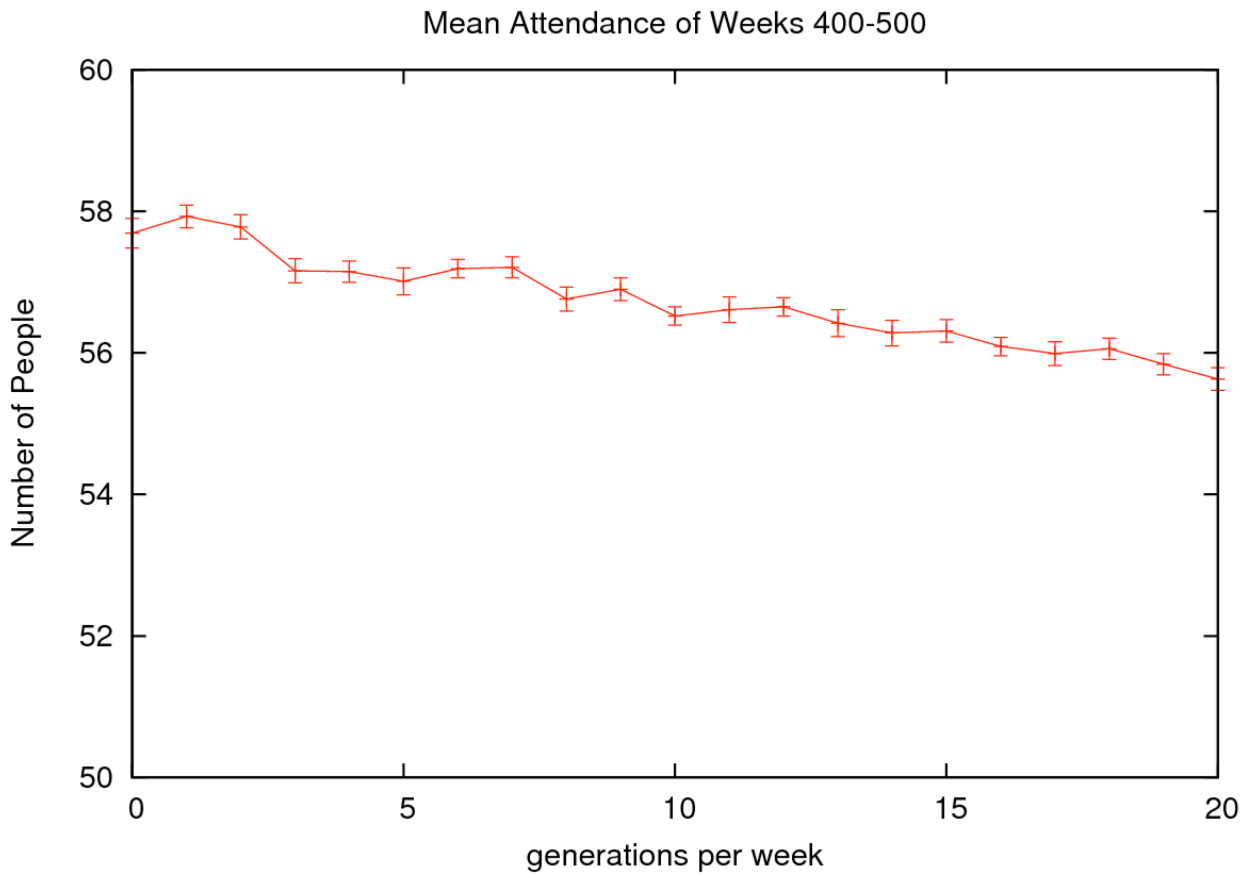


FIGURE 3 The mean (over 30 runs) attendance at the bar in the last 100 weeks versus the number of generations evolved each week. Standard error bars are shown.

Fogel et al., using 10 generations of evolution per week, found a mean attendance of 56.32. Our result for 10 generations per week was 56.52. The difference between these values is on the border of statistical insignificance, and given that we made several differing choices in our experimental setup, we feel our results support the previous findings of Fogel et al. Furthermore, our model exhibited the same non-convergent oscillations around the mean, found by Fogel et al. In addition, as we noted above, the 0 generations-per-week case is similar to Arthur's model of bounded rationality. However, as shown in Figure 3, even without evolution, our mean attendance was only 57.69, falling well short of Arthur's predicted convergence around 60.

While making comparisons to previous results is interesting, our primary goal, which was not investigated by Fogel et al. (1999) and Arthur (1994), is to determine the effect on mean atttendance of varying the amount of computational effort given to agents, i.e. the number of generations-per-week. Figure 3 generally supports the hypothesis that we explained earlier. As we increase the computational power of the agent, the average attendance at the bar decreases. As can be seen in Figure 3, the difference between consecutive points on the x-axis is not always statistically significant, but the general downward trend is statistically significant. As we explained previously, one possible explanation is that as the amount of computational power increases the agents' strategies start to converge, which drives down the average attendance at the bar. Thus Figure 3 lends some credence to this hypothesis but further work is required to confirm this hypothesis. In future work, we plan to investigate the diversity of strategies being employed across all agents.

This result was interesting, but it also prompted us to question whether mean attendance is the best metric to capture "resource utilization." For instance, consider that there are two qualitatively different attendance ranges – if 61 people attend the bar, then no one is happy, whereas if 59 people attend the bar, then 59 people are happy. It is not clear that it makes sense to simply average the values 59 and 61, when computing resource utilization. For this reason, we defined a separate metric, which we shall call "societal benefit", which is simply the number of happy bar patrons per week. When the bar is overcrowded, the "societal benefit" for that week is 0. We measured the average societal benefit across the last 100 weeks, and the results are shown in Figure 4.

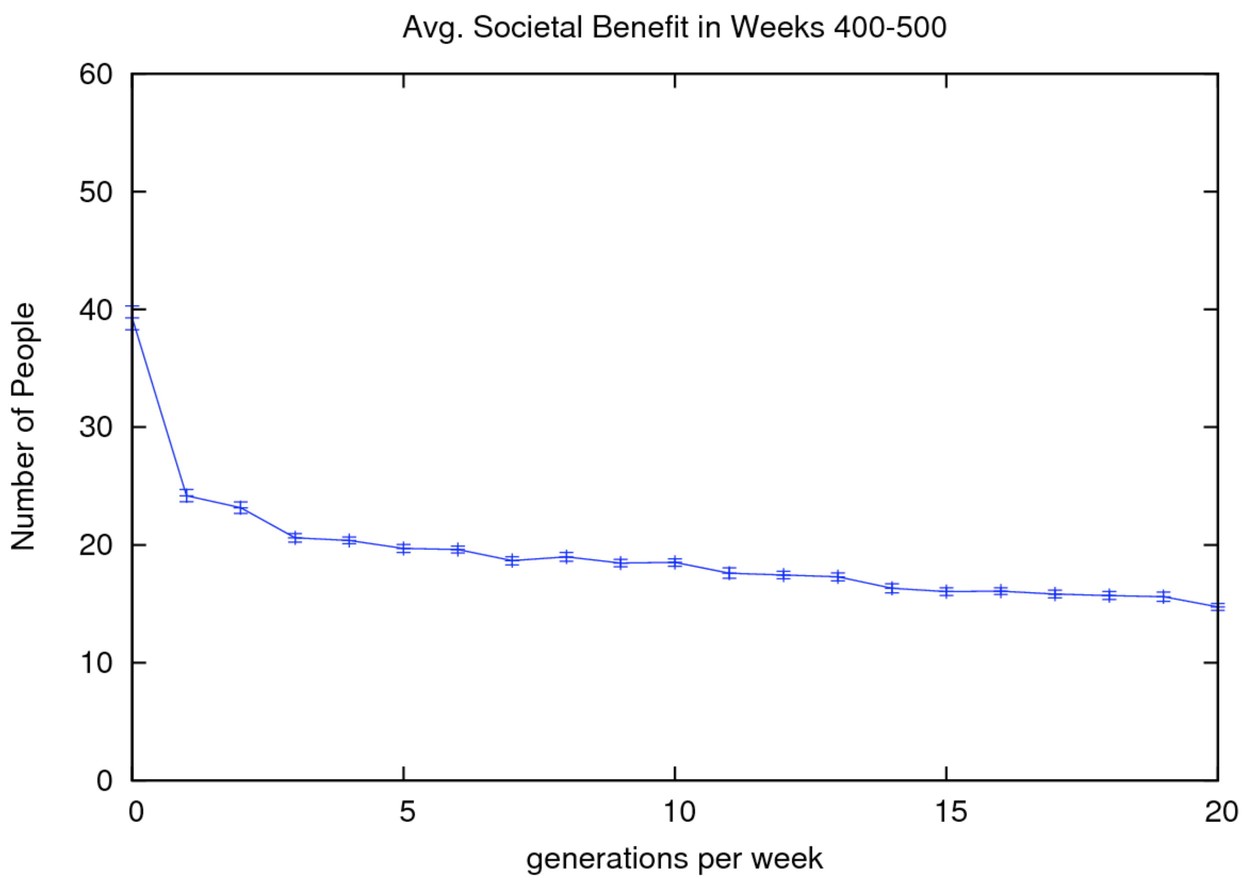


FIGURE 4 The average (over 30 runs) societal benefit in the last 100 weeks versus the number of generations evolved each week. Standard error bars are shown.

Similar to Figure 3, the relationship is that resource utilization decreases as we increase the amount of computational power given to the agents. The trend here is even more apparent. Part of this trend is obviously due to the decrease in attendance at the bar, but that is not the whole story. We know from the previous work of Fogel et al. (1999) that toward the end of the run, the average bar attendance usually fluctuated wildly. One explanation for the decrease in societal benefit might be that these fluctuations result in fewer people attending the bar when it is below capacity. To investigate this hypothesis, we examined the standard deviation of the attendance at the bar. Figure 5 illustrates the standard deviation of the mean attendance at the bar over the last 100 weeks versus the number of generations of evolution per week. As can be seen the standard deviation does increase as the computational resources increases – that is, allowing the agents more time to evolve their strategies results in a greater amplitude of the chaotic oscillation in attendance. This supports the idea that the population is fluctuating wildly and that is why the societal benefit is decreasing. However, precisely why additional computational resources causes an increase in the size of the fluctuations remains a subject for further study.

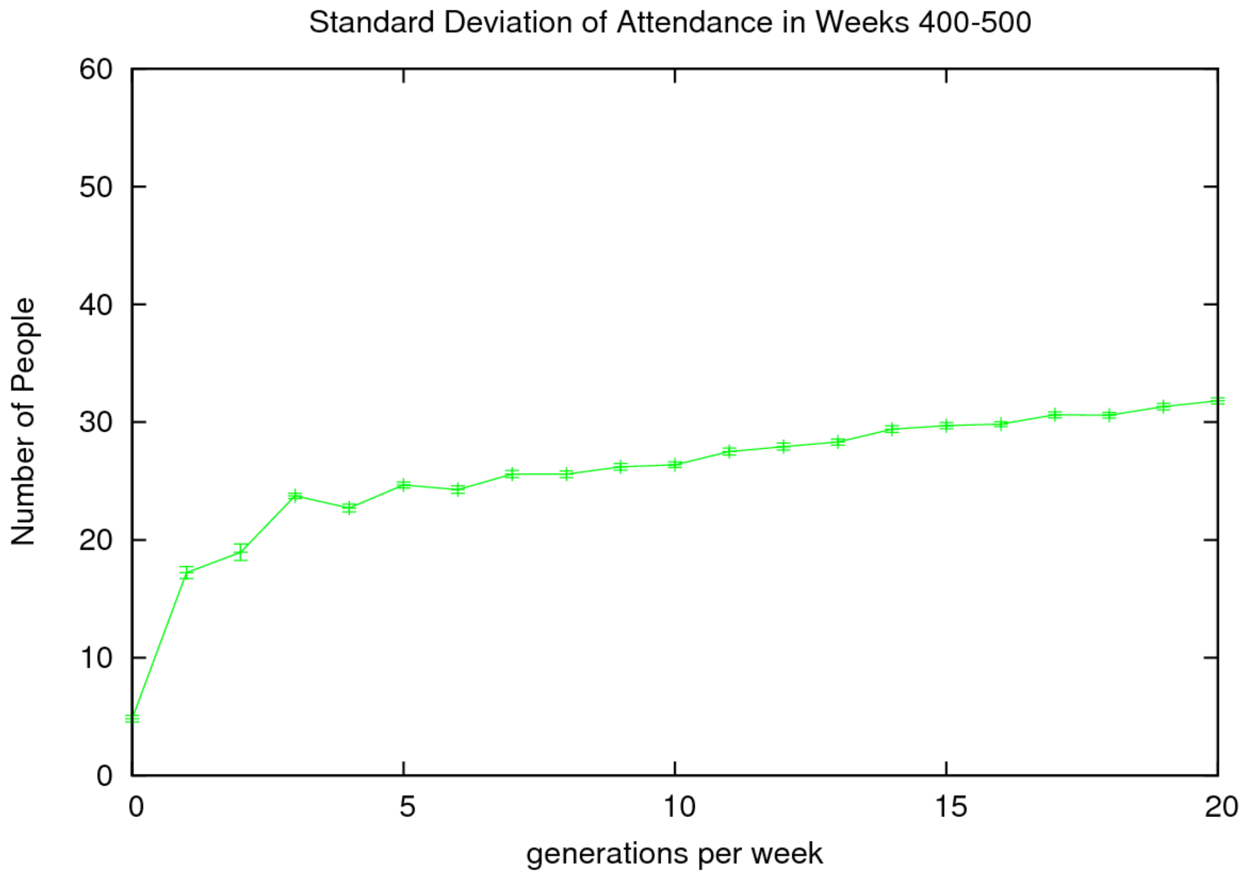


FIGURE 5 The average (over 30 runs) of the standard deviation of the mean attendance in the last 100 weeks versus the number of generations evolved each week. Standard error bars are shown.

FUTURE WORK AND CONCLUSION

These initial results are tentative, and there is more work that needs to be done to substantiate the hypotheses that we have suggested. For instance, the number of generations per week might not be the deciding factor that influences the societal benefit and the mean attendance at the bar. Another possible factor is simply the total amount of evolution. For instance, in the 5 generations per week case at week 500, each agent has undertaken 2500 evaluations, while in the 20 generations per week case at week 500, each agent has undertaken

10000 evaluations. One possible explanation is that the behavior of the 5 generations per week case at week 500 is comparable to the 20 generations per week case at week 125. Initial investigations indicate this is not the case, but additional verification is warranted. Investigating the diversity of the final strategies employed by the agents would also help substantiate some of the claims that we have discussed. Finally, it would be useful to look at the entire attendance distribution histogram rather than just aggregate measures like means and standard deviations.

Still, we have begun to investigate what the relationship between computational power and efficient resource utilization is. This experiment is a first step toward understanding if there are general claims to be made about this relationship. These tentative results indicate that it might be possible for simple machine learning algorithms to be given a limited amount of computational power and still achieve an ecology of strategies that produces a greater resource utilization than a more complex learning algorithm with greater computational power.

REFERENCES

Arthur, W. B. (1994). "Inductive Reasoning and Bounded Rationality." The American Economic Review **84**(2): 406-411.

Challet, D., M. Marsili and Y. Zhang (2005). Minority Games. Oxford, UK, Oxford University Press.

Edmonds, B. (2000). “Gossip, Sexual Recombination and the El Farol bar: modeling the emergence of heterogeneity.” Journal of Artificial Societies and Social Simulation. 2 (3).

Fogel, D. B., K. Chellapilla, et al. (1999). "Inductive Reasoning and Bounded Rationality Reconsidered." IEEE Transactions on Evolutionary Computation **3**(2): 142-146.

Hastie, T., R. Tibshirani, et al. (2001). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. New York, Springer.

Holland, J. H. (1975). Adaptation in Natural and Artificial Systems, MIT Press.

Holland, J. H. (1995). Hidden Order: How Adaptation Builds Complexity. Reading, Massachusetts, Perseus Books.

Holland, J. H., K. J. Holyoak, et al. (1986). Induction, MIT Press.

Mitchell, T. M. (1997). Machine Learning, McGraw-Hill.

Rand, W. (2006). “Machine Learning meets Agent-Based Modeling: When no to go to a Bar.” In the Proceedings of Agent 2006. Chicago, IL.

Wilensky, U. (1999). NetLogo. Evanston, IL: Center for Connected Learning and Computer-Based Modeling. http://ccl.northwestern.edu/netlogo.

Wolpert, D. H., K. R. Wheeler, et al. (2000). “Collective Intelligence for Control of Distributed Dynamical Systems.” Europhysics Letters. **49** (6), p.708.

Thursday, November 15, 2007

Toolkits Track

Parallel Track II

Toolkit Surveys

ANATOMY OF A TOOLKIT:
A COMPREHENSIVE COMPENDIUM OF VARIOUS AGENT-BASED MODELING TOOLKITS ON THE MARKET TODAY

C. NIKOLAI*, University of Notre Dame, Notre Dame, IN
G. MADEY, University of Notre Dame, Notre Dame, IN

ABSTRACT

With so many toolkits available, the choice of which one is best suited for your project can be overwhelming. Moreover, different communities of users prefer different aspects of a toolkit. This paper is a survey of the toolkits that are available today and how they compare to each other from a multi-stakeholder perspective. Our goal is to provide users the ability to better choose a suitable toolkit based on the features abstracted from various documentation and the first hand experiences of a broad range of communities of users and compiled into an easy to use compendium. In addition, we expand the Agent Based Modeling body of knowledge to include information about a breadth of characteristically and historically diverse platforms.

Keywords: Agent Based Modeling and Simulation, Agent Based Modeling Toolkits, Multi-stakeholder Community

INTRODUCTION

Agent Based Modeling (ABM) toolkits are as diverse as the community of people who use them. With so many toolkits available, the choice of which one is best suited to a project can be overwhelming. Current toolkit surveys are helpful but are limited to four or five mainstay and characteristically or historically similar platforms (Railsback et al 2006; Tobias et al 2004; Castle et al 2006). Moreover, recent surveys are presented from the point of view and for the intended audience of one or two communities of interest (Railsback et al 2006; Tobias et al 2004). However, different groups of users prefer different and sometimes conflicting aspects of a toolkit. For example, social scientists, who may have little or no programming experience are concerned more with ease of use, the degree of programming skills required, and the inclusion of intuitive interfaces to manage simulations. Many, in general, are not concerned about whether the software is open source or restricted open source. To computer scientists, however, the type of license that comes with the toolkit is a big consideration; they want the ability to "get behind the scenes" of a toolkit and to have the programming flexibility to modify or extend the software with third party applications if necessary. They also generally prefer saving execution time by programming simulations themselves rather than using built-in interfaces. Teachers of ABM, on the other hand, want packages that are easy to learn, that offer pedagogical insights, and that provide the student with the ability to transition to more difficult and comprehensive toolkits in the future.

In this paper, we survey the current state of the art in ABM toolkits, and we compare them to each other from a multi-stakeholder perspective. Our goal is to provide users the ability

• *Corresponding author address*: Cynthia Nikolai, Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556; email: cnikolai@nd.edu.

to better choose a suitable toolkit based on the features abstracted from various documentation and the first hand experiences of a broad range of communities of users and compiled into an easy to use compendium. We use a combination of both scaled and quantifiable evaluations to create a taxonomy of toolkits for each characteristic of interest. This is followed by a text explanation of each feature, including how and why a feature is ordered in each paradigm. Some of the characteristics we evaluate include supported platforms, programming language and degree of programming skills required to create a model/simulation, major domains for which a toolkit may be used, type of license the toolkit includes, ease of use and completeness/robustness of a toolkit, the maximum number of agents supported, and the ability to extend a toolkit with third party software. We also capture a history of the toolkits, explaining the influences that united to produce them and how different parallel threads of the ABM community emerged over time.

This paper is structured as follows. In section I, we provide a short introduction to ABM. This is followed in section II by a description of our methodology; In section III, we include a preliminary compendium of taxonomies.

BACKGROUND

Agent based modeling is a framework for modeling a simulation based on creating a set of autonomous objects, called agents or entities. An agent is “an abstract or physical autonomous entity which performs a given task using information gleaned from its environment to act in a suitable manner so as to complete the task successfully. The agent should be able to adapt itself based on changes occurring in its environment, so that a change in circumstances will still yield the intended result.”[1] The goal of agent based modeling is to have a many simple entities, by which we can discover the emergent behavior of a system. In this paper, we evaluate the toolkits on the market today which use this framework for modeling and simulating agents.

METHODOLOGY

Goals

We began this research by studying survey design techniques (Arsham 2002; Creative Research Systems 2006; US GAO 1993; US GAO 1992; Walonick, 1997). First we outlined the specific goals we hope to achieve through this survey so that we could identify our measurement variables, and structure our question designs appropriately. The specific goals of this research are:

1. to help multi-stakeholder users choose an ABM toolkit based on the characteristics available
2. to compile a broader list than is available of toolkit characteristics into one easy to use reference for users
3. to find out why type of classes of users are using each ABM toolkit
4. to ascertain what characteristics different communities of users when choosing an ABM toolkit

[1] there are many different and equally valid definitions of ABM, but for the purposes of this research, we will choose this one. This one is an informal definition by G.W. Lecky-Thompson quoted in (Hermans 97).

Population Sample

Next we selected a sample population to achieve each of the goals. In order to achieve goals 1 and 2, we have decided to contract a developer from each toolkit under consideration. We have developed a specific survey for this group and this set of goals. (See appendix A for a list of toolkits under consideration) In order to achieve goals 3 and 4, we have decided to sample the users of each toolkit. We will sample about 5-10 users from each toolkit. We will achieve this by contacting the user's groups and/or the mailing list of the toolkits. We have developed a specific survey for this group and this set of goals.

Data Analysis

In order to analyze the data, we will use three main analysis techniques: measures of central tendency; measures of distribution, measures of association, and measures of causation. In order to facilitate this, we have designed the survey questions to facilitate these types of data analysis. For example, we have structured the survey with as few open ended questions as possible. For most values, we have a list of qualified answers. We also have incorporated many questions from an open ended format to a qualified, anchored scale.

Sources of Error and Countermeasures

The major sources of error in this research are various forms of biases that may be inadvertently or intentionally introduced. In order to reduce inadvertent biases introduced in the structure of the questions themselves or from the respondents, we researched and applied proven survey design techniques that address exactly these issues (Arsham 2002; Creative Research Systems 2006; US GAO 1993; US GAO 1992; Walonick, 1997). For example, we familiarized ourselves with the population by reviewing the literature on the subject and talking with subject matter experts. We specifically selected the sample populations and determined sample sizes to eliminate biases and errors and be able to generalize to the population at large. We developed the goals and identified corresponding measurement variables and then designed our questions to evaluate the measurement variables and achieve our goals. We determined appropriate sample populations and sample sizes. We also structured the questions such that we addressed the limitations of each type of question to reduced its inherent biases. Some of the techniques we used include writing clear questions, using syntax and linguistics to facilitate question understanding and respondent recall, developing unscaled response lists, developing questions to minimize question bias and memory error, tailoring questions to minimize respondent bias, tailoring questions to minimizing measurement error, using odd numbered Likert scales to allow for neutrality in decisions by the respondent, quantifying all scaled values, anchoring our scaled lists, allowing escape choices for the respondent, including room for additional comments, incorporating a pledge of anonymity, avoiding inappropriate questions and questions that do not contribute to the goals, including an incentive, organizing the line of inquiry to maintain user interest and avoid bias, categorizing topics by heading, using lists to avoid biases in memory recall, qualifying the alternatives equally to avoid question biases, avoiding "yes" biases, asking more specify questions at beginning and more broad question at the end of the survey, initiated plans for follow-up for respondents and non-respondents, designing the questionnaire layout and graphics to facilitate user satisfaction and interest in the survey, defining words that could be construed in a non-standardized way or in a different context to facilitate standardization interpretation of the questions, and finally pretesting the questionnaire to help validate our survey.

In order to eliminate potential intentional bias, we only ask the developers to evaluation their own toolkit; we let the user's evaluate comparable toolkits on the market. However, because the developers can be biased toward their own software, in order to validate answers to these questions, we ask the users directly to evaluate important characteristics of the toolkit and comparable toolkits on the market. In order to eliminate skewed sample data toward one or two major platforms, we have chosen samples from each toolkit user's group.

Data Validation

We will use current literature and expert opinion to validate developer and user responses; we also will use information collected through open source channels and expert opinion to validate the responses.

Potential problems noted

The major potential problems we may encounter are low response rate. In order to reduce non-responses, we plan to implement proven techniques to make the surveys easy, simple, understandable, standardized, and pleasurable to the user. We also have included an incentive: at the end of the collection period, we will have a drawing to give away three $20 gift certificates to developers who respond to the survey. In addition, we will have a drawing for three $20 gift certificates for users who complete the survey. We will attempt follow-up contacts with the non-respondents per the survey design guidance. If we still do not have a response, we will try to fill in the missing data as best as we can. A similar problem we may encounter is if people do not answer questions or if answers to questions are unclear to the authors. In order to account for missing data and information for which we need respondent clarification, we will attempt to contact the individual, if follow-up contact has been authorized by the respondent.

Limitations

Some of the limitations of this research are that we have a relatively small sample size, so the extrapolation may be less accurate for the entire population. There are general limitations of survey data and of this form of questionnaire, which include missing data, non-responses, question biases, memory biases, respondent biases, unstandardized interpretation of the questions. Another limitation is that we assume that all toolkits written in different languages by the same developers or development groups have congruent capabilities. For example, we assume that anything one can do with the objective C swarm toolkit can also be implemented with similar results in Java based swarm. In reality, there are subtle differences and nuances between the two that may be important to users of the toolkits. Fourth, this survey is more of a broad study of the ABM field rather than an in-depth study of one or two platforms. As such, we do not go in depth for any one toolkit. Finally, the current list of important characteristics that are being evaluated for each toolkit are based on current literature, which is has been geared mostly toward the social science community.

Preliminary Results

Then we gathered and assembling as much information as possible on various toolkits from open sources and documentation. Taking some the questions that are important based on current literature (Railsback et al 2006; Tobias et al 2004; Castle et al 2006), we use the following list of characteristics that commonly are traded off in choosing a toolkit. Some of these characteristics include platforms supported, programming language required, degree of programming skills required, ease of use, maximum number of agents supported, license employed, ABM history/roots. Note that this list is not complete and may change as the responses from our questionnaires direct. They simply are a starting point and a preliminary point of validation for this research. (See appendices B-E for beginning taxonomies for several characteristics)

Completion of the taxonomies and more in-depth explanations will follow when as we obtain and validate results from our surveys. Note, in the final results, we also will include a full representation of features for each toolkit in an easy to use matrix format that allows for quick and comprehensive comparison of particular characteristics across different toolkits, or an examination at all characteristics across one specific toolkit.

FUTURE WORK

Currently, we are in the pretest phase of the survey design. In the next step, we will deploy our surveys, collect the responses, and start analyzing and interpreting the results.

CONCLUSION

Different communities choose a toolkit based on various sometimes conflicting and contradictory aspects as other communities. In this work, we explore what aspects different communities value in choosing a toolkit. We also survey the current capabilities of the toolkits that are available today to help users choose an appropriate toolkit for their purposes. We explore a breadth of the current state of the art, and we organize the information into a compendium of taxonomies for easy access and comparison of features. When we complete the work, we will include a tabular formulation of the results as well.

ACKNOWLEDGEMENTS

We are very grateful to Dr. Greg Madey, Tim Wright, Scott Christley, Ryan Kennedy, Alec Pawling, Matt Van Antwerp, and Ying Zhoufor their contributions to this paper.

REFERENCES

Arsham, Hossein, 2002, “Questionnaire Design and Survey Sampling,” available at http://www.sysurvey.com/tips/arsham.htm.

Castle, Christian J. E., and Andrew T. Crooks. 2006, “Principles and Concepts of Agent-Based Modelling for Developing Geospatial Simulations,” Working paper 110, Centre for Advanced Spatial Analysis, University College London.

Creative Research Systems, 2006, “Survey Design,” available at http://www.surveysystem.com/sdesign.htm.

Hermans, Björn, 1997, “Intelligent Software Agents on the Internet: An Inventory of Currently Offered Functionality in the Information Society and a Prediction of (Near) Future Developments,” *First Monday*, Vol II No 3, available at http://www.firstmonday.dk/issues/issue2_3/ch_123.

Railsback, S. F., S. L. Lytinen, and S. K. Jackson. 2006, “Agent-Based Simulation Platforms: Review And Development Recommendations,” *Simulation* 82: 609-623.

Tobias, R., and C. Hofmann. 2004, “Evaluation of free Java-libraries for social-scientific agent based simulation,” *Journal of Artificial Societies and Social Simulation* 7:1.

U.S. General Accounting Office, 1993, “Developing and Using Questionnaires,” GPO Access; available at http://archive.gao.gov/t2pbat4/150366.pdf.

U.S General Accounting Office, 1992, “Quantitative Data Analysis: An Introduction,” GPO Access; available at http://archive.gao.gov/t2pbat6/146957.pdf.

Walonick, David, 1997, “Survival Statistics,” StatPac, Inc, available at http://www.statpac.com/statistics-book/index.htm.

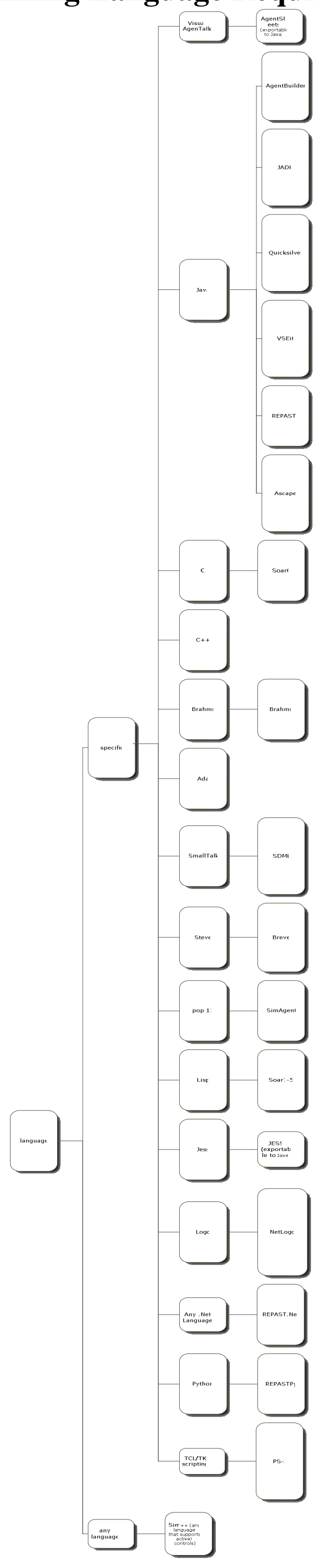
Appendix A

Toolkits Under Consideration

ABLE
Act-R
Ada
Agent Development Kit
AgentBuilder
AgentKit
AgentSheets
AnyLogic
Ascape
Brahms
Breve
Cormas
Cougaar
DeX
DOMAR
ECHO
ECJ
iGen
ISAAC
JADE
JAS
JASA
JCA-Sim
jES
JESS
LSD
Madkit
MAGSY
MAML
Mason
MAS-SOC
Matlab
MIMOSE
Moduleco
NetLogo
OBEUS
openStarLogo
oRIS
Ps-I
Quicksilver
Repast
SDML
Sim++
SimAgent
SimBioSys
SimPack
SME
SOAR
StarLogo
StarLogoT
Sugarscape
Swarm
TeamBots
Vensim
VSEit
ZEUS

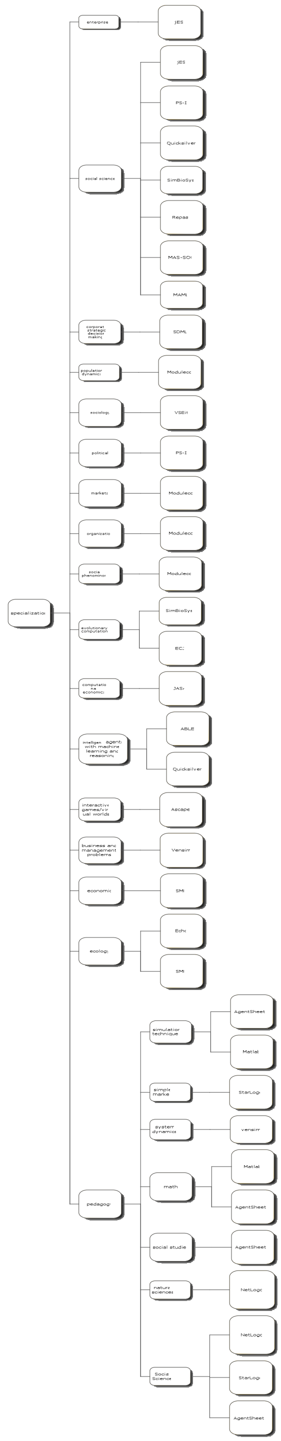
Appendix B

Programming Language Required



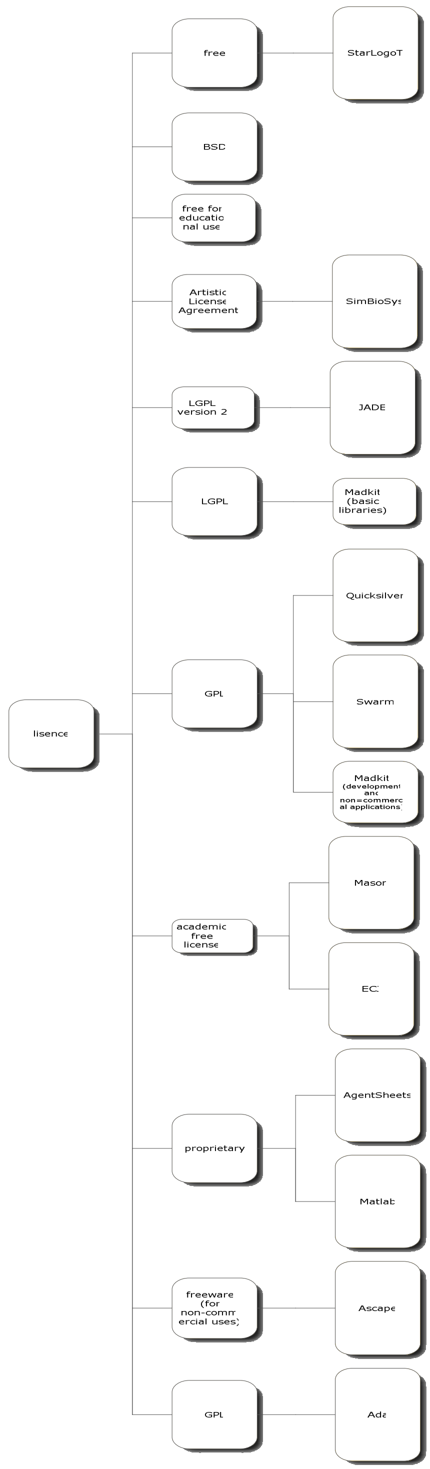
Appendix C

Domain Designed For



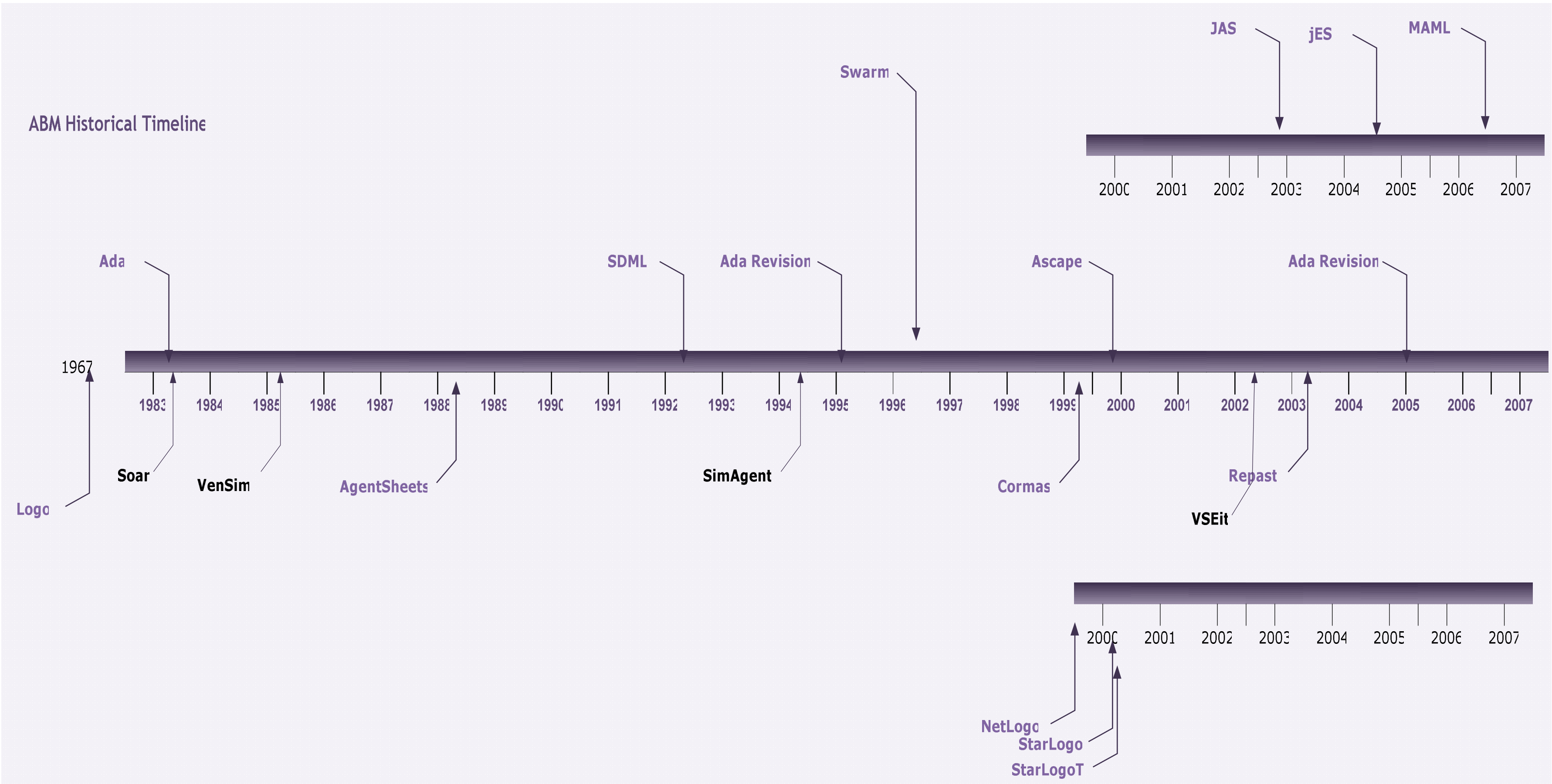
Appendix D

License Employed



Appendix E

History/Roots of ABM Toolkits



OUR SUMMER WITH REPAST: FORGING A MODELING AND SIMULATION FOUNDATION

T.A. BERGEN-HILL, *MITRE Corporation, McLean, VA
M.T. MCMAHON, MITRE Corporation, McLean, VA
B.F. TIVNAN, MITRE Corporation, McLean, VA

ABSTRACT

The success of a large-scale agent-based model and simulation (ABMS) depends on finding the right development tools for each phase of its development lifecycle. To that end, the authors spent three months evaluating free agent-based modeling software, with the goal of acquiring a tool set that would start with rapid prototyping and would progress to parallel and distributed runs on a cluster-type computing environment. The products we reviewed during this study include NetLogo, Repast J, Repast Simphony, and ProActive. In this paper, we describe the efforts required to create and maintain various models within each of these products, along with their respective strengths and limitations. We present our results as a progression through a candidate tool set that will serve as our foundation for upcoming ABMS efforts.

Keywords: Repast, NetLogo, ProActive, cluster

INTRODUCTION

During the development lifecycle of an agent-based model and simulation (ABMS), different needs arise at each phase. Initially, a "quick-and-dirty" prototype is required to verify the problem domain with the customer and to preview the simulation's look-and-feel. After feedback is received, what follows is the formulation of the model and a subsequent semi-robust implementation. Once it is working sufficiently, providing reasonable results for a small input set, the model is pushed to its computational limits and beyond, requiring the eventual use of large-scale computational hardware. The success of the simulation depends on finding the right development tools for each phase.

To that end, the authors spent three months evaluating free ABMS software, noting the steps required to utilize each tool and to transition the simulation to another tool. The goal of this research is to acquire a tool set that would start with rapid prototyping and would progress to a system capable of parallel and distributed runs on a cluster-type computing environment.

Our requirements include the capability of deploying a prototype model developed on a Windows workstation to a distributed-memory Linux cluster. In addition to investigating the issues with cross-platform development (e.g., Windows workstation to Linux workstation), we also discuss the issues with transitioning to parallel models (e.g., Linux workstation to Linux

* Corresponding author address: Tobin A. Bergen-Hill, The MITRE Corporation., 7515 Colshire Drive, McLean, VA 22102-7539; e-mail: tbergenhill@mitre.org.

cluster). Using the parallelism terminology found in North and Macal (2007), the initial goal for the cluster capability is "coarse-grained" parallelism, i.e. a parameter sweep for a trade study or objective function analysis for parameter optimization (Fujimoto 2000, Lin 1994). The eventual goal is a "fine-grained" parallel simulation, i.e., the use of separate processes within the model itself.

In this paper, we describe the efforts required to create and maintain various models within each of these toolkits, along with their respective strengths and limitations. We note which products have the ability to co-exist in a simulation. Where there are conflicts or issues, we investigate whether the problems are intrinsic to the tool itself or whether they stem from the ongoing development of the tool. We present our results as a progression through a candidate tool set that will serve as our foundation for upcoming ABMS efforts, and we discuss the decisions made for selecting the tools. Our hope is that others within the modeling and simulation community who are at the initial stages of a project and who have considered any of these tools can benefit from our experience and recommendations, and that they can apply our template to make their project a success.

Related Work

One of the more similar efforts we've found is the HLA_GRID_Repast system from a consortium led by the University of Birmingham (Zhang et al. 2005). As a system for executing large-scale distributed ABMS over the Grid, it is definitely relevant. However, we did not want to be bound by a single protocol for network communication nor by a specific target system. We're seeking a flexible framework that allows us to choose the network protocol for the situation and to run on a variety of operating systems and hardware configurations.

Another similar effort is the MACE3J testbed. According to Gasser and Kakugawa (2002), it is designed for multiple granularities of parallel execution and can run on a variety of work stations. We recognize that MACE3J is a likely candidate for further evaluation; however, our time constraints for a related, development project restricted the number of toolkits that would receive detailed attention.

RESEARCH

Seeking a progression of toolkits that would lead us from prototype to a massively parallel ABMS, we reviewed multiple existing tools across different development environments and systems. We started with the list of free toolkits cited in the ABMS text by North and Macal (2007), and narrowed it down to a handful of candidates. Before presenting our review notes, we'll briefly discuss our approach to simulation parallelization.

Parallelization Approach

When refactoring a simulation to a distributed architecture, there are numerous tradeoffs to consider. In a shared-memory parallel (SMP) environment, threads can utilize shared memory to exchange data. But in a distributed-memory implementation, agents don't have access to shared memory and must have an explicit communication facility, which yields model design overhead as well as runtime communications overhead. Also, the spectrum of parallelization granularity ranges from trade study/parameter sweeps (e.g., batch mode) to simulation-level granularity (e.g., HLA integration) to fine-grained distribution of a single agent. The most suitable level of granularity again is a factor of the desired quantity of network communications verses computational power.

In our investigation, we have kept in mind general parallel design approaches (Buyya 1999): partitioning of a model, inter-process communications, mapping to the hardware, and agglomeration of results. In the modeling and simulation domain, this general approach distills to determining, for a given model, the best approach to partitioning that model in a manner which diminishes run-time overhead (communications), while still enabling distribution across processors in a cluster environment. We describe our partitioning scheme in the Findings section of this paper.

Product Review

The products we reviewed during this study include: NetLogo 4.0 beta, Repast J v3, Repast Simphony Alpha 2, and ProActive v3.2.1. A summary of our notes can be found in Table 1 below. The rest of this section expands on the strengths and limitations of these toolkits. To help contrast the simulation architecture of each toolkit, we provide a comparison of different demonstration models in Appendix A: Demo Model Comparison.

Table 1 Toolkit Comparison Chart.

Toolkit	Strengths	Limitations
NetLogo	• Easy-to-use language • Java interface • 2D & 3D display	• Not meant for large scale simulations • No multithread support • No distributed simulation support • No batch run support
Repast J	• Handles medium-scale numbers of agents • Contains several useful simulation classes • 2D display • Supports multithreaded events • Supports distributed batch runs	• Designed for single host execution per run, not for distributed simulation • No 3D display • Made obsolete by Repast Simphony
Repast S	• Handles medium-scale numbers of agents • Contains several useful simulation classes • 2D & 3D display	• Still in development • No distributed simulation support • Bound to only one development environment (Eclipse) • Currently cannot function with

	• Supports multithreaded events	ProActive in a distributed fashion
ProActive	• Supports distributed processes (and consequently large-scale simulations) • Minimally intrusive API • Configure distribution through XML file	• No simulation architecture • No visualization capabilities • Clutters the command line • Makes debugging difficult

The selection of these toolkits over others warrants explanation. One toolkit similar in purpose and structure to Repast is Swarm (Minar et al. 1996); however, we chose to review Repast because of its availability in Java and because of our familiarity with Repast. We also selected NetLogo to review instead of StarLogo (Colella et al. 2001) because of our existing proficiency with NetLogo and because of NetLogo's ability to interface with native Java 1.5 applications.

NetLogo

An ideal rapid-prototyping toolkit for ABMS is NetLogo (Wilensky 1999). We started with version 3.1.4 and migrated to version 4.0 beta 5 halfway through the summer. NetLogo has many strong points. Its script language is easy to use, which is no surprise given that it was originally developed for educational purposes. It features a "Java extensions" interface that links the NetLogo script with external Java routines. The display, which is essential to understanding and debugging the simulation, can switch between 2D and 3D, although the simulation is based on a 2D grid. And there are over 150 models that come with the distribution, to serve as examples.

As a prototype tool, NetLogo is not meant for large-scale simulations. There is no support for multi-threading natively in NetLogo, only through farming the work to threaded Java code. Similarly, NetLogo is not set up to support distributed simulations; its HubNet feature is a form of participatory simulation meant for human interaction, not high performance computing.

Repast J

Repast, or the REcursive Porous Agent Simulation Toolkit, is a medium-scale ABMS toolkit that has various forms (North et al. 2006); for our study, we focused on Repast for Java (Repast J) and Repast Simphony (Repast S). Repast J provides an execution environment with an optional GUI for controlling the execution and monitoring of the simulation. It features several useful simulation classes including a discrete-event scheduler, representation of the model space, batch-mode utility classes, and a built-in 2D visualization capability. For agent representations, Repast J has no explicit agent class, but it does offer adaptive behavioral tools.

For our purposes, we used a subset of the Repast J classes:

- *SimModelImpl*: This is a partial implementation of the SimModel interface, through which Repast drives the simulation. Each Repast simulation must provide a SimModel subclass.

- *Schedule*: This serves as a discrete-event scheduler. The simulation can specify the order in which events (or "actions") are executed.
- *BasicAction*: All simulation events inherit this interface class in order to be stored and executed by the scheduler.
- *SimInit*: This class is responsible for loading and executing the model, with optional GUI controls or batch mode operation.
- *DistributedSimInit*: This experimental stand-in for SimInit uses ProActive to distribute batch runs of the model across a given set of hosts.
- *Random*: This class encapsulates CERN's Colt random number utilities. An instance of this is placed in SimModelImpl, and the seed can be either generated from a timestamp or explicitly specified for reproducibility.

Both versions of Repast feature a scheduler that supports concurrent actions, simply by specifying a non-zero duration argument. But there are no constructs to support a distributed simulation, only distributed batch runs. The Parameter Wizard makes it easy to specify parameter sweeps for batch runs. It produces a parameter file, which can be passed in on the command line arguments to the SimInit class. SimInit will run the model in batch – but in serial fashion, not simultaneously.

To perform simultaneous distributed batch runs, Repast-J provides an undocumented mechanism that utilizes ProActive. DistributedSimInit establishes the virtual nodes from the ProActive descriptor and creates RemoteBatchController instances on each virtual node. The HomeController class tells the group of RemoteBatchControllers which model to instantiate and run; it also hands them the parameters to use for that run. We've noted how to run simultaneous batch runs in Appendix B: Simultaneous Multi-Host Batch Runs In Repast J.

Repast S

Currently in the alpha stage of development, Repast Simphony is a complete redesign of the Repast toolkit. As described by North and Macal (2007), "the Repast S runtime is designed to include advanced features for agent storage, display and behavioral activation, as well as new facilities for data analysis and presentation."

Repast S makes use of configuration files (model.score and scenario.xml) to specify the roles of the classes in a simulation model. As stated in its preliminary documentation, the central class is the Context, which organizes the agents, denotes the relationships of its members (through "projections") and holds numerical data in data layers. There is also the ContextBuilder, which does the work of storing information in the Context. The configuration files identifies the ContextBuilder, the agent classes, and which Context to use.

Of the toolkits reviewed, we encountered the most platform-related issues when working with Repast Simphony and Eclipse, the development environment to which Simphony is tied. The case of the file names, ISO encoding of data files, and use of Java annotations (e.g. @override) created compile and run-time issues on Linux systems. Also, Simphony binds us to a specific Java version; on Fedora Core 5, SELinux inhibits the use of Sun's Java. Most importantly, though, is the different packaging used for the platform-independent distribution of Repast Simphony versus Windows, which uses the intrinsic Eclipse packaging system. Luckily, we found workarounds to these issues to allow us to continue research.

When integrating the ProActive toolkit into our Repast Simphony simulation, we experienced interference from the Repast Simphony runtime environment – something that we didn't encounter with Repast J. First, in order to allow third party JAR files to be used, we had to place them within the directory containing the repast.simphony.core package, and we had to explicitly add the names of the JAR files to that package's manifest and build properties. (With Repast J, it was sufficient to include the JARs in the ProActive descriptor's class path variable.) Second, when we configured the simulation to spawn separate processes with their own JVMs from within the simulation, any serialized application-level class that is passed to these processes results in an exception. The spawned Repast S process cannot resolve the temporary ProActive "stub" class. (The Repast J spawned processes had no problems with class resolution.) Finally, we could not locate the means for activating code upon simulation end. This is required in order to properly clean up the ProActive processes. While the ModelInitializer interface provides a "hook" that is activated upon startup, no such "hook" exists for shutdown – even overriding Object.finalize() did not work.

We also attempted to investigate the batch run capability of Repast Simphony on a Linux environment. But, following the instructions provided in the Alpha release, we were unsuccessful. Again, we anticipate that the next release will hopefully address these issues.

ProActive

As described by Baude et al. (2006), ProActive is "a Java-based middleware (programming model and environment) for object and component oriented parallel, mobile and distributed computing." It provides a minimally intrusive interface to managing distributed object instances. Consequently, it does not have any built-in simulation architecture or visualization capabilities, which makes it an ideal supplementary package to Repast.

To distribute an object, the application uses a ProActive method to turn it into an "active object" which can then be placed on a specified host, or "virtual node". The application interacts with the object through the normal method calls, as long as the arguments to the methods can be serialized. Return values from objects cannot be immediately accessed, since they are only proxies to the value. The application can check whether the proxy's value has arrived prior to accessing it.

The main configuration file for ProActive applications is called the Descriptor (or PAD), because it describes which machines are available and how many JVMs to run on each. It also describes the class path for the spawned processes. The file is not quite platform independent; on Windows machines, none of the specified paths should contain any spaces.

To support load balancing, ProActive provides the means to migrate active objects between virtual nodes. There is a fair amount of overhead to each active object, however. For large-scale simulations that need to migrate agents, an alternative is to make the agent serializable instead of an active object. Then the agent can be passed between virtual nodes using regular method invocations.

Running a ProActive-enabled application requires the use of extensive scripts that configure environment variables, add many JAR files to the class path, and JVM command-line arguments. Consequently, debugging a ProActive application within an IDE (such as NetBeans or Eclipse) is difficult because all these run-time parameters must be reproduced. We have yet to be successful in using a debugger with ProActive.

Both distributions of Repast come with ProActive file. However, this version is not current. In the case of Repast-J, which includes ProActive classes within its JAR file, this causes problems on Linux with NetBeans: when specifying the JAR files in the Libraries section of NetBeans, ProActive.jar must appear above repast.jar in order to use the more recent version.

FINDINGS

From the set of toolkits reviewed, we have found two potential paths of progression for an ABMS, as illustrated in Figure 1. The "current" path reflects what is possible with the current state of the toolkits; it is a complete progression from prototype to a large-scale distributed simulation. The "future" path is based on promised capabilities from the Repast Simphony development team.

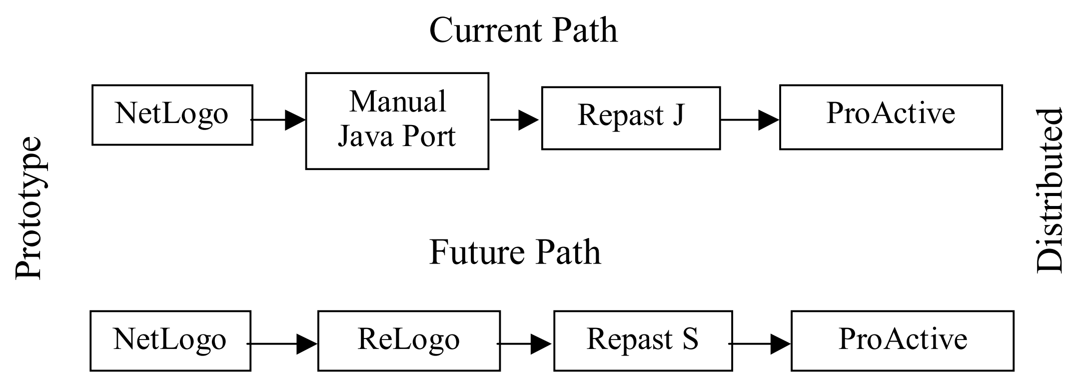


Figure 1 ABMS Development Progression Paths.

Current Path

Using the toolkits in their current state, we have had success with the following progression of development:

- Stage I: Prototype in NetLogo
- Stage II: Pure Java
- Stage III: Repast J simulation
- Stage IV: Distribute simulation via ProActive

The next few sections discuss the transition between these states.

Stage I to Stage II

Going from NetLogo script to Java code is currently a manual process. Initially, this process can start by using Java extensions from within the NetLogo script for key components of the simulation's business logic. But to go completely to Java, the NetLogo representations of turtles and patches require a Java counterpart, along with the visualization and event loop. The turtle classes will require movement methods, and both the turtles and patches will need NetLogo-like query operations. It turns out that the turtle classes become the simulation's agent classes.

It is debatable whether it is necessary to put the simulation into an intermediate pure Java form, or whether to go directly to using the Repast J classes. It is easier to debug the simulation's logic when in pure Java. However, the port will produce temporary classes that end up being replaced by their Repast J counterparts (e.g. schedule, visuals, and terrain). Working directly in Repast will prevent redundant work.

Once the port is complete, the NetLogo prototype can still be developed, as long as the changes are tracked and reproduced on the Java side. But at some point, modifications must stop flowing down.

Stage II to Stage III

Adapting a Java simulation to the Repast-J framework is relatively straightforward because the requirements are minimal:

- Provide a subclass of SimModelImpl with the begin() and setup() methods overridden
- Schedule events using the Schedule and BasicAction classes
- Fire a stop event when the simulation has finished

Repast J does not have an agent interface class, so there are no restrictions on the agent representation.

With regards to visualization, the simulation could use the 2-D visuals provided by Repast J. In order for agents to be rendered, the agent class must implement the Drawable interface. Also, the SimModelImpl subclass must create some representation of the model space (e.g. Object2DTorus), a DisplaySurface and an Object2DDisplay. Alternatively, the simulation could use a third-party visualization toolkit, such as Java3D. We chose to use Java3D because there are certain advantages to having the viewpoint at the agent level (e.g. obscuration of field of view affecting agent logic).

Another reason to use a separate visual toolkit is to prepare the simulation for a distributed, batch run. The results can be logged to data files and replayed through the visuals. If there's a need to see the visuals during the simulation run, then a communication link can be established between the Repast process(es) and the visual process. But this will introduce a performance penalty.

Stage III to Stage IV

Going to a distributed framework is the most difficult transition to make because of the impact on the simulation's architecture. The simulation engineer must determine how to partition the execution across different machines while keeping the amount of network traffic to a minimum. The initial impulse is to make the agents themselves become separate processes; but if the agents require constant communication with its neighbors, the overhead of the supporting network communications will outweigh any benefit to be had by distributing the simulation.

Instead, we recommend that the engineer work at a larger level, deciding how to carve out large pieces of the simulation that can operate nearly independently, which we'll call a "domain." Each domain will be responsible for managing all agents within an area – in the same process. To address the boundary condition problem - e.g., when an agent travels between domains – we can learn from parallel battlefield simulations and define the domains geographically (Nicol 1988). Each domain should also write its results to a uniquely named log file. When running on a cluster environment, we recommend writing to a file on the local file system. Otherwise, the cost of writing to a NFS-mounted drive negates the performance benefits of distributed processing.

Since the SimModelImpl subclass cannot be serialized and passed to the domains, another class is required to act as a coordination object. Domains would send any needed run-time statistics to the coordination object (in addition to logging data locally); however, this should be kept to a minimum because it is a major performance bottleneck. Domains can also notify the coordination object when they've finished, which can signal Repast J that the run has ended.

When the simulation begins, the following should occur within the SimModelImpl subclass's begin() method:

- Read in the PAD file
- Activate the virtual nodes
- Create the domains as a ProActive group, using the list of available virtual nodes
- Create the coordination object as an active object and pass it to the domains
- Schedule an action whose execute() method performs the following:
    - Check with the coordination object to see whether the domains have finished, and if so, fire a stop event
    - Display statistics (if enabled)
    - Update the domains
    - Schedule another action

Once the simulation ends - either because the domains indicated that they have finished, or because the user presses the stop button - a stop event will be triggered. The SimModelImpl subclass needs to handle this event by cleaning up the ProActive active objects and killing all the JVMs on the virtual nodes.

After the simulation has been outfitted with ProActive, it must be tested to make sure the functionality has been preserved. We recommend using the following progression of PAD files:

- One virtual node that is using the same JVM as the Repast application
- Multiple virtual nodes that are using different JVMs but on the same host as the Repast application
- Multiple virtual nodes, each on a different host

The final step is adding the ability to perform simultaneous batch runs using Repast J and ProActive. For details on how to accomplish this, see Appendix B: Simultaneous Multi-Host Batch Runs In Repast J.

Future Path

Based on the limited information we've heard about the next release of Repast Simphony, we anticipate that the following progression will be possible:

- Stage I: Prototype in NetLogo
- Stage II: ReLogo
- Stage III: Repast Simphony simulation
- Stage IV: Distribute simulation via ProActive

The next few sections explain the transitions between these states.

Stage I through Stage III

Our speculation is that Repast Simphony will be packaged with a utility dubbed "ReLogo", which will parse NetLogo script and turn it into Repast Simphony Java code. This will make obsolete the manual process of converting NetLogo to Java from the "current" path. The anticipated release of ReLogo is November 2007.

Stage III to Stage IV

As in the "current" path, the process of outfitting a Repast simulation with ProActive constructs is still performed manually. The intent was to demonstrate at summer's end the ability to run a Repast Simphony simulation in a distributed fashion. However, we were unable to successfully invoke methods on active objects that reside in a separate JVM (either on the local host or a remote host).

The test application we used is a modified version of ProActive's n-body simulation. We added the minimum constructs necessary to convert it to a Repast S simulation. For detailed steps on this procedure, see Appendix C: Repast Simphony – ProActive Integration Notes.

NEXT STEP

While we are content with using the "current" path as a template for our ABMS work, we look forward to the streamlined process featured in the "future" path. We hope to see the ReLogo mechanism in the next release of Repast Simphony, and we hope to solve the issues preventing ProActive from reaching its full potential in Repast S.

We wish to explore other ideas to reduce the work involved. Since the use of ProActive centers on properly configured descriptor files, we would like to construct a tool to compose and generate PAD files. Another improvement would be to write a better parameter sweep mechanism in Repast that would vary parameters while running on different groups of hosts; this would require removing the group name as a parameter and providing it through some other means of input. This custom parameter sweep could also support varying multiple parameters per run, going beyond what is provided in Repast. We are also seeking ways to incorporate ProActive constructs in a semi-automatic fashion, perhaps as custom hooks within the ReLogo conversion process.

REFERENCES

Baude F., L. Baduel, D. Caromel, A. Contes, F. Huet, M. Morel and R. Quilici, 2006, "Programming, Composing, Deploying for the Grid", *GRID COMPUTING: Software Environments and Tools*, pp. 205-229, London, England: Springer-Verlag.

Buyya, R., 1999, High Performance Cluster Computing, Volume 1, Architectures and Systems, NJ: Prentice Hall.

Colella, V. S., E. Klopfer, M. Resnick, 2001, *Adventures in Modeling: Exploring Complex Dynamic Systems with StarLogo*, New York, NY: Teachers College Press.

Fujimoto, R.M., 2000, *Parallel and Distributed Simulation Systems,* New York, NY: Wiley And Sons

Gasser, L. and K. Kakugawa, 2002, MACE3J: Fast Flexible Distributed Simulation of Large, Large-Grain Multi-Agent Systems, *The First International Joint Conference on Autonomous Agent and Multiagent Systems, Vol. I*, pp. 745-752, New York, NY: ACM Press.

Lin, Y., 1994, Parallel independent replicated simulation on a network of workstations, *Proceedings of the Eighth Workshop on Parallel and Distributed Simulation*, pp. 73-80, New York, NY: ACM Press.

Minar, N., R. Burkhart, C. Langton, and M. Askenazi, 1996, The Swarm Simulation System: A Toolkit for Building Multi-Agent Simulations, Working Paper 96-06-042, Santa Fe, NM: Santa Fe Institute.

North, M. and C. Macal, 2007, *Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation*, New York, NY: Oxford University Press.

North, M.J., N.T. Collier, and J.R. Vos, 2006, "Experiences Creating Three Implementations of the Repast Agent Modeling Toolkit," *ACM Transactions on Modeling and Computer Simulation*, Vol. 16, Issue 1, pp. 1-25, New York, NY: ACM Press.

Nicol, D.M., 1988, Mapping a battlefield simulation onto message-passing parallel architectures, *Distributed Simulation*, pp. 141-146, San Diego, CA: Society for Computer Simualtion International.

Wilensky, U., 1999, *NetLogo*, Evanston, IL: Center for Connected Learning and Computer-Based Modeling, Northwestern University.

Wilensky, U., 2002, *NetLogo Mousetraps model*, http://ccl.northwestern.edu/netlogo/models/Mousetraps, Evanston, IL: Center for Connected Learning and Computer-Based Modeling, Northwestern University.

Zhang, T., G. Theodoropoulos, R. Minson, S. Turner, W. Cai, X. Yong, and B. Logan, 2005, Grid-aware large scale distributed simulation of agent-based systems, European Simulator Interoperability Workshop (Euro-SIW) paper 05E–SIW–047, Toulouse, France.

APPENDIX A: DEMO MODEL COMPARISON

The comparison of various demonstration models distributed with the toolkits can be found in this appendix. These examples supplement the discussion of the toolkits found in the Product Review section.

NetLogo MouseTrap

One model that comes with three of the toolkits is the MouseTrap simulation, which originally came from the Swarm toolkit (Minar et al. 1996). The version for NetLogo (Wilensky 2002) is simply represented by three functions:

- setup: resets the traps (i.e. the patches) and the balls (i.e. the turtles)
- go: controls the triggering of the traps and introduction of new balls
- move: move the ball in a random direction up to a specified maximum distance

Repast J MouseTrap

The Repast-J representation of MouseTrap takes the form of three classes:

- *MouseTrapModel*: a subclass of SimModelImpl. This class manages the simulation, houses the simulation space and display, and tracks the current number of active balls. The begin() method constructs the model out of MouseTrap classes. During the run, its scheduleTrigger() method schedules a TriggerAction for each trap that fires.
- *MouseTrap*: the representation of the “agent”. The trigger() method randomly chooses which neighboring traps are triggered by the balls launched by the trap, passing them to the model’s scheduleTrigger() method.
- *TriggerAction*: a subclass of BasicAction, this holds a reference to a MouseTrap instance. The execute() method triggers the trap and updates the display.

Repast S MouseTrap

The Repast S representation of MouseTrap includes the following core classes:

- *MouseTrap*: This is the agent representation, as designated in the model.score file. Its trigger() method is similar to the Repast J version.

- *MouseTrapsCreator*: This is the ContextBuilder for the demo and is designated as such in the model.score and scenario.xml files. It creates a grid projection for the context and fills it with MouseTrap agents. It also adds a data layer to the grid projection.
- *TrapInitializer*: This ModelInitializer subclass triggers the first trap. The scenario.xml file identifies this as the model's initializing class.
- *TrapTrigger*: Implementing the IAction interface, this class is similar to the TriggerAction class in the Repast J version.

ProActive n-Body

Unfortunately, there was no MouseTrap simulation that came with ProActive. Instead, we'll use an n-body simulation to demonstrate how a ProActive ABMS is structured. Below is a description of the main classes used in the n-body simulation:

- *Domain*: This class manages each Planet in the simulation, calculating the forces exerted on it by its neighbors. It relies on distributing the resulting sum to all other Domains (and their planets). It then notifies the Displayer to update the planet's position. This is one of the "active objects" in the simulation.
- *Start*: This class reads the PAD file and activates the virtual nodes, then creates the Displayer, the Planets and their Domains. It launches the Domain processes on the virtual nodes, using the ProActive group construct. When the Domains finish the specified number of iterations, they notify this class.
- *Displayer*: Acting as a wrapper for the planetary display, this class is handed to each Domain by the Start class. Like the Domain, this is also an active object.
- *Planet*: This is the "agent" of the n-body simulation, and has several physical properties to regulate its motion.

APPENDIX B: SIMULTANEOUS MULTI-HOST BATCH RUNS IN REPAST J

This appendix discusses how to utilize Repast J and ProActive to perform simultaneous batch runs. The main difficulty is to prevent each run (which itself uses multiple hosts) from executing on remote hosts that are currently executing other runs. To accomplish this, the DistributedSimInit class can be used in combination with a special parameter sweep:

1. Make multiple virtualNode mappings in the XML file:
- "remote": these are the nodes that will have the SimModelImpl subclass
- "Workers*N*": these are the groups of nodes that will perform the work; *N* is the number of nodes in the "remote" group

2. Add a property: vnMapping; this will indicate which virtual node mapping for the model to use for its workers

3. Define a property sweep file where the inner-most sweep is the vnMapping property; have it cycle through the different WorkersN values.

4. In the model's main() method, have it instantiate a DistributedSimInit class. Call its open() method giving it the PAD filename, the model class name, and the sweep file.

Since vnMapping is a parameter that is being swept, and since the sweeping mechanism only varies one parameter per run, the same parameters will be run across all workers sets, which limits its usefulness. However, each run will use a different random seed, though – based on time.

HomeController creates a persistent DataWriter, but does not specify its file name. After all the batch runs occur, it tells DataWriter to write the current time at the end of the file; since the filename is null, the exception occurs. To work around this, create a DataRecorder in the model's begin() method. This will automatically set the filename of DataWriter, so when the batch runs finish, no exception will be thrown, and everything is cleaned up properly.

APPENDIX C: REPAST SIMPHONY – PROACTIVE INTEGRATION NOTES

The following notes reflect the steps needed to incorporate Repast Simphony and ProActive into an ABMS. They are geared towards a Windows environment.

- Repast S (and Eclipse) must be stored in a path that does not have spaces in it. For example: c:/program files/RepastS will cause problems with ProActive, but c:/projects/RepastS is fine.

- Create a new project in Eclipse according to the write-up

- In the model.score file, point the Base Path at ../../repast_workspace/PROJECT_DIR

- Add attributes only to the model.score file; nothing else is required by RepastS

- Have the main model file inherit ContextBuilder, and provide a build() method

- To the project, add external libraries for ProActive:
    - Right-click on the project and choose Build Path->Add External Archives...
    - Browse to the ProActive directory and add the following:
        - ProActive.jar
        - ic2d.jar
    - Repeat and browse to ProActive/lib and add the following:
        - xercesImpl.jar
        - bouncycastle.jar
        - javassist.jar
        - jsch.jar
        - log4j.jar
    - Repeat and browse to ProActive/lib/components and add the following:
        - fractal.jar
    - Copy these ProActive JARs to RepastS/repast.simphony/repast.simphony.core/lib
    - Add fractal.jar, xercesImpl.jar and bouncycastle.jar to the RepastS runtime component

  - In Eclipse, locate the repast.simphony.core project and open the MANIFEST.MF file
  - Of the various tabs for the manifest, choose Runtime
  - In the Classpath section, click New... then enter lib/your_lib.jar and click OK.
  - NOTE: This will add it automatically to the MANIFEST.MF file as well as the build.properties file
  - In Eclipse, open the plugin_jpf.xml file in the repast.simphony.core project
  - At the end of the <runtime> section, add a line for your JAR file, e.g.:
    - <library id="your_lib" path="lib/your_lib.jar" type="code" />

- Add a PROJECT.launch file to your project
  - Click on the pull-down menu for the Run button and choose Run...
  - Select the "SimpleHappy" run configuration and click the Duplicate button
  - Rename it to your project
  - On the Arguments tab:
    - Set the Program Arguments to "../../repast_workspace/nbody/scenario.rs" (without the quotes)
    - Set the VM Arguments to "-Djava.security.policy=file:c:/projects/proactive/scripts/proactive.java.policy" (without the quotes)
    - [Note: adding -Dlog4j.configuration=file:c:/projects/proactive/scripts/proactive-log4j actually prevents the log from appearing in the console]
  - On the Common tab, click the Browse button next to "Shared File" and choose your project
  - Click the Apply button, then Close

This will allow you to run the simulation with a single virtual node running on the local JVM. When the active objects are placed in a different JVM, the active objects are created without problems. But when an active object is passed from the main process to the other process, ProActive throws a ClassNotFound exception. The error still appears even when the application classes are explicitly added to the classpath section of the PAD file.

Emerging Toolkits

ADAPTIVE SIMULATION: A COMPOSABLE AGENT TOOLKIT FOR WAR GAME ADJUDICATION

D. DUONG, * US Office of the Secretary of Defense

ABSTRACT

The Office of the Secretary of Defense/Program Analysis and Evaluation Simulation Analysis Center is exploring war games and war game adjudicators, to improve its ability to analyze irregular warfare. Adaptive simulation is a suggested technique of composing simulations of cause and synchronizing them through connections found in correlational studies. This toolkit of composable simulations will be a hybrid of agent based models and fuzzy rules, that may be used to adjudicate war games or play them without humans in the loop (HITL). It may be used in an adaptive, quick turnaround modeling process in which models are quickly assembled upon HITL moves that can adjudicate moves based on a consensus of social theories.

Keywords: Agent Toolkit, Agent Based Simulation, War game, Constraint Satisfaction, Social Simulation, Computer Adjudication, Irregular Warfare, Fuzzy Systems

INTRODUCTION

The factor that is most significant to the type of computer analysis used on a war game is the factor of how adjudication is performed. If the all of the moves of the game are expressed in the computer, and the computer adjudicates all of the moves, then the computer may be able to play the game on par or superior to human players using game trees. Such a game can be entirely automated and analyzed by computer, and needs no human play. On the other hand, if humans adjudicate, or the computer adjudications are ignored because of "psychological" moves that are not adjudicated by computer, then the computer can learn from the moves of the humans in the loop (HITL), but statistically significant analysis is more difficult to obtain. The X-game, the extended game played by in the office of the secretary of defense to test irregular warfare strategies in the global war on terror (Dunlap 2007), is more of the later case. This paper presents a methodology for using computer technology to make the X-game more like the former kind of game, thereby bringing it into the horizon of analysis

In a war game, someone or something has to make an educated guess on how groups of persons will behave in situations. Subject matter experts (SMEs) would use heuristics on classes of persons to make generalizations. SMEs are not able to walk out the individual choices that humans would make to the degree that a computer could, but are more flexible in interpreting the meanings of events and applying heuristics in context. A better computational ability to draw

* Corresponding author address: Deborah Duong, OSD/PA&E SAC1401 Wilson Blvd., Suite 300, Arlington, VA22209. email: Debbie.duong.ctr@osd.mil

a picture of the social environment by walking through the implications of social actions would enable a full automation of analysis, without the need for HITL. Full automation would allow a complete statistical analysis of the robustness of strategies, as well as support the discovery of new strategies through data farming and data mining the results. Short of improved technology, duel analysis can be done, on both human and computer adjudicated games.

Advantages that Computer Adjudication has over SME Adjudication

Besides the obvious advantage of saving human resources, being around when SMEs may not be able to, and achieving a statistically significant number of runs, computer simulation can do a number of things that SMEs find difficult. Agent based computer simulation gives the same advantage that war gaming itself gives to analysis: the ability to walk problems through. A SME will tend to use heuristics to generalize about the attitudes of citizens, rather than walk through their individual reasons, choices, and actions. Agent based simulations allow individual behaviors to be walked through, so that the higher order effects of those behaviors are calculated. Macro effects of micro actions are the forte of agent based simulation.

It has been said that we only have a tactical understanding of irregular warfare, making the macro effects of micro actions the central mystery of irregular warfare science. Studies have shown that SMEs have great difficulty in predicting how populations will react to events. This is because social science does not have good theories of micro macro integration. Agent based simulation can help develop theories of how micro actions effect macro level social phenomena in irregular warfare scenarios. If there is no reliable way to connect the micro actions of warfare to macro social attitudes, then walking out the micro actions of irregular warfare in the X-game becomes of questionable use.

Many adjudications of the X-game could have been more accurate and less frustrating if agent based simulation was used instead of a spreadsheet adjudicator. For example, participants could not adjudicate comprehensive effects across national boundaries well, and international effects were the purpose of including very many nations in the game. X-game participants divided their assignments into individual countries without sufficient crosstalk, because complexity at that level is hard for individuals to think about, and takes computer simulation to think out. The spreadsheet adjudicator did not compute higher order effects, and as a result, the macro effects of micro actions were not computed. For example, instead of computing out the effect of a terrorist action on the value of currency, the spreadsheet adjudicator asked X-game participants to figure this out. Being asked so many questions about the macro effects of micro actions that they did not know the answer to was a frustrating waste of gamer's time.

Further, participants had difficulty walking through information operations (IO) incidents. This bothered them, because they felt that IO and the politicized environment, as well as how it contributes to intelligence gathering, is essential to irregular warfare. Rather than walk through the social dynamics of IO, they would use heuristics to adjudicate how a population would react. The heuristics the SMEs used tended not to include details about how a population's mind could be changed, which was really the point of the game, and the source of frustration for players who tried to improve their public support levels. X-game participants thought it was useless to walk through individual IO scenarios when they did not know how to aggregate them to the strategic level. Since "walking through" is the whole point of a war game, an agent based program that could calculate the strategic effects of micro IO outcomes would encourage participants to walk through the events that are the most relevant to the problem of irregular warfare.

An agent based simulation that walks through the effects of micro level actions on the macro level economy would be a low risk replacement for the linear adjudicator of the first run of the X-game. It would surpass its algorithms with only a small amount of work, and would continue to improve and grow as more social phenomena is simulated.

Advantages that SME Adjudication has over Computer Adjudication

Xgame participants felt that the computer adjudication results were so separated from the verbal game that they tended to be ignored by players, and not looked at to trigger events. They were simply scorecards, and poor ones at that, that did not adequately reflect the verbal game. This separation has more to do with computer adjudication in general rather than the particular linear computer adjudicator of the first run of the X-game. Computer simulations are not good at addressing human contexts. For example, X-game participants asked a contractor to put "a kidnapping" in a simulation. But, the result of that kidnapping was not believable, because how a population reacts to any event depends on the context, which they could not put in the simulation. The world is too complex to put all possible contexts into simulations before the particular details about a scenario are known. If basic processes of the model do not address the cruxt of a situation, and if the important factors are not even entered into a model, then the result is not valid.

An approach to dealing with the problem of context is quick turnaround agent based modeling. That is, to have data and many relevant models and modeling practices ready at the beginning of the study, and to write important parts of the model after the scenario is known, with the help of an agent based simulation toolkit. There are some 200 computational social models in academia, along with papers that describe them. Ideas from them that reflect social theories relevant to the scenario may be put into an agent based simulation during the preparation phase of a war game. Since the X-game had week long turns for extended study of actions, some of this time in the game can be spent in putting the context of new actions that modelers did not think of in advance into the agent based model. Quick turnaround agent modeling will allow creative moves to be put into the computer adjudication, and will be available for more excursions in the post-game analysis phase. Models will be available for future use, even if that use is just the borrowing of design patterns, so that the turnaround time will be quicker as the toolkit grows.

A Toolkit of Composable Modules for Quick Turnaround Adjudication

The social world is so complex that we can not build a general model of it that will be valid in every situation at the present state of social science. This complexity, in the context of irregular warfare, makes the quick turnaround approach less risky in terms of verification, validation, and accreditation (VV&A) than the traditional approach of using a previously VV&A-ed model and only changing the data. The model must also be modified to the particular situation.

Composable models also enable the switching in and out of social theories. With composable modules, the modeling phase would mostly consist of combining models or sub-classed models together according to research done on the scenario. If the models come in modules that roughly correspond to different social theories, then relevant sides of issues related to the scenario may be assembled together into plausible alternative social environments to branch on. Both sides of controversial issues would be represented. This is an important capability, as the US government can not test strategies against only one social environment when social scientists have not come to a consensus on what that environment is. Since the point of the game is to test the robustness of strategies, they should be tested against possible social environments as well as against possible enemy responses.

Agent based simulation should be used for this toolkit, not only because it can handle the micr-macro integration problems of irregular warfare better than SMEs can, but also because it is suited for recombination into new contexts. Agent based simulation is the most valid technique for quick turnaround causal modeling because it models with first principles, while other forms of adjudication tend to just model with correlated relations that do not apply outside of a particular context. When first principles or the root causes of phenomena are modeled rather than the appearance of phenomena, then a simulation becomes valid in new situations, and validity in new situations is an important requirement for a quick turnaround modeling toolkit. For example, X-game participants had difficulty with one of their computer models that simulated a "gravity point": a type of social homeostasis or healing, where the simulation goes back to the original state after a perturbation. However, because the model simulated the appearance of the homeostasis rather than the cause of the homeostasis, it occurred in every situation, even in situations where the point of the game was to find ways to prevent healing. If phenomena is simulated by assuming it is always true, then it can't be valid for exploring when it isn't true. In contrast, a good agent based simulation uses first principles and has fewer assumptions. Most phenomena in an agent based simulation is emergent from those few assumptions. Thus, agent based simulation is a technique that allows basic assumptions to be walked through in new contexts.

Combining Correlational Relations with Causal Models

In the first X-game, spreadsheets were used for computer adjudication because the agent based simulation was not ready. X-game participants put relations such as from the Fund for Peace's failed states index (Fund for Peace et al, 2007) into adjudication spreadsheets, combined with a random number generator to recognize that the relations were not always true. To put many relations from SMEs into a simulation so that they make sense together, modelers should no treat them as hard assumptions, as is done when relations are put into spreadsheet adjudicators. Modelers must take care to limit the number of assumptions in a simulation, and to understand the difference between correlational and causal models of social science, and how to apply both kinds of models to a simulation. Social science has thousands of studies of how one phenomenon correlates with another phenomenon. However, correlational relations can not be put into the assumptions of a simulation because they do not explain cause, and do not address why they are sometimes true and sometimes not. If they are put in as hard constraints, the model will not be able to explore anything outside of those relations, and the fact that the correlation coefficient is typically quite low in correlational studies of social phenomena makes it unrealistic to assume that many of them would be true at the same time. Moreover, correlational studies can not drive simulations : only causal models can do that.

Rather, correlational studies should be implemented as requirements to models of cause, or soft constraints on an answer that is determined by causal models. Observed correlations can and should be requirements for the higher order effects that result from theoretically based causes. Figure 1 illustrates composed causal models that are constrained in their outcome at designated points of correlation with events in other causal models.

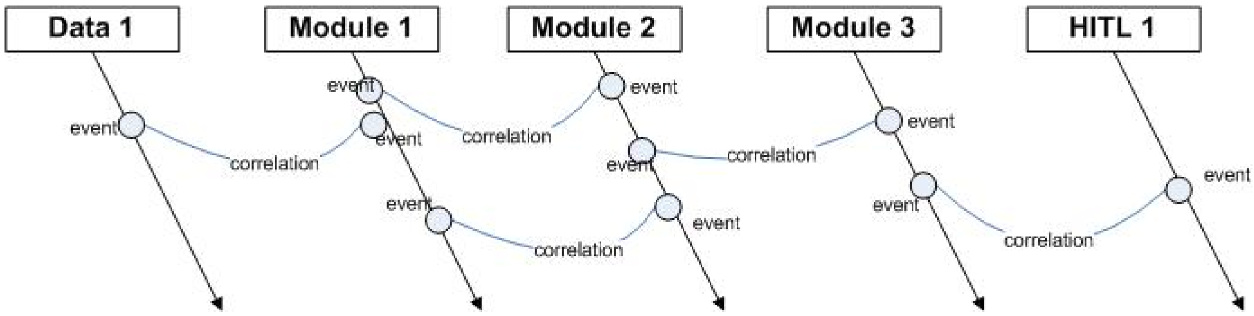


FIGURE 1 Correlational studies as soft constraints on models of cause, in a composition of models

A toolkit of composable models would cover both larger basic social theories of cause and smaller, more correlational studies. Causal theories would be expressed in simulation modules, while correlational studies would provide data upon which a designation of correlation between modules, or between modules and data, may be based. Designated correlations will place soft constraints on the consensus formed between sets of correlated modules at the places designated as correlated. Simulation modules would be synchronized at these places of correlation, so that they create a single picture of the social environment.

Fuzzy Rules to Implement Correlational Relations

Fuzzy rules are better implementations of soft constraining relations on simulation modules than spreadsheets are. Fuzzy rules work well with rich ontologies as distance metrics for simulation events. According to Weisel and Moya (2007), in order to be composed together, simulation events in different simulations must be related and described by a distance metric. Fuzzy rules can be used to decide whether a simulation event has come close enough to be designated as correlated with an event in another model, so that these models may be synchronized.

Fuzzy distance metrics can also represent gradient for use in data mining techniques and database retrieval of close cases, as is needed for case based reasoning. Fuzzy rules learned from HITL moves by data mining techniques such as the Center for Army Analysis' ACTOR program (O'Brian, 2003), can be added to hand-designed fuzzy rules from correlational studies, and put back into the composed model as soft constraints for post game runs without HITL. This would be useful for scaling, for example, to derive rules from war games of a few factions, and then test them on a more realistic simulation of 300 factions.

Adjudication rules in the form of fuzzy rules reflect the qualitative nature of social phenomena. As a method of soft computation, they are robust with respect to data, using the kind of verbal approximations that SMEs make. In a fuzzy system it does not matter if much of the data is approximate. Additionally, fuzzy systems can handle rules that contradict themselves, such as different SMEs or different correlational studies might make. Conflicting rules can just be added together.

Fuzzy cognitive maps can implement soft constraints leading modules to a consensus of the separate simulation models using constraint satisfaction methods. They can also implement the relations and feedback of systems dynamics models, another type of non-causal macro level model that can be used to put requirements on agent based simulations. With fuzzy rules, a quick turnaround toolkit can incorporate relations from spreadsheet and systems dynamics models, including feedback relations, into compositions of agent models.

Composing Simulation Models

The various simulations of major social theories, along with the designated points of correlation from correlational studies, will come to a consensus on a picture of the social environment, even though they may contradict each other. The phenomena in the different simulations will be made to correlate with each other at the correlation coefficients of published studies as much as possible. Individual simulations will have to adjust their states to the consensus state, and continue simulating from the point of consensus. This may involve iterative re-computations of simulation time until consensus is reached. Different simulations may be weighted differently in weighted voting schemes Since this is a constraint satisfaction problem, feedback such as in constraint satisfaction neural networks and fuzzy cognitive maps can help decide the consensus state. The National Science Foundation's Dynamic Data Driven Application System (DDDAS) program has developed techniques for feedback between data and simulations can also help.

The toolkit for composable models would have some qualities of a federation of independent models, and some qualities of a library of modules that need sub-classing in order to be instantiated to particular problems. As in federation composablity, modules would be viable without each other, for example, a model might use a draw from a distribution for phenomena it does not simulate, or if there is a designated correlation point it might use a draw that has been made to correlate with a draw from another simulation, or it might even replace the draw with an event from a trusted deterministic simulation. Since social phenomena are dependent on each other in the mathematical sense, theories can not switch in and out through neatly defined interfaces, as one might expect from a library. Figure 2 illustrates this point. Rather, there is functional overlap between theories, with different overlap depending on the theory. The same events, or correlated phenomena, in different simulations can be made to match up, synchronizing the simulations into one picture of the social environment at the designated points of correlation. Even replicated simulations, which are simulations of the same theories implemented differently, can benefit from coming to consensus through designated points of correlation, because artifacts of their implementation would be weeded out.

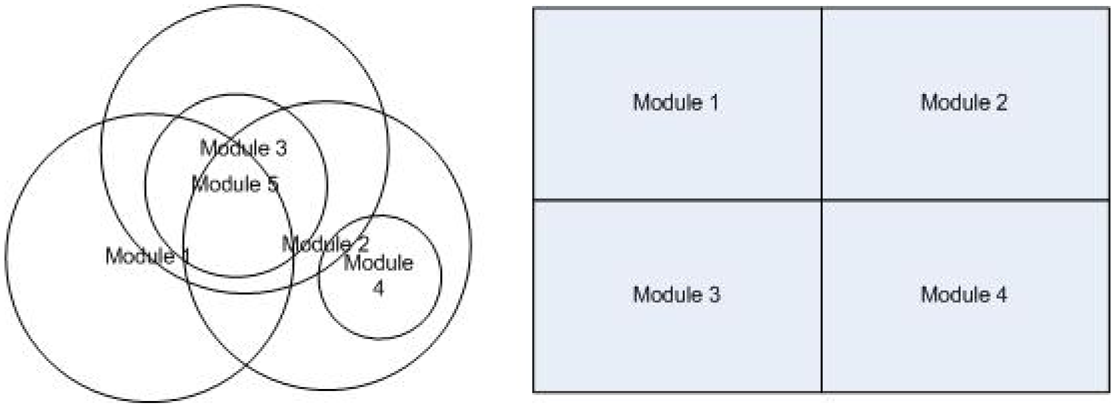


FIGURE 2 The modules of the toolkit will have functional overlap, on the left, as opposed to the traditional model of independent modules, on the right. Because social science theories look at the same phenomena in different ways, modules will be synchronized at their designated areas of overlap rather than through standardized interfaces.

Conclusion

Although SMEs have difficulty thinking about the macro effects of tactical moves in war games, they are still trusted more than computer simulation because they are better at understanding human contexts. A toolkit for composable models would give us many capabilities, including the ability to quickly adapt an agent model to the context of a move in the war game and the ability to switch different models of the social environment in an out. As more of the human context is put into a simulation, the simulation can replace the HITL game, so that irregular warfare strategies may be tested for robustness against many more scenarios than is practical for HITL games.

References

Dunlap, P., 2007, "Global War on Terror Analysis: An Extended Wargaming Approach," 75th MORSS Symposium.

The Fund for Peace and the Carnegie Endowment for International Peace 2007, "The Failed States Index 2007," *Foreign Policy*, August.

National Science Foundation, 2007, *Dynamic Data Driven Applications Systems* http://www.nsf.gov/cise/cns/dddas/

O'Brien, S. 2003, "Analyzing Complex Threats for Operations and Readiness (ACTOR)," 72cd MORSS Symposium.

Weisel, E. and L. Moya, 2007, "Towards an Architecture for Distributed Component Based Simulation," European *Simulation Interoperability Workshop.*

INTRODUCING GROWLAB: A TOOLKIT FOR LAYERED AGENT-BASED MODELING

N.B. WEIDMANN,* ETH Zurich, Switzerland
L. GIRARDIN, ETH Zurich, Switzerland

ABSTRACT

In this paper we introduce GROWLab (Geographic Research on War Laboratory) – a software toolbox to facilitate the modeling, simulation, analysis, and validation of complex social processes, with a special focus on geographic aspects. The paper aims to give a general, non-technical introduction to agent-based modeling with GROWLab. It focuses especially on the toolkit's support for model structure, i.e. the creation of complex agent configurations and hierarchies. An important feature of GROWLab is that it makes possible the integration of real-world empirical data collected with a GIS. More specifically, GROWLab can automatically create model structures from GIS datasets. This way, it is possible to run a model either on real or artificial geographies without changing the underlying data structures. We also introduce GROWLab's GeoModel, a geopolitical template model which makes different geographic and non-geographic datasets readily available to the modeler.

Keywords: Simulation toolkits, GIS, conflict process models, agent hierarchies

INTRODUCTION

As agent-based simulations are getting increasingly complex and sophisticated, simulation toolkits have become indispensable for the social science research community. Toolkits help the model designer to quickly setup, run and evaluate a model, without having to write the complete code from scratch. There are various toolkits out there (North et al. 2006; Parker 2001; Luke et al. 2004; Minar et al. 1996). However, these toolkits aim to be all-purpose products and try to target the entire broad field of social simulation. There is of course nothing wrong with creating an all-purpose toolkit. However, the fact that an all-purpose toolkit must necessarily remain general in order to make it applicable to a wide range of models limits the functionality offered to a particular subfield.

This paper reports on our efforts to create a simulation toolkit especially tailored to our requirements in the modeling of geopolitical processes. While developing a series of agent-based models in this field, we increasingly became aware of the shortcomings of general-purpose toolkits. We were often in need for agent structures more complex than grid spaces or simple networks. Also, as many of our models are increasingly relying on geographic data, we required support for different GIS data formats. Since there were quite a number of requirements common to all our models, it was natural to create a library including this functionality and in doing so avoid code redundancy.

* Corresponding author address: Nils B. Weidmann, International Conflict Research, ETH Zurich, 8092 Zurich, Switzerland; e-mail: weidmann@icr.gess.ethz.ch.

The result of these efforts is the “Geographic Research on War Laboratory” (GROWLab) library which supports modeling in the field of geopolitics and conflict research. The features of this toolkit might be useful to other disciplines as well, especially since it attempts to address challenges not specific to our area as for example the representation of agent hierarchies, and the integration of GIS data. In general however, the increased specificity of GROWLab as compared to other toolkits is likely to be useful to a narrower range of models.

Put very generally, ABM toolkits assist the researcher in three broad tasks: (1) setting up the *model structure*, (2) specifying the *simulation dynamics* and (3) collecting *output*. Model structure support is the storage and retrieval of (often different types of) agents and the representation of their relationships. Whereas model structure is about the static parts of the model and their relationships, toolkits also support simulation dynamics: What are the actions in the model, and when are they carried out? Here, the toolkit supports both scheduling within a single run of the model, but also more advanced executions such as batch runs across different parameter settings. Finally, toolkits usually provide considerable support when it comes to the collection of information from the model. Information about the current state of affairs can either be provided by graphical displays of the model space and dynamics charts, or can be collected as numerical output in files. This paper introduces GROWLab concepts and features along the three categories of *model structure*, *simulation dynamics* and *output collection*.

MODEL STRUCTURE

In many agent-based models agents live in a two-dimensional grid world. These Object2DGrids (in RePast) have two major functions: They store the agents themselves, and they define relations (such as neighborhood) between agents. Correspondingly, in GROWLab we introduce two interfaces capturing the two tasks: A *layer* is any collection of alike agents, and a *topology* is a set of relationships between them. In addition, in order to represent hierarchies of agents, we introduce the configuration interface. The following paragraphs explain layers, topologies and configurations in detail.

Layer

A *layer* is a container for a set of alike and atomic agents. Layers offer general functionality to manage the agents contained in them, but can also be used to collect aggregate data about the entire population. A layer itself does not know about the neighborhood relations of its agents – instead, this is achieved by imposing one or more *topologies* on a layer.

Topology

A topology is always defined on a layer of agents and defines a set of neighborhood relationships between them. In this sense, a topology is equivalent to a network. Based on the connectivity between agents, it can compute the neighborhood set of a given agents as well as their distance from each other.

Configuration

Whereas topologies can only exist between agents of the same kind, GROWLab offers the possibility to connect agents of different types to yield agent hierarchies. This is done using *configurations*, which typically connect agents from two layers – the parent layer and the child layer, as we call it in GROWLab. Configurations exist in different forms. The most general one is the many-to-many configuration, which allows the connection of a parent to many children, but also of a child to many parents. A more restrictive configuration is the one-to-many type, relating one parent to many children, but permits at most one parent per child. The one-to-one configuration adds the final constraint of only allowing exactly one child per parent.

A GROWLab model structure created with these building blocks is automatically kept in sync: For example, an agent removed from a layer is also removed from the topologies defined on that layer. Figure 1 illustrates the three core interfaces with a simple example of states and their provinces.

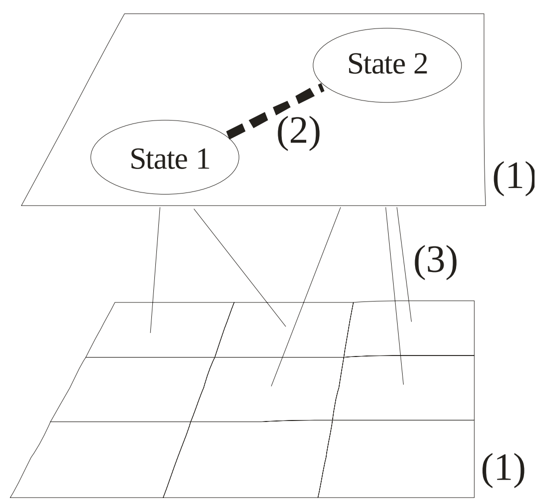


Figure 1: Illustration of the GROWLab model structure building blocks: (1) *Layers* are containers for agents. The top layer holds two states, and the bottom layer serves as a container for province agents. (2) A *topology* defined on the state layer keeps track of the relations between states. (3) A *configuration* stores the membership of provinces in states.

Spaces and Mappings

In order to represent agents in a spatial environment, we distinguish between a space which is an empty set of locations, and a mapping which takes care of the assignments of agents to locations in this space. This flexible design allows us to put agents at more than one position (e.g. states can occupy more than one province in a grid), or even to use one space for many different mappings. For example, this is useful when representing the extent of states and ethnic groups in the same geographic space: Only one space object is required, whose locations are then linked in two mappings.

GROWLab provides different types of spaces. On the one hand, it supports abstract spaces such as grids and hexagonal spaces. On the other hand, there is support for spaces with and explicit geographic reference, for example a GIS rastered space. Here, a location not only knows its x- and y-coordinates, but also its precise coordinates in latitude/longitude. Moreover, geographic spaces can compute the geodesic distance between locations.

MODEL EXECUTION

buildModel() and step() methods

Model execution in GROWLab follows closely the procedure introduced by RePast. Each model essentially needs to implement two methods: buildModel() and step(). The former is executed when a model run is initialized. Its purpose is typically to create data structures and agents required for this run. The latter is called at every time tick and contains the simulation steps to be run repeatedly. An important difference to RePast is the implementation of the simulation engine. In GROWLab, a simulator object takes care of initializing and running the model. The advantage of this approach is that one can select a simulator according to one's needs: GROWLab offers simulators with different graphical and batch run features.

Parameters

All parameters required for a model must implemented using the parameter classes offered by GROWLab. More precisely, a parameter is encapsulated in a special class that not only allows the storage of the parameter's value but also its name and description. All parameters are registered when the model is constructed initially such that they can be used both in graphical and non-graphical runs. Parameter classes exist for all kinds of numeric parameters, booleans, strings and enumerations.

Batch Runs

As stated above, the simulation toolkit must also provide a facility for automatic parameter sweeps, i.e. "batch runs" in RePast terminology. By automatically initializing and running the model for different values of the input parameters, the researcher can collect statistics about the behavior of the model under varying conditions. Batch runs rely on the set of parameters as described above. Batch runs can also be performed in parallel to get the results faster, both on multiprocessors machines and on distributed grid computers.

MODEL OUTPUT

Separating Visual and Batch Models

We encourage, and enforce to a certain extent, the developer to provide separate implementations of the model for graphical and non-graphical output. Essentially, apart from the basic Model interface specifying the buildModel() and step() methods, two extensions define how visual models and batch models should look like. A visual model will have to implement the buildUI() method where all graphical elements are set up. A batch model should provide information about which parameter sweeps are to be executed. Of course, GROWLab simulators tailored to either visual or batch runs will only be able to run the corresponding model. This structure makes sure that the behavior of the model – regardless of the desired way of output – remains the same.

Visualizing Model Structure

Model structure is displayed both as a graphical representation of model structures, like agents in a space, and a textual output with a detailed list of information about agents. The latter resembles the “probes” introduced by Swarm and still present in many other toolkits. For each of the three core concepts layer, topology and configuration introduced above, GROWLab has a set of predefined dynamic graphical visualizations for the inspection of the model. Layers can be portrayed by a list of agents and their attributes. Neighborhood relationships of a topology can be displayed graphically as a network structure, and textually as a paired list of connected partners. The structure of a configuration can be examined as a tree table. A set of two-dimensional graphical displays takes care of visualizing spatial layers and the agents contained in them.

Collecting Statistics from the Model

The way to extract statistics from a GROWLab model is done with the help of so-called “collector” classes. This mechanism is very flexible and can be used both for visual and batch models. Collectors are standardized data collection facilities storing the data in the format required for the analysis. For example, in a batch run one will typically use file-based collectors which simply output the assembled data to a file. For visual simulations, GROWLab offers collectors which prepare the data for display in a chart. Similarly, we provide a collector outputting a sequence of image files which can then be assembled to an animation of the simulation. Collectors are registered in the simulator executing the simulation. It takes care of activating the collector after each tick or at the end of a run.

Collectors get their data from variables within the model. However, as for the parameters introduced above, variables are implemented using the wrapper classes offered by GROWLab. Beyond the storage of a value these classes add meta-information about the variables such as their name and description. Additionally, variables can also be computed on the fly.

The GROWLab User Interface

During the development of GROWLab, special emphasis was put on the design of the graphical user interface. Our general approach is to have a GUI where multiple views on different aspects of the simulation are closely linked together. At the present stage, the GROWLab GUI features a set of interconnected views on the simulation, such as *spatial views* which display the simulation space, *configuration views* which allow for agent hierarchies to be displayed, and *process views* tracing the actions performed and the results produced in the model over time. Figure 2 illustrates the GROWLab user interface with the different views. The views are interconnected in such a way that selecting an agent in one view causes this agent to be displayed in another view.

Figure 2: The different elements of the GROWLab user interface: The spatial view (top right), the configuration view (top left), and the process view (bottom). There can be more than one view per type.

USING GEOGRAPHIC DATA IN GROWLAB MODELS

Using GIS Data for Agent-based Modeling

We can distinguish two ways of how GIS data can be used in modeling applications. The first category of models takes the geographical input as a realistic landscape where the model dynamics is then run on. The crucial feature of this approach is that typically the geographic data remain constant throughout the model run. Examples include the creation of a realistic road network to run traffic simulations. The second category of simulations is more complex. Here, the geographic data is not kept constant but rather endogenous to the model. For example, in all Geosim-like models state borders vary over time. In order to be able to represent these changes, we need a data format which is able to accommodate time-variance in geographic features.

The models GROWLab is designed for typically belong to the second category. Whereas in GIS vector data the degrees of freedom for changes are unlimited, in a raster-based representation this complexity is significantly reduced. For example, a country represented as a polygon can be modified by moving the polygon's corners, or by adding or removing existing corners. Obviously, the possible alterations are infinite which makes a vector format less well applicable for simulations with an endogenous geography. On the other hand, we could represent a country as a (mostly contiguous) set of raster cells. The tradeoff we incur is the lower resolution and precision, but since the atomic spatial unit – the grid cell – is fixed, changes to the shape of the country can be represented as a re-assignment of the spatial units to other states.

GIS and GROWLab data structures

GROWLab is able to read GIS data and to create its own data structures from it. The development of a model with geographic reference typically starts with the definition of a geographic space – a raster space with geographic reference. All spatial data added to the model uses this space as a reference.

Raster data to be included in the model has to be provided at the same resolution as defined in the underlying space. It is then up to the user to tell GROWLab which kind of data structure it should create from a raster input file. For example, a raster of countries (where cell values indicate the country a cell belongs to) is best represented as a one-to-many mapping of a country object to locations in the space, in other words, an assignment of country objects to locations where each country occupies more than one location. To give another example: When representing ethnic groups and their location, we use a many-to-many mapping of groups to locations. Obviously, a group can occupy many locations, but one location can also be shared among different groups.

The data structures briefly described here enable the researcher to craft an agent-based model with geographic reference according to one's needs. However, if only some standard GIS datasets are required, one can also rely on a readily implemented template model.

GeoModel: A Geopolitical Template Model

Based on the template model GeoModel, geo-coded real-world data can be integrated in the modeling process. This template can be extended by inheriting the built-in functionality and by adding some custom behaviors and mechanisms or complement it with additional layers of data.

GeoModel's default space is a rasterized representation of the entire globe, using the WGS84 projection. The raster can be used in two different resolutions: 15 arc-minutes (~30km), and 30 arc-minutes (~60km). All the geographic data is based on this space.

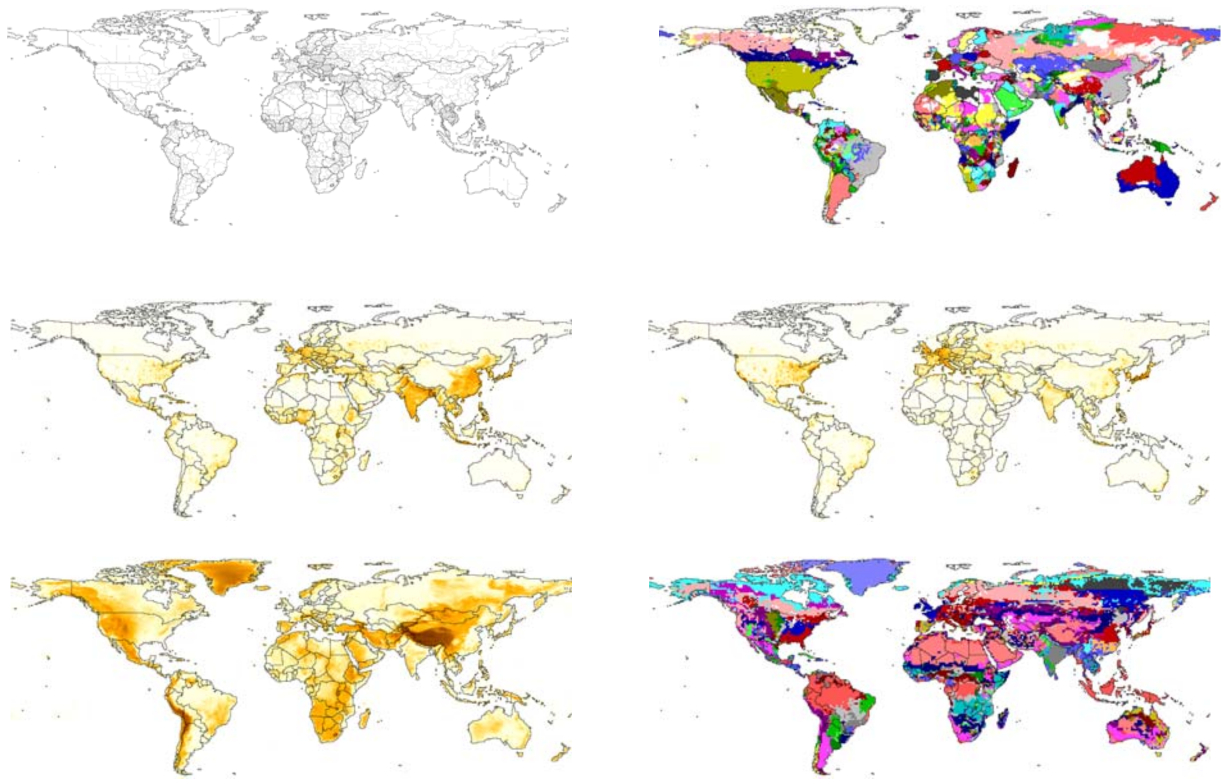


Figure 3: Geographic data contained in the GeoModel template: borders, ethnic groups, population, spatial GDP, elevation and vegetation (left to right panel).

Figure 3 shows some of the information contained in the GeoModel template including (1) country border and administrative divisions, (2) ethnic groups across countries, (3) population density, (4) spatial GDP figures, (5) elevation data, and (6) vegetation type. For each country, we provide their borders as of 1964 and 1994, and also try to reconcile their ISO, FIPS and COW codes through customized mapping. To check adjacency of countries, the Minimum Distance data from Gleditsch and Ward (2001) is also included to query for neighboring countries that are separated by water. At this point, all ethnic groups are directly based on the GREG definitions. For each ethnic group in a country, there is also information about the "ethnic group in power" (EGIP) coding by Cederman and Girardin (2007).

In addition, we provide disaggregated data for every cell in the system for population (downsampled from the Gridded Population of the World v. 3 provided by CIESIN (2005)) and elevation (downsampled from GTOPO30 (2007)), as well as local GDP estimates, compiled by the G-Econ project (at a 1-degree resolution) from Nordhaus (2006). They relieve the modeler from the tedious task of having to collect and merge complicated datasets and thus provide a prototyping environment for geographic agent-based models.

EXAMPLES IMPLEMENTED IN GROWLAB

GROWLab comes with some twenty models that serve as example that can be used as template for developing new models, as test cases to verify the inner working of the simulator, as well as stereotype models to evaluate the effectiveness of GROWLab architecture.

To give few examples, we use the iterated prisoner's dilemma model (Cohen et al. 1998) as an example for teaching purpose and to test GROWLab basic spaces (torus, grid, soup) and neighborhood functions. Schelling's segregation model (Schelling 1978) is used to showcase and test moving agents. For more elaborated agent structures, we mainly rely on the Geosim model (Cederman 1997) , which features hierarchical agents and moving state borders. To showcase the GeoModel template model, we provide some statistical and exploratory models at the forefront of research.

CONCLUSION

In this paper we presented our GROWLab simulation toolkit. With GROWLab, we tried to improve the computational infrastructure for geopolitical models, especially with regard to the use of geographic data. We feel that our approach of using native GROWLab data structures to represent GIS data is promising, as it does not require the use of special GIS classes in the model. This way, one can for example switch back and forth between an artificial space and a space with geographic reference in the same model.

An issue which we have not yet taken into account is the incorporation of real time in the model. Many GIS datasets are available with explicit temporal coordinates, as for example ACLED (Raleigh and Hegre 2005). Further development of GROWLab will also focus on a better support for different agent activation regimes: In our models, synchronous updating is mostly the desired scheme, but in many cases it brings with it a lot of computational problems. A toolkit could provide implementations of agent prototypes with built-in data structures for synchronous activation.

ACKNOWLEDGEMENTS

The authors are grateful to Lucas Serpa Silva for research assistance. Nils Weidmann's work is supported by ETH (Research grant TH –4/05-3).

REFERENCES

Cederman, Lars-Erik. 1997. *Emergent Actors in World Politics: How States and Nations Develop and Dissolve*. Princeton, NJ: Princeton University Press.

Cederman, Lars-Erik, and Luc Girardin. 2007. "Beyond Fractionalization: Mapping Ethnicity onto Nationalist Insurgencies." *American Political Science Review* 101 (01):173-85.

Center for International Earth Science Information Network CIESIN, Columbia University; and Centro Internacional de Agricultura Tropical CIAT. 2005. "Gridded Population of the World v3 (GPWv3)." Available at http://sedac.ciesin.columbia.edu/gpw.

Cohen, Michael D, Rick L Riolo, and Robert Axelrod. 1998. "The Emergence of Social Organization in the Prisoner's Dilemma: How Context-Preservation and Other Factors Promote Cooperation." *SFI Working Papers*.

Gleditsch, Kristian Skrede, and Michael D Ward. 2001. "Measuring Space: A Minimum-Distance Database and Applications to International Studies." *Journal of Peace Research* 38 (6).

Luke, Sean, Claudio Cioffi-Revilla, Liviu Panait, and Keith Sullivan. 2004. MASON: A New Multi-Agent Simulation Toolkit. Paper prepared for the SwarmFest Workshop.

Minar, N., C. Burkhart, C. Langton, and M. Askenazi. 1996. "The Swarm simulation system: a toolkit for building multi-agent simulations." Santa Fe. SFI Working paper 96-06-042.

Nordhaus, William D. 2006. "Geography and macroeconomics: New data and new findings." *Proceedings of the National Academy of Sciences USA* 103 (10):3510-7.

North, M. J., N. T. Collier, and J. R. Vos. 2006. "Experiences Creating Three Implementations of the Repast Agent Modeling Toolkit." *ACM Transactions on Modeling and Computer Simulation* 16 (1):1-25.

Parker, Miles T. 2001. "What is Ascape and Why Should You Care?" *Journal of Artificial Societies and Social Simulation* 4 (1).

Raleigh, Clionadh, and Havard Hegre. 2005. Introducing ACLED: An Armed Conflict Location and Event Dataset. Paper prepared for the Disaggregating the Study of Civil War and Transnational Violence Conference, San Diego, CA.

Schelling, Thomas C. 1978. *Micromotives and Macrobehavior*. New York: Norton.

US Geological Survey. 2007. "GTOPO30 Digital Elevation Model." Available at http://edc.usgs.gov/products/elevation/gtopo30/gtopo30.html.

IDEAS - INTERACTIVE DEVELOPMENT ENVIRONMENT FOR AGENT-BASED SIMULATION

A. PERRONE ,* University of Venice, Venice, IT
A. PELLIZZON,+ University of Padua, Padua, IT

ABSTRACT

Recent discussions in the agent-based modelling community have showed disparate opinions about the level of programming skills to be expected from modelers. Everybody seems to agree that the more skill candidates have, the better. However, most of today's students lack these capabilities and developing them requires substantial efforts from the adventurous entrepreneur. Therefore, lowering the requirements would help agent-based modeling becoming a more widely accepted methodology.

Some suggest that with the spread of general computer literacy, the problem outlined above will eventually go away, sooner than we would imagine. Others argue that the problem is inherently social: mastering mathematics or statistics is not in the least easier than learning to program, still no aspiring scientist can afford to avoid it.

It would be hard to deny any of these arguments. Nonetheless, we argue that today's requirements can be lowered. This statement is not very surprising either. Various model building tools (such as NetLogo or AgentSheet) demonstrate that by limiting the 'space' of possible models, the task of modeling can be efficiently assisted. The real challenge is to bridge the gap between the potential open-ended nature of Swarm-alike modeling environments (e.g., Swarm, RePast, Ascape) and the ease of use provided by the former frameworks.

Graphical model building interfaces for general ABM platforms, such as SimBuilder (for RePast) and the IDEAS are attempts to achieve exactly this. Still, they impose certain limitations on the modeler and require a certain level of programming.

This contribution describes the IDEAS, an interactive IDE for Agent based simulations.

Keywords: Agent Based Modelling, IDE, RePast, Swarm Simulation Toolkit, Java

INTRODUCTION

Agent-based economic simulations are becoming part of the professionXs toolkit. Two recent special issues of the Journal of Economic Dynamics and Control and Computation collect

* *Corresponding author address*: Alessandro Perrone, University "Ca' Foscari" Venice Italy, e-mail: alex@unive.it

+ *Corresponding author address:* Andrea Pellizzon, Department of Mathematical Methods and Models for Scientific Applications, University of Padova, Italy e-mail: andreap@dmsa.unipd.it

several contributions of agent-based economics. They are the best evidence of the increased interest expressed by the profession.

As stressed by Testfatsion in her introduction to the JEDC special issue, agent-based simulations address at least four issues. First, heterogenous agents and interactions among them can be explicitly modeled in agent based simulations. Second, agents behaviour can change due to the interaction with the environment and other actors. Third, evolutionary processes can be implemented at the agent level, rather than dictated by population-level laws of motion. Finally, these models can lead to genuine emergence phenomena and provide a way out of the clockwork dynamics usually given by more traditional models.

Unfortunately, however, the spreading of this new tool is slowed down by the absence of a common language as underlined in Luna and Stefansson (2000) and Luna and Perrone (2002). This limitation prevents a cumulative learning process and finally the emergence of an agent-based school of thought.

This contribution describes the IDEAS, an interactive IDE for Agent based simulations

MOTIVATION

Initially we posed ourselves the questions:

- Why develop yet another IDE when good systems already exist (e.g., Eclipse[1] , Bluej[2] , Jbuilder[3] , NetBeans[4] , Jcreator[5])?
- Is there the need on the community of a new IDE? Will someone use this IDE?

Why another IDE?

This project started 3 years ago, when the use of IDE was not so "popular" as nowadays.

What's an IDE? Upon a standard definition "IDE, integrated development environment is a system for supporting the process of writing software. Such a system may include a syntax-directed editor, graphical tools for program entry, and integrated support for compiling and running the program and relating compilation errors back to the source.

Such systems are typically both interactive and integrated, hence the ambiguous acronym. They are interactive in that the developer can view and alter the execution of the program at the level of statements and variables. They are integrated in that, partly to support the above interaction, the source code editor and the execution environment are tightly coupled, e.g. allowing the developer to see which line of source code is about to be executed and the current values of any variables it refers to." We have tried, over the course of the last few years, just about every Interactive Development Environment out there to build the simulation. In my opinion they all share two things.

1. They try to do too much, which makes them all large, slow and painfully hard to use.
2. They force the user to change the way he works. He wants an easy to use development system that works without any particular programming skill and "visually" easy to use.

That's why we have developed this IDE. With this package anyone, with or without specifics skills on Agent based Modelling, can create models using

---

[1]http:/www.eclipse.org
[2]http://www.bluej.org
[3]http://www.borland.com
[4]http://www.netbeans.org
[5]http://www.jcreator.com

- objective-c swarm
- javaswarm
- RePast

Most important reasons we choose to program an IDE for agent based modelling can be summerize into following points:

- First of all, there's not an unique IDE which permits to build models in 3 ABM environments (Javaswarm, Objective-C Swarm and Repast).
- The Size of the program.

environment	Size in Kb	Support Objc-swarm	Support Javaswarm	Support RePAst	Support other ABM
Eclipse Classic	137 Mb	NO	YES	YES	YES
Jbuilder	506.8MB	NO	YES	YES	YES
NetBeans	83.8 MB	NO	YES	YES	YES
IDEAS	588Kb	YES	YES	YES	YES

Table 1: Comparison among different Java IDE

According to table 1[6] the dimension of this package is very small and so it does not require computer with high performance.

- It is very easy to insert other ABM environment in this IDE, since it is all coded in java, the code is well documented. All the project of this IDE is Open Source.
- This tools permits in an easy way, to create tutorial for each language, without writing a lot of lines, but using only commands in menus.
- Each model is composed by a project, and it has multiwindow visibility of the code of the simulation.

Was there the need on the community of a new IDE? Will someone use this IDE?

In our opinion, there was the need of such tools, due to the fact, that it encapsule all three ABM environments in "less than 1 Megabyte" program. Anyone can generate its model (even if it has no experience of either computer programming and ABM philosophy) using all three libraries and then, seeing the code, can choose, among them, which one he could use.

With single clicks of mouse he can create the skeletron of a large functional model, and then he can spend all the time to modify the working skeleton upon its wishes.

The package is totally written in Java, and so it virtually runs on every operating systems which has java virtual machine installed[7] .

[6]All datas of this table has been taken from the official web page of the developer of each IDE
[7]The IDE has been written on a Macintosh powerbook using Tiger as Operating system, and on a Intel notebook with Linux Fedora

IDEAS PACKAGE

The figure 1 shows a snapshot of the IDE.

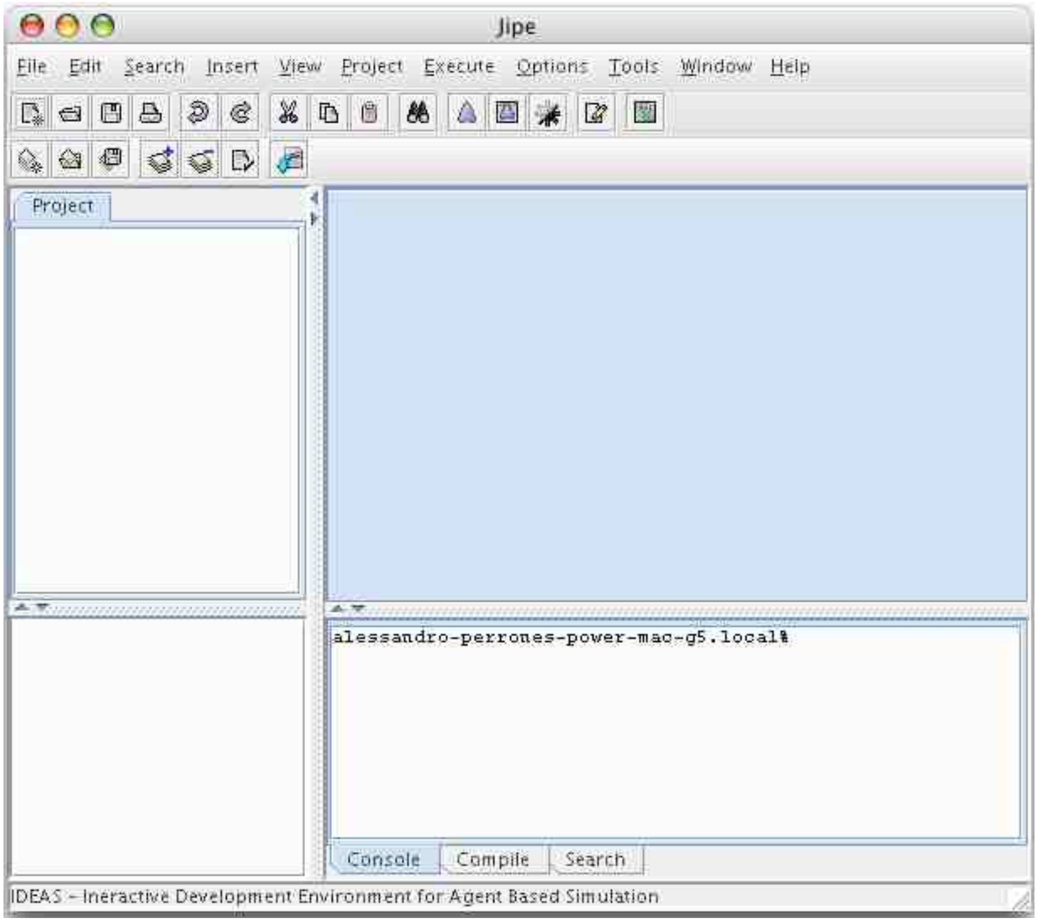


Figure 1: A screenshot of main IDEAS window

IDEAS Overview

To run the program, simply double click on IDEAS icon and a window like that in figure 1, appears. The Workbench window displays one or more perspective. The worbench window is divided into four different windows:

1. **A project Window** that contains the collection of items that make up the particular simulation. The project window displays a list of these elements to give easy access to them. In a standard simulation project, for example, according to swarm paradygm, there's the main, the ModelSwarm, and the ObserverSwarm objects, while, in RePast Model, it contains the Model, ModelGUI and standard Agent Classes. It provides window access to project resources through the logical view.
2. **A Navigation Window**. it displays the members classes as well as the Inheritance view of class methods and variables of the currently open class.

3. **A Code Editor Window**. The IDE's built-in source editor enables to view, create and edit the source code. It assist the programmer while coding with features like sintiax highlighting, code completation[8] , and editor hints.
4. **A console Window**. It displays compiler messages while building and running the project, and it is an interface to the Operating System.

IDEAS's feautures

Some of the features of this package are

- **Intuitive Interface** This package has been coded and thought with the GUI of standard IDEs.
- **Project manager** . All simulations are encapsulated into projects
- **A language sensitive editor**. All java and C language reserved words are highlighted.
- **Graphical display of the class structure**[9]
- **Compilation facilities**. Anyone can write particular flags that they will be used during the compilation of the project.

  • **Code Generation** Design your application visually. IDEAS generates all the Agent based toolkit code[10] ,so no developer time is wasted coding a user interface. . **Multiplatform**. It's been intentionally written totally in Java, so it runs anywhere where Java runs **Code Editor**, It generates 100% Pure editable code.

This paper gives an overview ofK the environments showing how is simple to write simulation using different Agent based environments, even if there's a little knowledge of the "agent based paradygm".

Working with IDEAS

In this section provides a quick tutorial which takes the user along the steps required to create a couple of simulation with simple clicks of mouse. In this paper we do not describe all menus of the package, since the package comes with a well-written manual which cover all aspects of the IDE.

Our first experiment - hello world

In this paragraph there'll be explained all steps to follow to obtain a simulation whose output is showed into the figure 2

---

[8]In this version this features runs only on ABM in java, but in near future it will be extented to other ABM toolkits.

[9]only for the java ABM

[10]It generates, depends on the kind of the running project

- **java code** for javaswarm and RePast environment,
- **objective-c code** for objective-c swarm toolkit

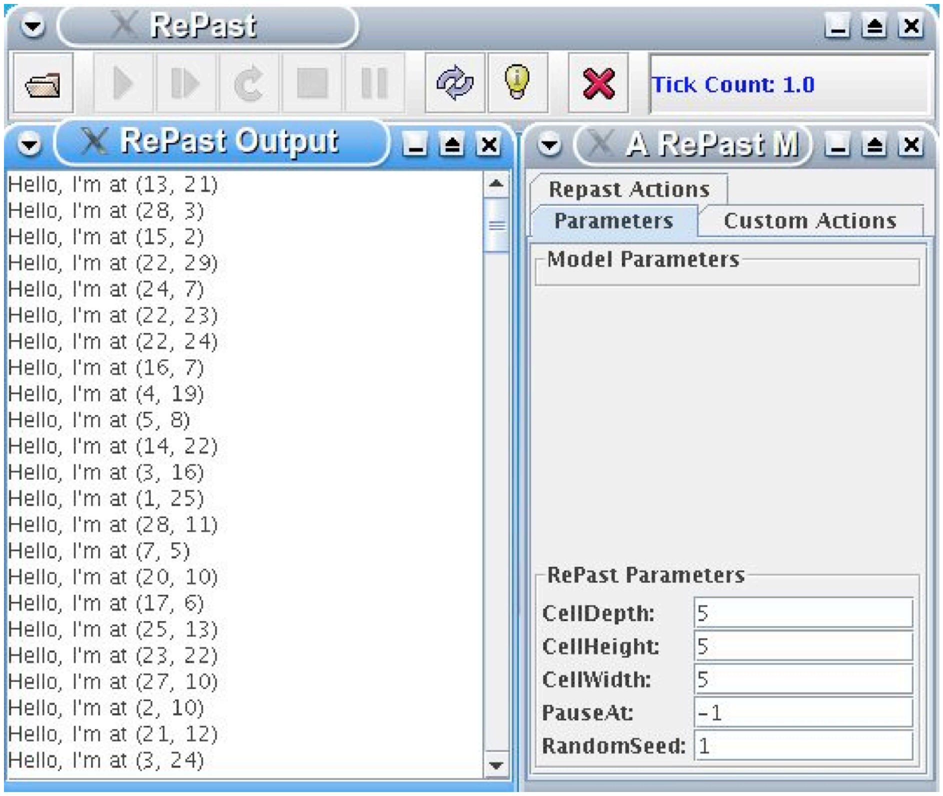


Figure 2: **Output of the first experiment**

Here are the steps to follow:

1. Create a new project by double click Project Menu -> New Project Item. A window like that in figure 3 appear

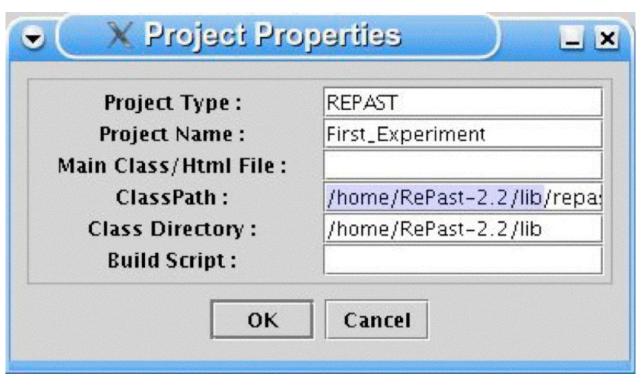


Figure 3: **The dialog box displayed when you start a new project, or want to alter a projects properties.**

When starting a new project the first thing you will be asked for some information about the model you want to program such as

- **Project type.** It it the type of ABM the user wants to program. This item accepts, in this version, only three kind of ABM projects

- JAVASWARM to use javaswarm paradygm toolkit;
- OBJCSWARM for objective-c swarm toolkit;
- REPAST to use Repast environment;

- **Project Name**. This project name will then be displayed in the project window.
- **Main Class.** It is the name of the simulation.
- **Other properties** as particular classpath, buildpath.
- **BuildScript.** It is useful for objective-c to run the program.

2. Once filled the items of figure 3, click the Ok button[11] and the framework will be presented as the picture of figure 4

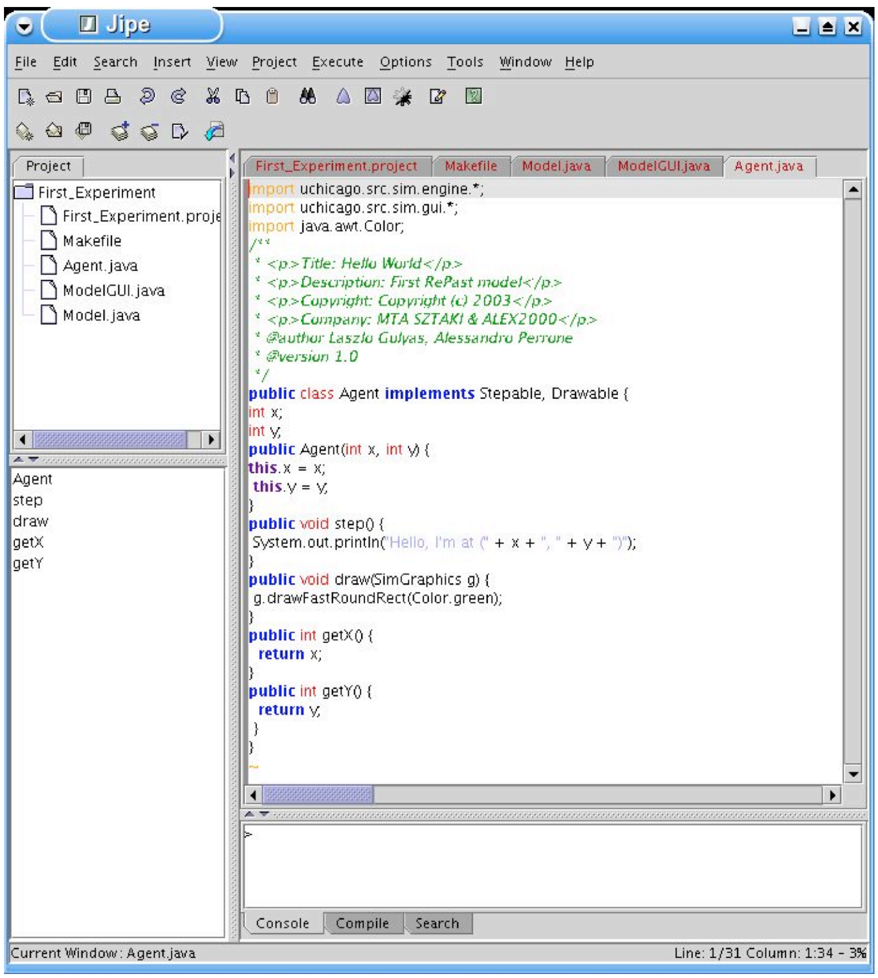


Figure 4: **The IDEAS framework of the first experiment**

3. Save each file of the project, using "save as" item of Menu FIle, or choosing each file on Code Editor, pressing right button of the mouse, and choose the save item.
4. Save the project using the "save project" item under the Project Menu.

---

[11]It is important to fill the classpath item, because it tells the compiler the PATH to find the RePast library, for this simulation, the swarm.jar file, if the simulation is based on javaswarm toolkit

5. Compile the project using F9 shortcuts, or using compile item under Execute Menu[12]
6. Run the project using the "run app" item

The output of this first experiment is showed on the figure 2

It is important to note that, without writing a single line of code we have written a simple running simulation which can be used as skeleton for our real model.

An quite-complete experiment

In this section we describe a simple experiment, in which we can see some GUI features of the package[13] It produce, as output, the same as last paragraph, but with two more windows, one with a barchart and the other with a graph of the variable X versus time. Here are the steps to follow:

- Create a new project, fill the item of window like figure 3 and name the project as "SecondExperiment"
- Insert a linechart using the "Line X" item of Insert Menu
- Insert a barchart using the "Bar Chart" item of Insert Menu
- Save each file of the project, using "save as" item of Menu FIle, or choosing each file on Code Editor, pressing right button of the mouse, and choose the save item.
- Save the project using the "save project" item under the Project Menu.
- Compile the project using F9 shortcuts, or using compile item under Execute Menu[14]
- Run the project using the "run app" item

The output of this experiment is showed in the figure 5

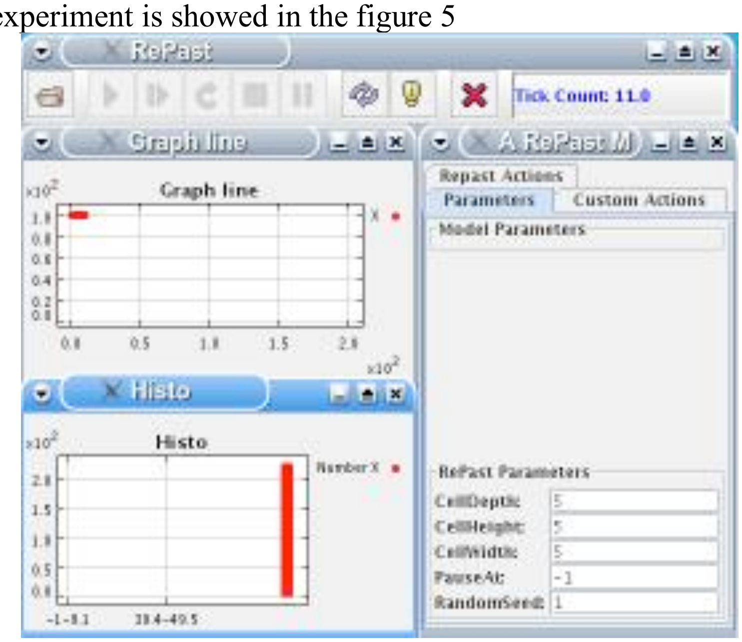


Figure 5: **The output of the second experiment**

[12]In the same menu, the Execute, there's also the "Build All" command which compile all files of the project

[13]We are still, as example, using the RePast toolkit

[14]In the same menu, the Execute, there's also the "Build All" command which compile all files of the project

Other experiments

In this example we have chosen to write a simulation using RePast, but, in the same way, we could write the same model using javaswarm, [15] of the objective-c swarm[16]

In these examples, we have intentionally chosen to write simple simulations without writing any lines of code, so the reader can duplicate the experiment using different ABM toolkit without problems.

The packages is provided with several projects of different toolkit which come from Internet world[17] .

It is important to note, that in this paper is described some of all features of the package. Using project manager it is possible to edit all ABM files, create new classes, edit classes, insert code, etc. etc.

IDEAS is an Interactive Development Environment customized for Agent based modelling.

CONCLUSIONS AND FUTURE WORK

In this paper we have described the implementation of IDEAS, an IDE written specifically for Agent based modelling using tools as Swarm (either in objective-c and java language) and RePast frameworks.

IDEAS provides the user with a wide range of functionality such as : Project management, project templates, code-completion, debugger interface, editor with syntax highlighting, wizards and a fully customizable user interface.

We believe that IDEAS has tremendous potential in solving many of the problems faced by researchers regarding lack of computer programming experience.

Therefore, we view the work described in this paper as only the beginning of a large project. We intend to develop a complete implementation of all different aspects of the ABM programming along with the implementation of other ABM environments.

We note that the IDEAS interface defined in this paper may change as our implementations and studies reveal the need for providing additional/different functionality.

As the project progresses we intend to consider the use of sourceforge.net as a software repository, since we would like IDEAS to be open source

As a future work we are going to implement some new fautures of the program such as:

- better integration with repast and swarm toolkit;
- better colour syntaxing for reserved words in Java, C, objective-c Language;
- Color coding that can differentiate between objects and methods
- integration of gdb debugger for c/objective-c models
- easy to save single projects and single classes
- better facility to load /save/run models
- integration of "jarfile Manager" for java models and "zip manager" for other ABM languages

---

[15]Just write into field "ProjectType" of the figure 3 the word "JAVASWARM"

[16]Just write into field "ProjectType" of the figure 3 the word "OBJECTIVEC"

[17]Examples in RePast can be found into RePast official web site http://repast.sourceforge.net, while swarm models can be found into Swarm Web site www.swarm.org.

- support for backmapping for objective-c language, which means that if you click on the error message in the Output Window, IDEAS will highlight the line of code that generated the error
- better documentation
- more examples in different ABM environments

The package is a very dynamic work in progress. For an update on the environment development, to download the latest release or for updated information about the VSB, including sample code, errata and preview of further versions, forums visit my site http://www.planetagents.org in IDEAS section.

REFERENCES

Apple Computer Inc. Inside Macintosh, 1985 Addison Wesley

Axtell, R Epstein, Growing Artificial Societies

IBM - The NetBeans€ Tools Platform: A Technical Overview. 2001. http://wwws.sun.com/software/sundev/whitepapers/netbeanstp.pdf]

IBM, Visual Age for Java, version 2.0, 1995

Luna, F and Perrone A: 2001 Agent Based methods in Economics and Finance: Simulation in Swarm, Kluwer Accademic Press

Luna, F and Stefansson: 2000, Economic Simulations in Swarm, Kluwer Accademic Press.

Minar, N, Burkhart, R., Langton, C. and Askenazi, M. (1996). The Swarm Simulation System: A Toolkit for Building Multi-Agent Simulations. Santa Fe Institute Working Paper 96-04-2.

Perrone A.. Agent Based design modelling using the Visual Swarm Builder presented on *"Agents in Design 2002" MIT Cambridge* MA 29-31 August 2002

Perrone, A. SwarmJournal - 2001

Pinson, Wiener, 1991 “Objectice-C, Object-Oriented Programming Techniques”, Addison Wesley Publishing company

RePast. A software framework for creating agent based simulations using the java language. http://repast.sourceforge.net

Stefansson, B. (1997). Swarm: An Object Oriented Simulation Platform Applied to Markets and Organizations

Terna, P. (1998). Simulation Tools for Social Scientists: Building Agent Based Models with SWARM. *In Journal of Artificial Societies and Social Simulation*.Vol 1 No. 2, available at http://www.soc.surrey.ac.uk/JASSS/1/2/4.htm l

Tesfatsion, L. (2001). “Introduction to the special issue on agent-based computational economics”. In *XX Journal of Economic Dynamic and Control*, Vol. 25, Issue 3-4, pp.281-293.

Visaj - The Visual Application Builder for Java. Imperial Software Technology

NetLogo Toolkit Developments

TURTLE HISTORIES AND ALTERNATE UNIVERSES: EXPLORATORY MODELING WITH NETLOGO AND MATHEMATICA

E. BAKSHY*, Northwestern University, Evanston, IL & University of Michigan, Ann Arbor, MI
U. WILENSKY, Northwestern University, Evanston, IL

ABSTRACT

This paper presents the design of a development platform integrating NetLogo, a multi-agent programmable modeling environment with the *Mathematica* scientific computing environment. We will discuss the affordances of such environments, which can simplify and enrich the research process for agent-based modelers. More specifically, we will demonstrate the advantages of having real-time exchange of complex data structures between agent-based modeling environments and symbolic mathematical software such as *Mathematica*. Together, such tools can provide researchers with a highly interactive, self-documenting workflow that neither tool can provide alone. This paper will give an overview of how the integrated environment can be used for common tasks in agent-based modeling, the construction of interfaces for exploring simulation dynamics, and the effective design patterns for representing simulation results.

Keywords: Agent-based modeling, exploratory analysis, NetLogo, *Mathematica*

INTRODUCTION

The behavioral dynamics of agent-based models contain vast quantities of information for which analysis can often be daunting to researchers. Nevertheless, for the purposes of model verification, validation and replication, it is essential for researchers to carefully and extensively study their models and analyze the behavior at several different levels (Wilensky & Rand, 2007). This paper presents a framework for representing, measuring, and visualizing the behavior of agent-based models. We will discuss some limitations of software systems used in the development and analysis of agent-based models, and demonstrate the ways in which our framework attempts to address these issues. We propose that many of these tasks can be resolved through the integration of agent-based modeling environments and scientific computing environments, such as NetLogo (Wilensky, 1999) and *Mathematica* (Wolfram, 2003). Our approach is, in some respects, similar to that of Macal & Howe (2005), which provides an

---

*Corresponding author address: Eytan Bakshy, School of Information, University of Michigan, 1075 Beal Ave., Ann Arbor, MI 48109; e-mail: ebakshy@umich.edu

extensive engine-level integration between Repast (2007) and *Mathematica*. We will further elaborate upon the affordances of such environments in the context of current issues faced by agent-based modelers.

Mathematica is an interactive programming environment which can support many of the tasks common to agent-based modelers. These tasks include pre-processing and analysis of external data used to motivate or calibrate models, model prototyping, interactive model exploration, data collection, storage, analysis, and documentation among other tasks. In contrast to using several special purpose or compiled programming languages for each of these tasks, the integration of such tools with high-level agent-based modeling environments like NetLogo can bridge the gap between model development, inquiry, and analysis.

Mathematica essentially consists of two processes, the kernel and the front-end. The kernel stores and executes all program code and data, which are represented in a uniform fashion as *expressions*. The front-end allows users to manipulate, retrieve, and graphically represent expressions stored in the kernel. Users interact with a *notebook*, which can contain text, program code, data output, and graphics. Typically, users enter commands into the notebook, which are executed by the kernel, and its output is displayed below, giving a line-by-line documentation of a user's session. In *Mathematica 6* (2007), notebooks can contain dynamic elements, which can display the state of expressions in the kernel in realtime, as well as relay information from interface objects back to the kernel.

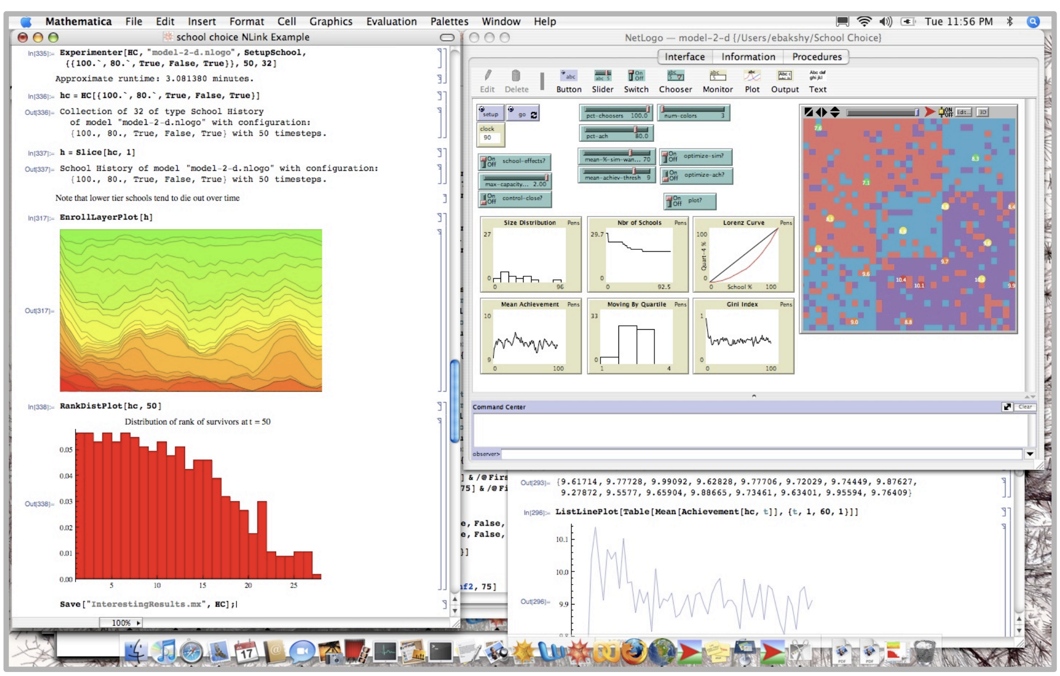


FIGURE 1 The NetLogo-Mathematica Modeling environment

The integrated NetLogo-Mathematica environment, depicted in Figure 1, includes many aspects that make it particularly well suited for conducting research with agent-based models. *Mathematica*'s data connectivity supports automatic format recognition and type conversion of files, as well as support for SQL database connectivity. In conjunction with pattern matching and rule-based programming functionality, such routines can reduce the amount of time spent preparing and organizing data for use with NetLogo. This integration makes accessible, for use with NetLogo models, *Mathematica*'s functions for statistics, non-linear optimization, linear algebra, graph theory, and a number of other functions suited for the execution and analysis of agent-based models. These methods can be combined with high-level graphical interface constructs to rapidly create custom tools for exploratory analysis of models. The environment's document-centered interface lets users combine comments, code, visualizations, and annotations in a single working notebook that can be viewed side by side with the NetLogo graphical interface. Finally, because all definitions, data, and graphics are serializable, the storage and retrieval of complex data structures representing model data (e.g., simulation histories) can be accomplished with minimal effort. These technical aspects of *Mathematica*, combined with the NetLogo-Mathematica interface, provide a flexible foundation upon which agent-based research frameworks can be built.

OVERVIEW OF THE NETLOGO-MATHEMATICA INTERFACE

The NetLogo-Mathematica toolkit provides a high-level interface to NetLogo from the *Mathematica* kernel via the J/Link Java interface. Once installed, one can load the package and launch NetLogo with no additional configuration.† At its core, the interface comprises of two simple functions: `NLCommand[]`, which executes a NetLogo command, and `NLReport[]`, which returns data from NetLogo. Other high-level primitives for repetitive tasks and acquiring structured interaction topologies, such as patches or grids (via `NLGetPatches[]`), and links or networks (via `NLGetGraph[]`) are included in the toolkit as well.

`NLCommand[]` is often used to programmatically initialize a model and execute the main loop. The function performs automatic type conversion, expression splicing, and concatenation, which allows users to easily access or modify NetLogo data using any combination of numerical, string, boolean, color, and list expressions. Additionally, numbers, strings, and lists are automatically converted back to native *Mathematica* types when requested from NetLogo using `NLReport[]`. For example, `NLCommand["set foo", {{True, 12, 8.4}, {False,`

---

† requires NetLogo 4.0 or greater and licensed version of *Mathematica* 6.0

`13, 8.9}}]` will set the NetLogo global variable, `foo`, to the NetLogo list expression, `[[true 12 8.4] [false 13 8.9]]`. Similarly, `NLReport["foo"]` will return a *Mathematica* expression containing the original nested list of boolean, integer, and floating-point types. These two basic functions are often sufficient to collect data and create plots on the fly that might otherwise require the use of file I/O or complex graphical interface programming.

We present the NetLogo-Mathematica environment in the context of several modeling tasks. First, we will discuss the validation in a suite of statistical mechanics models with traditional analytical descriptions. Second, we will show how the interactive visualization features of *Mathematica* 6 can be used to replay model dynamics in an agent-based model of cultural dissemination. Finally, we will present effective design patterns for representing simulation results, and show how they may be used to perform an exploratory analysis of the parameter space of a forest fire model.

COMPARING AGENT-BASED MODELS WITH ANALYTICAL MODELS

The NetLogo-Mathematica kit was first used to solve the following problem: are the agent interaction rules in GasLab suite of NetLogo models (Wilensky, 1997a) sufficient to generate velocity distributions found via traditional analytical treatments of ideal gases? In this example, the model's initial conditions are set using `NLCommand[]`. We define the function `Resample[]` to execute the NetLogo model for 50 "ticks," and return a list of speeds back to *Mathematica*

```
Resample[]:= Module[{},
 NLCommand["repeat 50 [go]"];
 NLReport["[speed] of particles"]
];
```

To collect a sufficient number of moments of velocities, the distributions are resampled forty times using the list constructor, `Table[Resample[],{40}]`, which will generate a list of 40 elements, each element being the a consecutive resampling of the simulation.

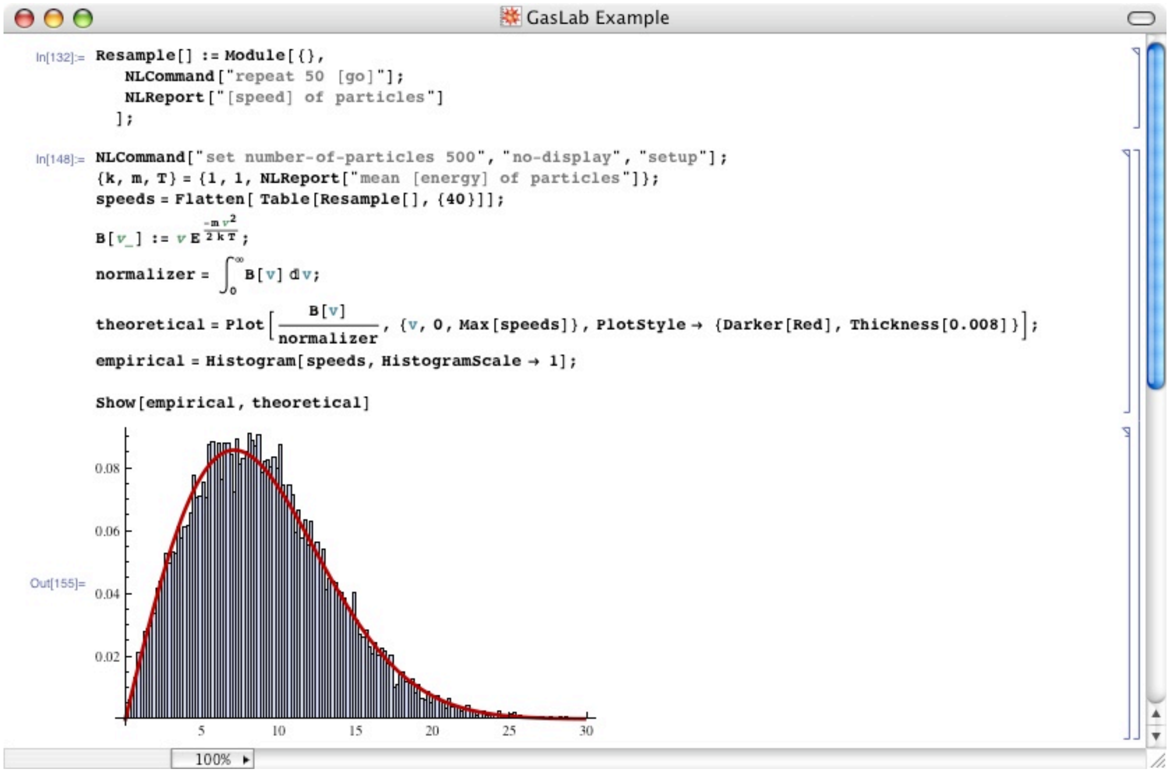


FIGURE 2 Comparing simulated results with analytical distributions

To compare the observed distribution with the analytic description of the speed distribution, we must find the mean energy of the system, which can be sampled directly from NetLogo: `NLReport["mean [energy] of particles"]`. With this data, we compare the observed distribution with the Maxwell-Boltzmann distribution. The two distributions are in close agreement with one another, as illustrated in Figure 2. This first example is a fairly straightforward example of data collection and visualization. We now turn to a more complex example involving multi-dimensional, time-varying data.

VISUALIZATION AND INTERFACE CONSTRUCTION

Programmable agent-based modeling environments like NetLogo allow developers to rapidly construct realtime visualizations of their model. However, an ABM environment typically supports a single modality of visual representation viewed forwards in time. This can make the analysis of complex systems, whose components evolve in a parallel fashion at multiple levels, quite difficult. The NetLogo-Mathematica environment provides a convenient way to store the complete simulation "history" in memory, and rapidly prototype interactive, multi-modal visualizations for understanding this data.

The following example shows how `NLGetPatches[]`, the NetLogo-Mathematica function for retrieving patch-based data, can be combined with interactive visualization procedures in the analysis of patch-based models. The time-varying patch data will be represented in several ways over time using `Manipulate[]`. In this example, we use a modified version of a NetLogo implementation (Centola, 2007) of Axelrod's model of cultural dissemination (Axelrod, 1997). In NetLogo, each patch agent owns a variable called `culturalFeatures`, which is an array of *k* features. At every time step, we execute the model and retrieve the matrix of patches, with each entry of the matrix being a list of cultural features, using `NLGetPatches["culturalFeatures"]`.

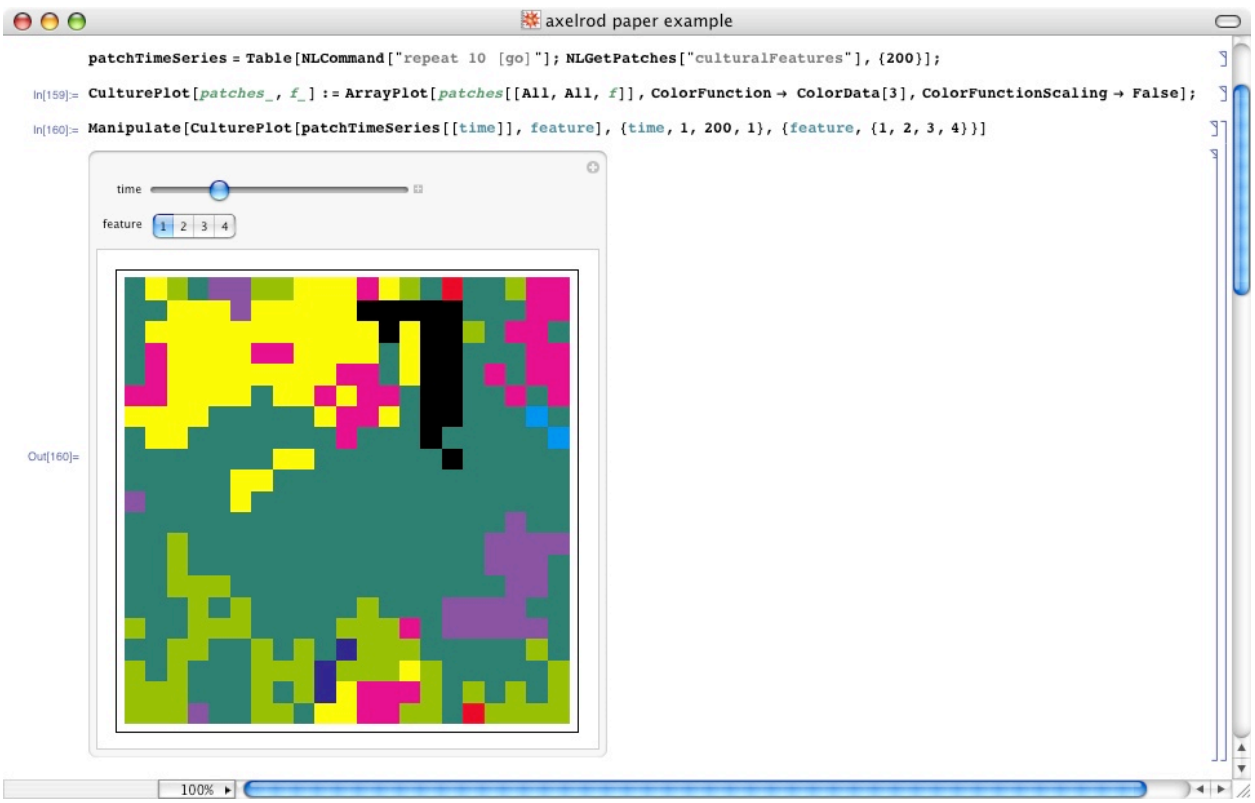


FIGURE 3 An interactive interface for exploring model dynamics

The interface in Figure 3 is generated by a single call to `Manipulate[]`:

```
Manipulate[CulturePlot[patchTimeSeries[[time]], feature],
 {time, 1, 200,1}, {feature, {1, 2, 3, 4}}]
```

The code specifies an interface which lets the user to view any of the four features at time steps 1 through 200, which may be animated both forwards and backwards. Dynamic visualization constructs such as `Manipulate[]` are just one example of a host of other high-level tools for constructing interfaces. These functions can also be used to display the progress of a parameter-space exploration in realtime, or compare aggregate "between realization" visualizations with individual simulation time-series. Such interfaces are particularly useful for collaborative analysis of models, since they allow a team of scientists, including those with less programming experience, to readily find patterns and test hypotheses in a model.

DESIGN PATTERNS FOR EXPLORATORY ANALYSIS OF AGENT-BASED MODELS

The traditional cycle of rigorous analysis of NetLogo models commonly involves writing software to specify a region of parameter space to explore in an automated fashion, either through the use of shell scripts, or specialized tools such as BehaviorSpace (Wilensky, 2003). Users must specify ahead of time the ranges and increments of parameters they would like to vary, and how many times each model run is repeated. Other structured data, such as lists or graphs can be difficult to format and read in by most tools for analysis, so most data written to disk is in the form of pre-aggregated scalar data. In this section we will propose a method for effectively executing and storing structured simulation data on a call-by-need basis. This approach is similar to memoization in dynamic programming, where "subproblems" (measures on a parameterization of a model) are stored in memory to speed up the execution of larger problems, such as finding critical points in a model's behavior or developing visualizations involving potentially thousands of runs.

A model realization can be thought of as a collection of data representing the execution of a deterministic program, or model. Agent-based models typically exhibit some degree of stochasticity. That is, the execution of the model involves psuedorandom processes, which may result in models that have identical initial conditions but produce a variety of possible outcomes. Thus, it is often important that there is a way to encode multiple realizations of the same parameterization of the model, but with different random seeds. With this method, any particular realization can be reproduced, given that the random seed is properly initialized and stored. These model realizations can be parameterized by their configuration settings and a labeling of their realization. A realization can be represented in *Mathematica* using some form, such as:

```
Realization[{p1, ..., pn}, repetitionNumber] = <model data>
```

In *Mathematica*, we will take advantage of the fact that the system can "remember" its value using the idiom:

```
f[x_] := f[x] = function to be evaluated at x
```

Each time the function `f[]` is evaluated at some *x*, its value is calculated and stored as part of the definition of the function. This is a convenient mechanism for storing the results of often computationally intensive realizations of models for later use. Below the typical structure of a NetLogo-Mathematica Realization object is specified:

```
Realization[{var1_, ..., varN_}, repetition_] :=
  Realization[{var1, ..., varN}, repetition] =
   Module[{intermediate data structures},
    (a) setup model using parameters;
    (b) execute model and store intermediate results;
    (c) return result structure
   ];
```

(a) NetLogo variables are initialized according to the model's parameters using NLCommand[]. Side effects of the initialization, such as NetLogo-generated random seeds, or initial placement of agents may be recorded in intermediate data structures

(b) The main NetLogo loop is executed, and agent variables or aggregate measures are recorded to intermediate data structures. At this point, we may process NetLogo data using *Mathematica* and insert new values into agents. This is typical for models in which agents utilize *Mathematica* functionality to carry out their rules.

(c) The intermediate structures are combined into a single expression representing the simulation.

In addition, users may define several functions that operate on the Realization data. These operators come in three common varieties:

- •Directly accessing an element of the resultant expression, such as a time-series array of some measure, or the distribution of agents' variables
- •Aggregating agent data, such as calculating the Gini index of the stored population or the clustering coefficient of a network
- •Visualization, such as plotting the time dynamics or network structure of a realization

Together, these functions and structures can provide a flexible framework for dealing with modeling tasks ranging from exploratory analysis, sensitivity analysis (Miller, 1998), to validation and docking (Axtell et. al., 1996, Wilensky & Rand, 2007).

EXPLORATORY ANALYSIS WITH REALIZATION OBJECTS

Here we will consider an instance of this Realization object prototype. This function executes the NetLogo forest fire model (Wilensky, 1997b) with a particular density and reports back the fraction of trees burned.

```
PctBurned[density_, rep_] := PctBurned[density, rep] = Module[{},
    NLCommand["set density ", density, "setup"];
    NLCommand["while [any? turtles] [go]"];
    NLReport["(burned-trees / initial-trees)"]
    ];
```

We may attempt to find the phase transition by plotting the result of this function with the entire range of densities, from zero percent to one hundred percent in increments of ten:

```
ListPlot[Table[PctBurned[density,1],{density,0,100,10}]]
```

Finding that a phase transition occurs approximately between forty and eighty percent density, we can execute the model over this range in increments of five percent, and observe its variance over ten additional repetitions using a box and whisker plot:

```
BoxWhiskeryPlot[Transpose[Table[Table[PctBurned[density,rep],{rep,10}],
 {density,30,60,5}]]]
```

Finally, we can plot the transition averaged over twenty runs at a higher resolution in increments of one:

```
ListPlot[Table[Mean[Table[PctBurned[density,rep],{rep,20}]],
 {density,30,60,1}]]
```

CONCLUSION

The environment and techniques presented in this paper can provide researchers with a rich environment in which they can rigorously debug, analyze, and make inferences from agent-based models. It provides an integrated workflow which enables users to focus on experimentation rather than the implementation. We hope that the methods proposed here can be of use to the agent-based modeling community and promote a more intimate understanding of phenomena observed in our models and a more robust treatment of our results. We have used these tools in several of our research projects at the CCL. In the context of an NSF-funded research project on modeling educational policy, we have profitably applied the NetLogo-Mathematica interface tools to explore a large-scale model of school choice calibrated with empirical data. The integrated environment has enabled the iterative construction of the model, including the model calibration, analysis of runs, and even model documentation. In addition, the environment has aided in the collaboration with members outside of our immediate research group by enabling us to rapidly examine new hypotheses and analyze the data in multiple ways. In this respect, the NetLogo-Mathematica integrated environment provides a powerful addition to the model builder's toolkit.

ACKNOWLEDGMENTS

The preparation of this paper was supported by the National Science Foundation DHB grant #0624318. We would like to thank Spiro Maroulis, Bill Rand, Seth Tisue, and Lada Adamic for their feedback on this work.

REFERENCES

Axelrod, R. The Dissemination of Culture: A Model with Local Convergence and Global Polarization The Journal of Conflict Resolution, Vol. 41, No. 2 (Apr., 1997), pp. 203-226

Axtell, R., Axelrod, R, Epstein, J. M, & Cohen, M. D. (1996), "Aligning Simulation Models: A Case of Study and Results", Computational Mathematical Organization Theory, 1(2), pp. 123-141

Macal, C.M. & Howe, T.R. (2005). Linking Repast and Computational Mathematics Systems: Mathematica and MATLAB. Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms, Gleacher Center. Chicago, IL, USA. 13-15 October 2005. Argonne National Laboratory and the University of Chicago. Pp. 5-23.

Miller, J. H. (1998) Active Nonlinear Tests (ANTs) of Complex Simulation Models Management Science, Vol. 44, No. 6 pp. 820-830

Tisue, S., & Wilensky, U. (2004). NetLogo: Design and implementation of a multi-agent modeling environment. Paper presented at the Agent 2004 conference, Chicago, IL, October 2004.

Wilensky, U. (2001). Modeling nature's emergent patterns with multi-agent languages. Paper presented at the Eurologo 2001 conference, Linz, Austria.

Wilensky, U. & Rand, W. (2007). Making Models Match: Replicating an Agent-Based Model" Journal of Artificial Societies and Social Simulation, 2007, 10(4). http://jasss.soc.surrey.ac.uk/10/4/2.html .

Wolfram, S., (2003). The Mathematica Book: 5th Edition, Wolfram Media, Champaign, IL.

Models and Software

Centola, D. (2007). Axelrod Culture Model. http://hsd.soc.cornell.edu/curricular/Axelrod/AxelrodModel.html. Networks and Social Dynamics Group, Cornell University, Ithaca, NY

Wilensky, U. (1997a). NetLogo GasLab Gasinabox model.http://ccl.northwestern.edu/netlogo/models/GasLabgasinabox. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

Wilensky, U. (1997b). NetLogo Fire model. http://ccl.northwestern.edu/netlogo/models/Fire. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

Wilensky, U. (1999). NetLogo [Computer software] (Version 4.0). Evanston, IL: Center for Connected Learning and Computer-Based Modeling. http://ccl.northwestern.edu/netlogo.

Repast, 2007, [Computer software] Repast home page; available at http://repast.sourceforge.net/

Mathematica, 2007, [Computer software] Mathematica documentation (Version 6.0); Champaign, IL: Wolfram Research Inc. http://reference.wolfram.com/mathematica/.

Wilensky, U. (1999a). NetLogo [Computer software] (Version 4.0). Evanston, IL: Center for Connected Learning and Computer-Based Modeling. http://ccl.northwestern.edu/netlogo.

Wilensky, U. (2003). Behavior Space [Computer Software] (Version 3.0). Evanston, IL: Center for Connected Learning and Computer Based Modeling, Northwestern University. http://ccl.northwestern.edu/netlogo/behaviorspace.

EXAMINING GROUP BEHAVIOR AND COLLABORATION USING ABM AND ROBOTS

PAULO BLIKSTEIN*, Northwestern University, Evanston, IL
WILLIAM RAND, Northwestern University, Evanston, IL
URI WILENSKY, Northwestern University, Evanston, IL

ABSTRACT

Agent-based modeling has been extensively used by scientists to study complex systems. *Participatory simulations* are similar to agent-based models except that humans play the role of the virtual agents. The *Bifocal modeling* approach uses sensors to gather data about the real-world phenomena being modeled and uses that information to affect the model. In this work, we are interested in automatically extracting, analyzing and modeling group behaviors in problem solving. Combining these three systems into one unified platform would be useful for those purposes, since it would facilitate a synthesis of their main affordances: understanding the role of locality, mapping human action to emergent behaviors, and controlling embedded physical objects in noisy environments while receiving sensory feedback. We will demonstrate a technological platform based on the NetLogo/HubNet architecture that supports simulated agents, participatory agents and physical agents. We place this platform within a more general framework that we call Human, Embedded and Virtual agents in Mediation (HEV-M). We have run several studies using an instantiation of this platform that consists of a robot-car with four users who navigate a maze. We believe that this tool has potential for three main reasons (1) it facilitates logging of participant's actions, so as to identify patterns, (2) it offers researchers in the field of computer-supported collaborative learning an easy-to-use tool to design engaging collaborative learning activities and, (3) it foregrounds the role of individual actions within the accomplishment of a collective goal, highlighting the connections between simple individual actions and the resultant macroscopic behaviors of the system.

Keywords: group behavior, agent-based modeling, collaboration, robots

INTRODUCTION

Agent-based modeling has been used by scientists to study phenomena such as the interactions of species in an ecosystem, the collisions of molecules in a chemical reaction, and the food-gathering behavior of insects (Bonabeau, 1999; Troisi, Wong & Ratner, 2005; Wilensky & Reisman, 2006). Typical of agent-based models is that the aggregate patterns or behaviors at the macro level are not premeditated or directly actuated by any of the micro-elements. Participatory simulations are similar to multi-agent simulations except that humans play the role of the virtual agents (Wilensky & Stroup, 2002). As yet another extension to ABM methods, Blikstein & Wilensky (2006) have been exploring the use of physical devices in agent-based modeling, using sensors to gather data about the real-world phenomena under scrutiny (*bifocal modeling*).

* *Corresponding author address*: Paulo Blikstein, Center for Connected Learning and Computer-Based Modeling, 222 Annenberg Hall, 2120 Campus Dr., Evanston, IL 60208; e-mail: paulo@northwestern.edu.

The three aforementioned areas (agent-based modeling, participatory simulations, and bifocal modeling) are concerned with the creation, manipulation, and development of agents in one form or another. In this work, we are particularly interested in automatically extracting, analyzing and modeling group behaviors and collective strategies for problem solving. Combining these three systems into one unified platform would be useful for those purposes, since it would facilitate a synthesis of their main affordances: understanding of the role of locality, mapping human action to emergent collective behaviors, and controlling embedded physical objects in noisy environments while receiving sensory feedback. We will demonstrate a technological platform based on the NetLogo/HubNet architecture (Wilensky, 1999; Wilensky and Stroup, 1999) that supports simulated agents, participatory agents and physical agents (Rand, Blikstein, & Wilensky, 2006). Within this platform, designers can create participatory simulations in which each participant controls one micro-element within a physical system (a car, a mini-factory, etc.), while at the same time interacting with virtual agents. We place this technological platform within a more general framework that we call Human, Embedded and Virtual agents in Mediation (HEV-M). This framework facilitates general discussion about the components of the overall system and their interaction across particular technologies and instantiations.

We have run four studies using an instantiation that consists of a robot-car with four motors, each connected to a robotics interface, the GoGo Board (Sipitakiat, Blikstein & Cavallo, 2004), which communicates with the server. Each user is assigned a motor to control, and turning the car is achieved by activating, deactivating, or reversing the correct wheels. Participants were given the task of moving the robot from a start area to a goal area while avoiding obstacles along the way.

Initial results were intriguing. In our first studies, with university professors and researchers (Blikstein, Rand & Wilensky., 2006; Rand, Blikstein & Wilensky, 2006), before the start of the activity, participants were confident that they could easily accomplish the task. However, as soon as the first turn was necessary, participants started to report increasing frustration[1] with their ability to solve the problem, and we observed the emergence of strategies for optimizing the process, such as delegating leadership to one participant, or formation of two groups acting fairly independently. Also, at the beginning, many participants seemed unaware that an error from *any* of the participants could ruin the group's goal, no matter how well other participants were doing. However, in the present study, with computer science students as subjects, resulted in a diverse set of strategies for managing the task, as we will explain in this paper.

We present the current study as one example of how collaboration with embedded objects can be observed, but the potential of this framework and technology goes beyond this instance. As an example, almost any agent-based model could be recreated using physical agents and human agents interacting with those agents. For instance, traffic simulations in which participants controlled remote control cars, could offer insight into human behavior in traffic systems. The virtual agents in the current study are fairly passive, serving as conduits from the participants to the robot. However, these agents could be given a greater level of interaction, allowing them to interpret and respond to data from both the participants and the robot, and make

[1] It should be noted that the participants found this frustration humorous, since they were amazed that they could not solve such a simple problem.

their own autonomous decisions. This would add another level of complexity to the overall system.

We believe that the framework instantiation presented in this paper has significant potential for three main reasons (1) it facilitates logging of participant's actions, so as to identify patterns and match them to observations, (2) it offers researchers in the field of computer-supported collaborative learning an easy-to-use tool to design engaging collaborative learning activities and, (3) it foregrounds the role of individual actions within the accomplishment of a collective goal, highlighting the connections between simple individual actions and the resultant macroscopic behaviors of the system.

THE HEV-M FRAMEWORK

On a certain abstraction level, human, robotic (also called embedded) and virtual agents can be viewed as equivalent: all of these agents have properties (i.e., descriptions of themselves, and knowledge about the world) and methods (i.e., actions that they can take to achieve goals). In all three cases, the agents, regardless of being human, embedded or virtual, will examine the world around them and their own internal state and decide what action to take on the basis of this input.

Each of these systems, virtual, robotic, and human, present their own challenges. In the case of human agents, the logic that connects the input to the output may not be well known by outside observers, and thus the actions taken may be quite unpredictable. But confusion about the relationship between inputs and outputs is not limited to the human case. Robots can have noisy sensors that affect their perception of the world, and their actuators, also subject to a noisy environment, may not always work perfectly. In addition, there are many challenges to designing virtual agents correctly. Often low-level rules do not result in anticipated emergent patterns. Nonetheless, there are many reasons to motivate the combination of these systems into one integrated platform.

Robotic agents and virtual agents working within a shared model can be complementary. Robotic agents could use virtual agents to plan out routes and to simulate their movements ahead of time, which would assist in the development of some robotic agents, like planetary rovers. However, this is not a simple task. Robotic agents operate within the physical world (which often interferes with the task) and they have noisy sensors and fallible actuators. As mentioned, the integration of virtual systems with robotic systems can present researchers with many difficulties. How does one model the noisiness and inefficiency of the physical world within a virtual system, so that virtual and robotic agents can remain in step with each other? How should virtual agents interpret data from a robotic agent?

In much the same way that robotic agents are different from virtual agents, so are human agents different from virtual agents. The integration of human agents into a unified system also presents many of the same issues that challenge the integration of robotic agents, since they also have noisy sensors and inefficient actuators. Moreover, they present additional problems from a virtual agents' standpoint – human agents can adapt to their surroundings in new and surprising ways, which means that they are less predictable, and can be deliberately obstinate or malicious, attempting to confuse and take advantage of virtual agents. Notwithstanding these challenges, the integration of human and virtual agents within a shared system has a lot of potential. For instance, a model developer can have humans play the role of agents, subsequently capturing and embedding the decisions made by humans into virtual agents, enabling a richer and more elaborate examination of the behaviors employed by the humans (for more information on work on virtual and humans agents using the HubNet platform, see Abrahamson & Wilensky, 2004; Berland & Wilensky, 2006; Wilensky & Stroup, 2002). Alternatively, human agents could work together with virtual agents to accomplish some mutual goal. For instance, in a war simulation, humans could place emphasis on different targets while allowing the virtual agents to take care of the low-level planning. However, all of this requires the development of new protocols – for example, how does one automatically capture human decisions and embed them in agent-based rules? How can human agents express new beliefs, desires and intentions to a virtual agent?

We have been discussing these relationships between human and virtual agents, and robotic and virtual agents as separate entities, but these relationships can also be combined within a unified framework. In this paper, we will explore the combination of all these agents within one integrated platform (Blikstein, Rand & Wilensky., 2006; Rand, Blikstein & Wilensky., 2006). Our unified conceptual framework is the HEV-M framework, which stands for the integration of **H**uman agents, **E**mbedded sensory-enabled robotic agents, and autonomous **V**irtual agents, which communicate via a central **M**ediator (see Figure 3). The three different agent groups may have different goals and even different tasks. The mediator takes messages from any of the three groups of agents, transforms the messages, and relays the information to the other groups within a well-established protocol.

We have previously speculated (Rand, Blikstein & Wilensky, 2006) how this framework might be useful through the use of three hypothetical examples: Widget Factory, Planetary Rover, and Demon Soccer. In Widget Factory humans and virtual agents control simple machines that create parts of widgets. This environment can show, for example, that minor errors in the creation of the parts can dramatically alter the resultant outcome. In Planetary Rover humans cooperate with virtual agents to control a robotic agent. The virtual agents utilize sensory data about their environment to make independent decisions. This environment can enable the exploration of collaborative human-robot protocols. In Demon Soccer, human agents interact with virtual agents to control a soccer ball. The human agents play on opposing teams and attempt to steer the soccer ball in to their opponent's goal. Four different agents control the four wheels. Two of the agents are humans, and two of the agents are demon agents that either malignantly or randomly alter the speed and direction of the wheels. This environment enables the exploration of mediation between hostile agents, and could offer insight into how humans adapt to new and challenging situations.

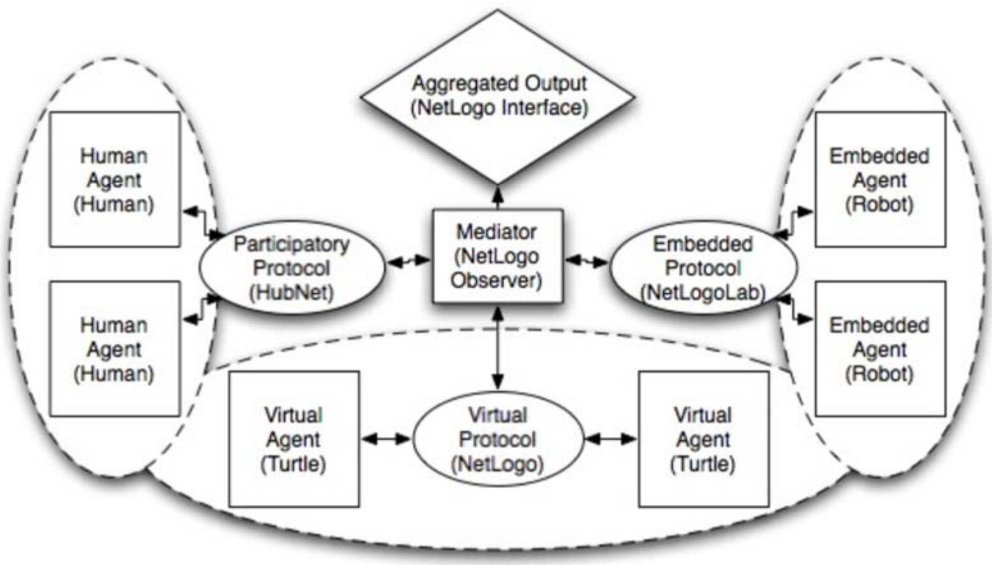


Figure 1: **HEV-M** Framework**.**

TECHNOLOGICAL PLATFORM

In this paper, we describe one technological platform that implements the components of the HEV-M framework. This platform is based on the NetLogo/HubNet modeling environment, and on the GoGo Board, an open-source piece of hardware for interfacing the computer with sensors and actuators. The system has three components:

1. **Robot-car:** the car has four motors, each connected to a wheel and controlled independently. The wheels cannot be steered, thus turning the car is achieved by selectively engaging different wheels in different directions. For example, a slow turn to the left can be accomplished by turning on both of the right wheels, and a faster turn can be accomplished by also turning the left wheels on, but in reverse. The motors have three power levels (high, medium and low), and are connected by long wires to the robotics interface. The interface, in turn, is connected to the server.

2. **Client computers:** each of the four client laptops have a simple interface for wheel control, enabling the user to turn his/her own wheel on and off, set the power level, and toggle the direction of rotation of the wheel.

3. **Server:** the server receives information from the four client computers and controls the robot-car accordingly. It also keeps a log of all the actions performed by the users.

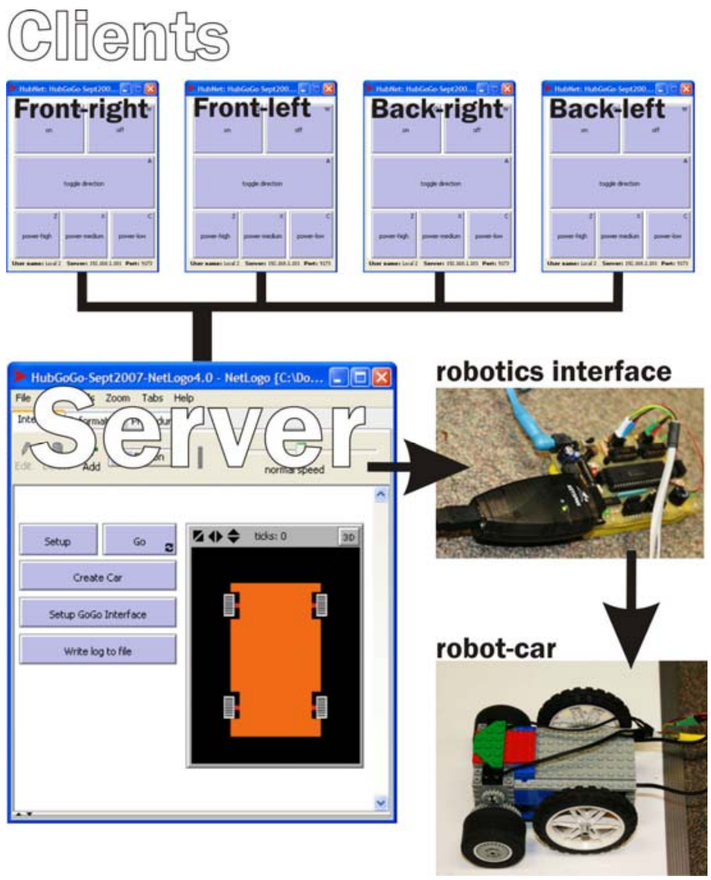


Figure 2 Diagram of the system, with its three components: the client computers, the robot-car, and the server.

EXPERIMENTS

This framework for agent integration is not just hypothetical– we have implemented it in several projects. (Blikstein, Rand & Wilensky, 2006; Rand, Blikstein & Wilensky, 2006). These preliminary prototypes had human and virtual agents working together to guide a robotic agent through a maze.

To extend those preliminary studies, we defined a methodological framework to conduct experiments. First, we standardized the size of the track and generated three fixed mazes. We also implemented a logging feature to capture keystrokes and mouse clicks from the participants. Finally, we defined a sequence of four activities to propose to participants:

Act. 1. **Maze with one obstacle, with communication** – we tell participants that they can talk to each other.

Act. 2. **Maze with two obstacles, without communication** – we tell participants that they should conduct the activity in silence, although they can observe at each other.

Act. 3. **Maze with three obstacles, with leader.** We randomly pick one of the participants and ask them to lead the other ones.

Act. 4. **Maze with three obstacles, randomized.** All wheels are randomized at the beginning, so users don't know beforehand which wheel they control. They have to figure it out during the activity.

Two main sources of data were used: video data and log files of students' interactions. For the video data collection, two cameras were utilized, one fixed, facing the participants, and one mobile, mainly facing the whiteboard and the robot-car. The log files recorded all of participants' interactions with the system.

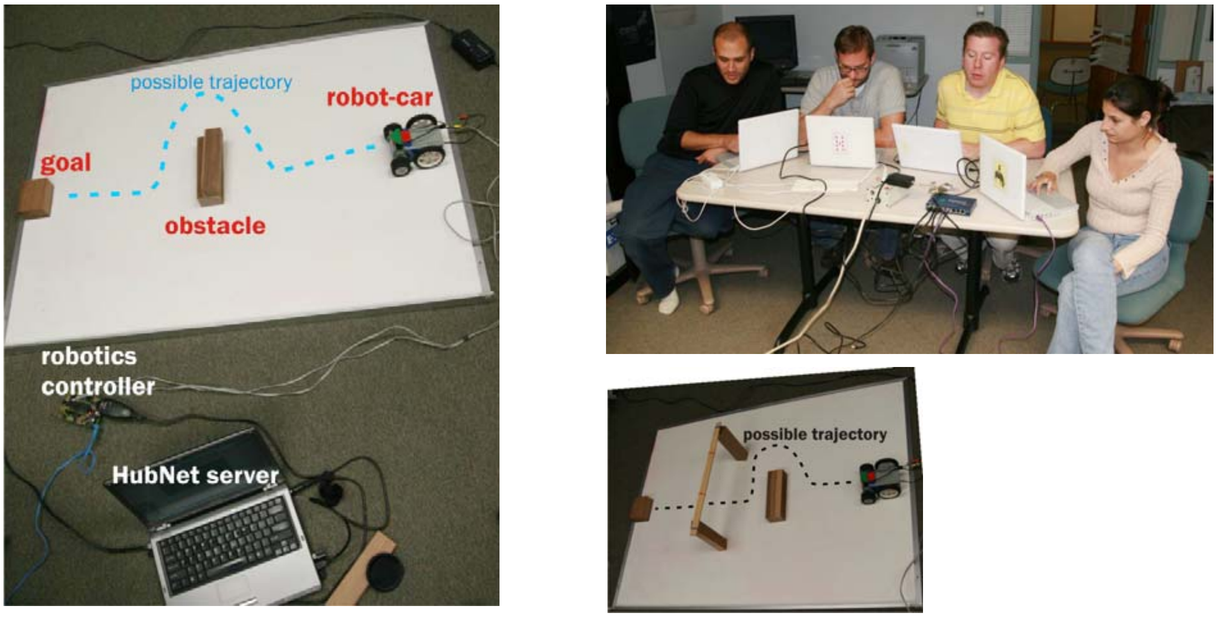


Figure 3 Clockwise from the left: The experimental setup for Act. 1, the four participants, and the experimental setup for Act. 2.

DATA ANALYSIS AND DISCUSSION

In previous work (Rand, Blikstein & Wilensky, 2006), we reported on a group of four university professors and researchers that had great difficulties in successfully completing the maze. We observed that inter-subject communications were confusing and out-of-sync with the required speed of action, and users could not establish clear leadership. The group of professors apparently underestimated the difficulty of the task and over-engineered their own strategies, resulting in poor performance. For that study, however, the logging mechanism was not yet in place, so our understanding of participants' reaction was partial, based on their own utterances and our observations of the robot-car. For the following study, with the logging mechanism in place, a group of four computer science students was selected. We began the study with the hypothesis that, being young students, they would be more spontaneous and communicate extensively; being experts in computer science, they would try to engineer elaborate strategies to control the car. Both of these hypotheses were proved wrong, and other results became apparent from our data analysis, which we will explain below

Fading inter-personal communication patterns

Participants started out communicating extensively during the first activity. The second activity was supposed to be in silence, but even after communication was permitted again, on the third activity, participants did not resume verbal communication: they were paying attention exclusively to the car. Below is the transcript of the dialogue during the first activity, showing that participants were able to devise a successful strategy and orally coordinate their activities:

> **John:** I have a plan: Jim and I don't do anything, you do it all. You guys are the front.
> [after a few second, the car stalls]
> **Marcia:** Uh Oh. More power.
> **John:** Do you think you need us?
> **John:** We need some back…
> **Marcia:** Nice
> **John:** You guys got it, you don't need us.
> **Marcia:** Great success, guys.

However, from the third activity on, there was barely any verbal communication. This was in contradistinction to our initial hypotheses. Somehow the participants developed a personal heuristic as to how to control their wheel which did not require communication. One explanation is that they "read" other participants' states and intentions *through the state of the car*, with no need of explicitly asking questions. As we will show below, this explanation is supported by both the post-interviews that we conducted with the participants and an examination of the log files.

Diversity of personal strategies

Participants' post interviews further corroborate the hypotheses of decreased oral communication, since their self-reported strategies and heuristics did not include talking or asking question of the other participants:

> **Edward:** I was not paying attention to anyone; *I was paying attention to the car*. I was just paying attention to my wheel. If I did something and it went bad, because another person did something else, I would just go back to my previous state.
>
> **Jim:** *I did very little.* I figured that if everyone was hitting buttons and moving forward, the car wouldn't go anywhere, so I waited for opportunities in which I was pretty sure I would make a difference.
>
> **John:** I was back-left, when I was on, the car would go to the right. When my thing is going forward, the car would go right. If my thing is going backwards, the car is going left.
> **Interviewer:** But if the other guy is doing the opposite…
> **John:** Then the car wouldn't go anywhere. [long pause] That's ok.

For John, looking at the car (which was exhibiting behavior that resulted from aggregating each group member's directives) and reacting to it on-the-fly was more efficient than explicitly discussing strategies (which we observed in our previous study with the university professors and researchers). Despite the car's behavior being a collective construction, John was reacting to the resulting emergent behavior of the group, and not discussing every single individual action. Jim had a very different strategy: he realized that, by simply doing nothing, the car would probably achieve the same goal, since there was redundancy in the system – by staying out, he thought he could help the group achieve the goal faster. Edward, conversely, was very active, and devised a strategy of trial-and-error – if his move resulted in undesired behavior, he would just undo the movement, without negotiating every move. In all three of these cases the individual decision criteria is focused on the aggregate behavior of the car, and completely excludes any involvement of the other participants.

Use of the different motor control commands

Additionally, the log files show that, notwithstanding the symmetry of the car, each user had a different approach to their use of the six different commands. One commonality between participants was observed: power-high, medium and low were used very infrequently. Two users (back-right and front-right) realized that just leaving the motor on and using ‘reverse-direction’ during the activity was the most effective strategy. As the log files show, as time went on, these participants employed this strategy with increasing frequency. John, who was controlling the ‘back-right’ wheel, used ‘reverse-direction’, or ‘rd’, almost exclusively toward the end of the study (Activity 3). Comparing the log files and the verbal data, we observed that this learning process took place tacitly, without any oral communication between users. However, John was conscious that he had learned an important technique, since when he was asked to lead the group, on activity 3, he asked everyone to “turn on and just use reverse-direction”. Another surprising observation was that, even after John asked all users to exclusively use ‘rd’, only those users who had had a significant increase in ‘rd’ from activity 1 to activity 2 followed his advice. This can be seen in the ‘rd’ lines on the plots in Figure 4. Compare the ‘rd’ lines of ‘back-right’ [John himself] and ‘front-right’, as opposed to ‘back-left’ and ‘front-left.’ One explanation is that two of the participants employed their own personal theory on how to control the car and were resistant to follow the directions of the leader. This hypothesis is further supported by the aforementioned transcriptions of users’ self-reported techniques for car control.

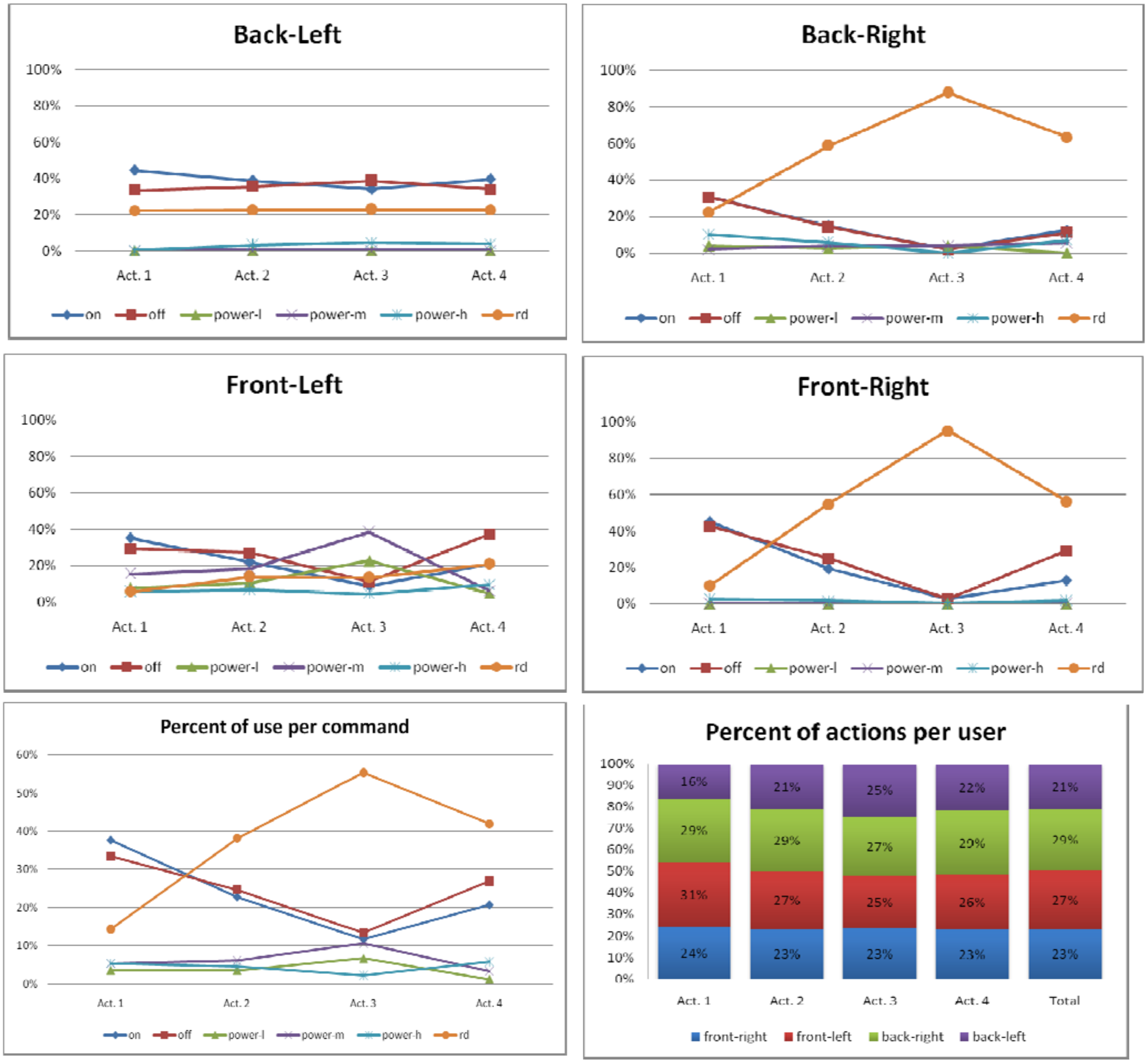


Figure 4 The use of each command over the four activities for each user (first four plots), the overall percent of use per command for all users (bottom left, note the clear increase in the use of reverse-direction), and the percent of actions per user (bottom right), showing an almost uniform distribution, with the exception of user 'back-left'.

CONCLUSION

The HEV-M framework and the implementation described in this paper proved to be a useful tool in exploring the interactions and interoperability among human, virtual and physical agents. We developed data collection tools and techniques that reveal tacit individual and collective strategies for problem solving and communication. The approach of pairing verbal data and log files described in this paper could enable other researchers to unveil unexpected communication and behavior patterns that would otherwise go unnoticed. For example, one behavioral pattern that we observed was that users' final strategy resulted in a focus on the car's actions and movements, instead of observing or communicating with the participants – despite being in the same room. Surprisingly, a simple robot-car ended up mediating interpersonal communication more effectively than oral discourse. Seeing as how the humans involved did not actually communicate and seemed to settle on final strategies quickly, it might have been possible to replace them with virtual agents able to observe the robot-car and make decisions similar to the humans. As we have observed there would need to be different types of virtual agents to represent the different human behavioral styles, but that is a simple task. These results suggest that the nature of the agent controlling the device – human or virtual – could be of less importance than is commonly thought. If this result is confirmed by further research, this could be an important contribution to the study of human-computer interaction within the field of agent-based modeling.

AKNOWLEDGEMENTS

The preparation of this paper was supported by the National Science Foundation (Information & Intelligent Systems Division grant #0713619, "Advancing the Science of Agent-Based Modeling Through Frameworks, Tools, and Pedagogies", PI: Uri Wilensky).

REFERENCES

Abrahamson, D., & Wilensky, U. (2004). *SAMPLER: Collaborative interactive computer-based statistics learning environment.* Paper presented at the 10th International Congress on Mathematical Education, Copenhagen, July 4 - 11, 2004.

Berland, M., & Wilensky, U. (2006). Constructionist collaborative engineering: Results from an implementation of PVBOT. Paper presented at the annual meeting of the American Educational Research Association, San Francisco, CA.

Blikstein, P., & Wilensky, U. (2006). The Missing Link: A Case Study of Sensing-and-Modeling Toolkits for Constructionist Scientific Investigation. In *Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies (ICALT 2006),* Kerkrade, The Netherlands, 980-982.

Blikstein, P., Rand, W., & Wilensky, U. (2006). Participatory, Embodied, Multi-Agent Simulation. Paper presented at the AAMAS-06 Conference, 2006.

Bonabeau, E., Dorigo, M., Théraulaz, G. (1999). *Swarm intelligence: From natural to artificial systems.* London: Oxford University Press.

Rand, W., Blikstein, P., & Wilensky, U. (2006). *Widgets, Planets, and Demons: the Case for the Integration of Human, Embedded, and Virtual Agents via Mediation*. Paper presented at the Swarmfest 2006.

Sipitakiat, A., Blikstein, P., & Cavallo, D. P. (2004). *GoGo Board: Augmenting Programmable Bricks for Economically Challenged Audiences.* Proceedings of the International Conference of the Learning Sciences, Los Angeles, USA.

Troisi, A., Wong, V., & Ratner, M. (2005). An agent-based approach for modeling molecular self-organization. *Proceedings of the National Academy of Sciences, 102*(2), 255-260.

Wilensky, U. (1999). NetLogo. Evanston, IL: Center for Connected Learning and Computer-Based Modeling. http://ccl.northwestern.edu/netlogo.

Wilensky, U., & Blikstein, P. (2005). NetLogoLab curriculum. Evanston, IL: Center for Connected Learning and Computer Based Modeling, Northwestern University.

Wilensky, U., & Reisman, K. (2006). Thinking like a wolf, a sheep or a firefly: Learning biology through constructing and testing computational theories -- an embodied modeling approach. *Cognition & Instruction, 24*(2), 171-209.

Wilensky, U., & Stroup, W. (2002). *Participatory Simulations: Envisioning the networked classroom as a way to support systems learning for all.* Paper presented at the Presented at the Annual meeting of the American Research Education Association, New Orleans, LA, April 2002.

Repast Toolkit Developments

VISUAL AGENT-BASED MODEL DEVELOPMENT WITH REPAST SIMPHONY

M.J. NORTH,* Argonne National Laboratory, Argonne, IL,
and The University of Chicago, Chicago, IL
ERIC TATARA, Argonne National Laboratory, Argonne, IL
N.T. COLLIER, Argonne National Laboratory, Argonne, IL,
and PantaRei Corp., Cambridge, MA
J. OZIK, Argonne National Laboratory, Argonne, IL

ABSTRACT

Repast is a widely used, free, and open-source agent-based modeling and simulation toolkit. Three Repast platforms are currently available, each of which has the same core features but a different environment for these features. Repast Simphony (Repast S) extends the Repast portfolio by offering a new approach to simulation development and execution. This paper presents a model of physical infrastructure network interdependency as an introductory tutorial and illustration of the visual modeling capabilities of Repast S.

Keywords: Agent-based modeling and simulation, Repast, toolkits, and development environments

INTRODUCTION

Repast (ROAD 2005; North et al. 2006) is a widely used, free, and open source agent-based modeling and simulation toolkit with three released platforms, namely Repast for Java, Repast for the Microsoft .NET framework, and Repast for Python Scripting. Repast Simphony (Repast S) extends the Repast portfolio by offering a new approach to simulation development and execution, including a set of advanced computing technologies for applications such as social simulation. North et al. (2005a and 2005b), Howe et al. (2006), and Parker et al. (2006) provide an overview of the Repast S runtime and development environments.

We use a model of networked physical infrastructure to demonstrate the visual design capabilities of the Repast S toolkit and as an introductory tutorial. While the example is not intended to model real phenomena, the model's complexity is high enough to illustrate how the user may develop multi-agent models.

It is important to note that Repast S and its related tools are still under development. This paper presents the most current information at the time it was written. However, changes may occur before the planned final release.

---

* *Corresponding author address*: Michael J. North, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439; email: north@anl.gov.

THE REPAST S MODEL IMPLEMENTATION BUSINESS PROCESS

As discussed in North et al. (2005a and 2005b), the Repast S model implementation business process is as follows:

- The modeler creates model pieces, as needed, in the form of plain old Java objects (POJOs), often using automated tools or scripting languages such as Groovy.

- The modeler uses declarative configuration settings to pass the model pieces and legacy software connections to the Repast S runtime system.

- The modeler uses the Repast S runtime system to declaratively tell Repast S how to instantiate and connect model components.

- Repast S automatically manages the model pieces based on (1) interactive user input and (2) declarative or imperative requests from the components themselves.

The POJO model components can represent anything but are most commonly used to represent the agents in the model. While the POJOs can be created by using any method, this paper discusses one powerful way to create POJOs for Repast S: the Repast S development environment. However, modelers can use any method—from hand coding to wrapping binary legacy models to connecting into enterprise information systems—to create the Repast S POJO model components.

Regardless of the source of the POJOs, the Repast S runtime system is used to configure and execute Repast S models. North et al. (2005b) detail the Repast S runtime system, the design of which includes:

- Point-and-click model configuration and operation;

- Integrated two-dimensional, three-dimensional, and other views;

- Automated connections to enterprise data sources; and

- Automated connections to powerful external programs for conducting statistical analysis and visualizing model results.

SIMPLE PHYSICAL INFRASTRUCTURE NETWORK MODEL

A model of interconnected physical infrastructure networks is presented as an introductory tutorial and illustration of the visual modeling capabilities of Repast S. The model consists of a natural gas transmission and DC electric power network (Tatara et al. 2007b). The natural gas transmission model consists of a network of interconnected links and nodes, where the nodes function as delivery, receipt, and/or pipeline termination points, and the links function as gas pipelines that transport natural gas between nodes. The DC electric network model considers a balance of demand and generation given the transmission topology. The nodes in the electric network represent generators and load points, while the links function as electrical transmission lines. The two networks are connected via links between the natural gas network and gas-fired electric power plants (generators) in the electrical network. The simple networks presented here will model propagation of pressure and power along the gas and electrical networks.

VISUAL PROJECT CREATION AND AGENT DESIGN

While previous versions of Repast required the user to set up and configure an appropriate integrated development environment (IDE), Repast S provides a preconfigured Eclipse-based IDE such that no *a priori* programming experience is required to build a model. Although the Repast IDE is aimed at novice developers, the full Repast S Java application programming interface (API) and advanced IDE configuration options are available at any time. Previously, Tatara et al. (2006) discussed using Repast S to create a model of wolf-sheep predation through the Java API.

After the Repast S IDE is started, the user may choose to continue working with an existing project or create a new project. The project creation wizard prompts the user for the type of project to create (Figure 1). The user is prompted for basic project information such as the project name (Figure 2). Additional project options are available to the advanced user, although these options may simply be left as the default.

When the Repast project is created in the workspace, a set of project components is visible in the package explorer, shown on the left side of Figure 3. These components include things such as directories for storing user data and the project source code. Also visible in Figure 3 is the Score editor, which specifies the hierarchical structure of the model contexts, agents, and projections. Model elements are represented graphically in a tree, and components may be added on a point-and-click basis. Once model elements have been placed in the Score editor, their properties may be edited in the Properties window shown at the bottom of Figure 3.

After the model Score has been completed, the user may start creating the agent objects. At this point, the advanced user may choose to create the agent classes using the Java API, while those users not familiar with Java may chose to use the Repast agent editor. The agent creation wizard is accessed via the package explorer and allows the user to create a number of Repast objects (Figure 4). When a new agent is created (Figure 5), the IDE view switches to the agent editor view as shown in Figure 6.

As discussed in detail in Ozik et al. (2007), the visual agent behavior editor, the new project wizard and the new agent wizard are modified forms of Alexander Greif's free and open source Flow4J-Eclipse components (Greif 2006) that have been adapted specifically for agent-based modeling. Greif (2006) has made the Flow4J-Eclipse system available under a BSD-style free and open source license. The Repast project team has built on Greif's contribution to create the above-mentioned Repast S components. From the Flow4J home page, Greif (2006) states:

> Flow4J is an Eclipse Plug-in for modeling process flows in a drag and drop manner. A process flow can contain process steps (I call them flowlets), which can be linked together [in]to a complex flow.

Ozik et al. (2007) provides the details on what Repast has both inherited and modified/adapted from Greif's (2006) Flow4J-Eclipse system.

Like the Flow4J-Eclipse visual editor, the Repast agent editor itself consists of an editable icon panel and a palette of behavior icons, which may be dragged into the edit panel and modified. As discussed in Ozik et al. (2007), the icons are analogous to blocks in a flowchart that may be connected in flexible ways to create the agent behavior logic. Figure 6 shows the creation of an agent property "pressure" for the GasNode agent class. The property parameters may be edited in the bottommost panel in Figure 6. The user is asked to specify a number of required elements such as the property name, data type, and initial values, while several optional data, such as a long description of the property, may also be defined.

Behaviors are defined by creating a behavior element in the workspace as shown in Figure 7. The behavior element may be either a scheduled behavior or one that is event-driven. The desired behavior for the nodes in the natural gas network is to react to changes in pressure upstream. Therefore, the behavior at the gas nodes will be event-driven and caused by changes in pressure from connected gas nodes.

The behavior block defines how and when the behavior occurs and not what actually happens next. A Task block is used to define the active part of the behavior to which it is associated (Figure 8). The Task block specifies what the agent does when the behavior is triggered. Continuing with the gas network node behavior, the task should adjust the node's pressure based on the pressure of the upstream node. Finally, the behavior logic is terminated with an End block as shown in Figure 8.

When the user saves the diagram, the Repast IDE automatically compiles the diagram into usable code that may be immediately loaded into the Repast runtime without the user ever needing to write Java code. The agent behavior editing step is repeated in this example (not shown) for the electric network nodes, using power rather than pressure as the propagated variable.

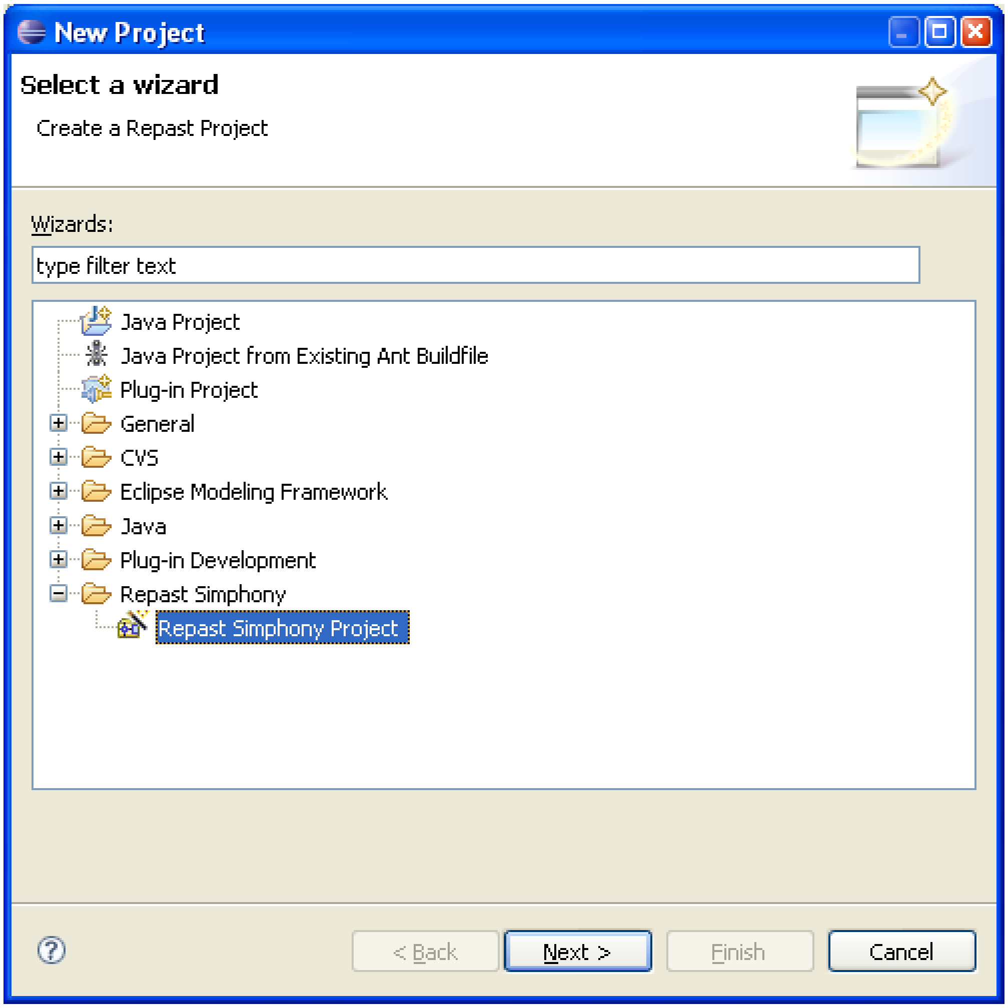


FIGURE 1 New project display wizard with Repast Simphony Project selected

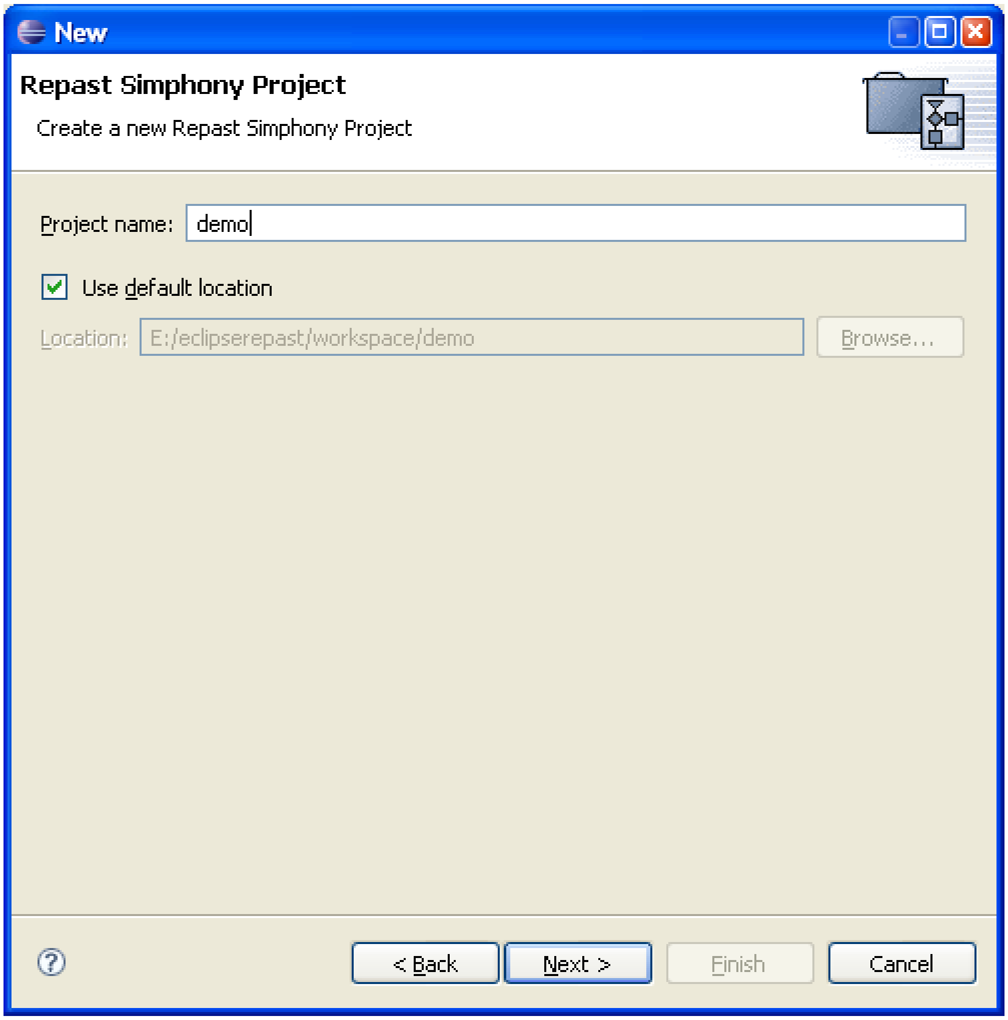


FIGURE 2 Repast Simphony Project new project basic options

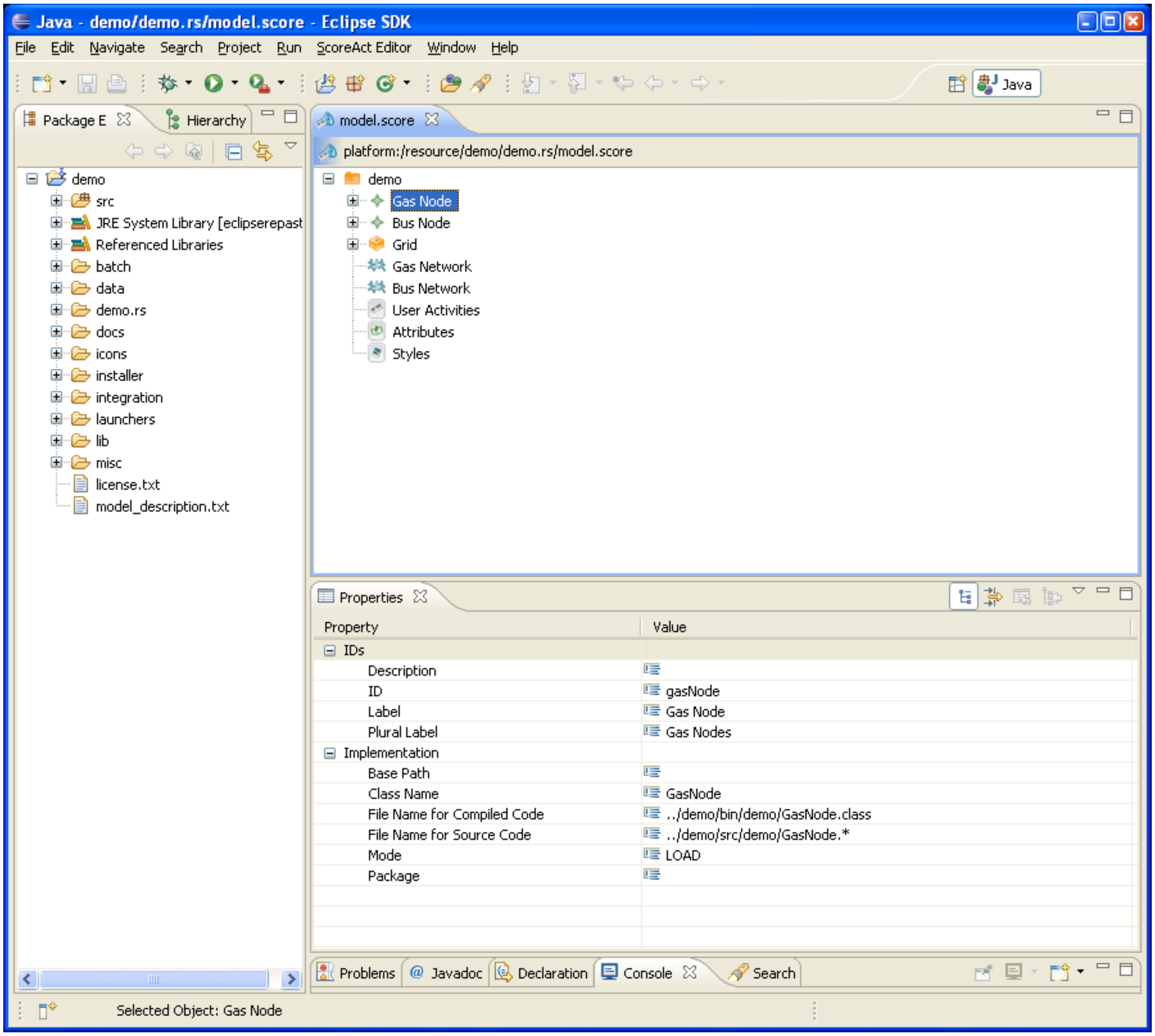


FIGURE 3 Repast Simphony project workspace showing Score editor view

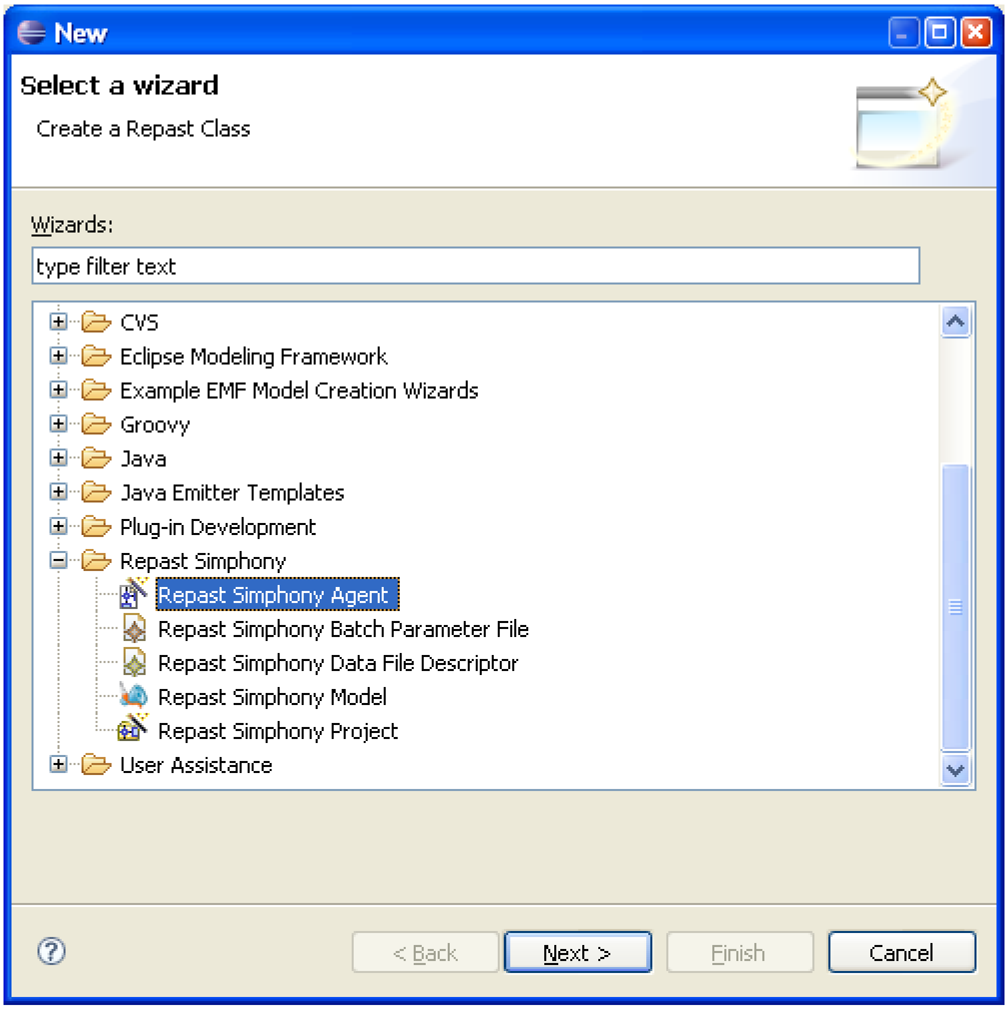


FIGURE 4 New agent creation wizard

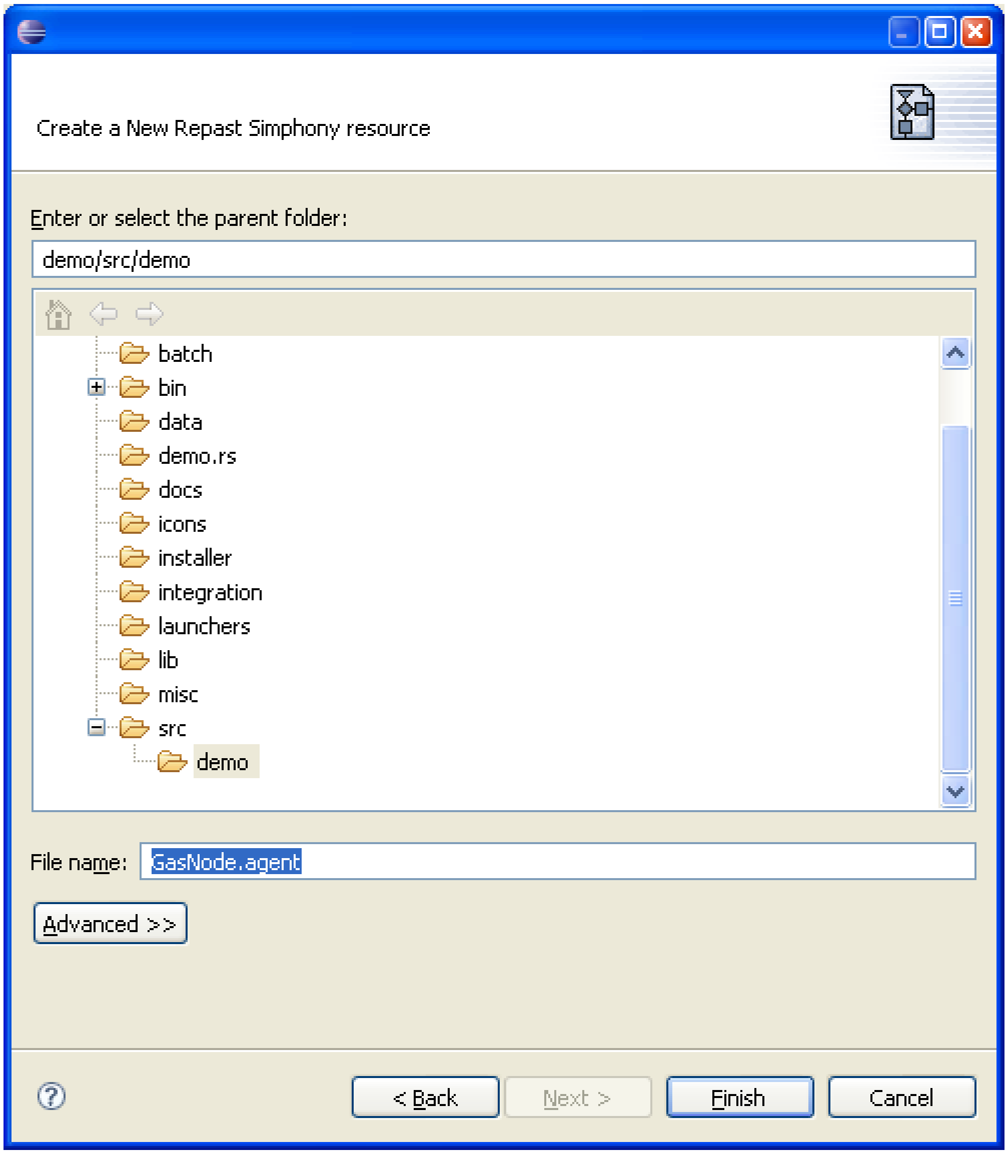


FIGURE 5 New agent creation wizard name option

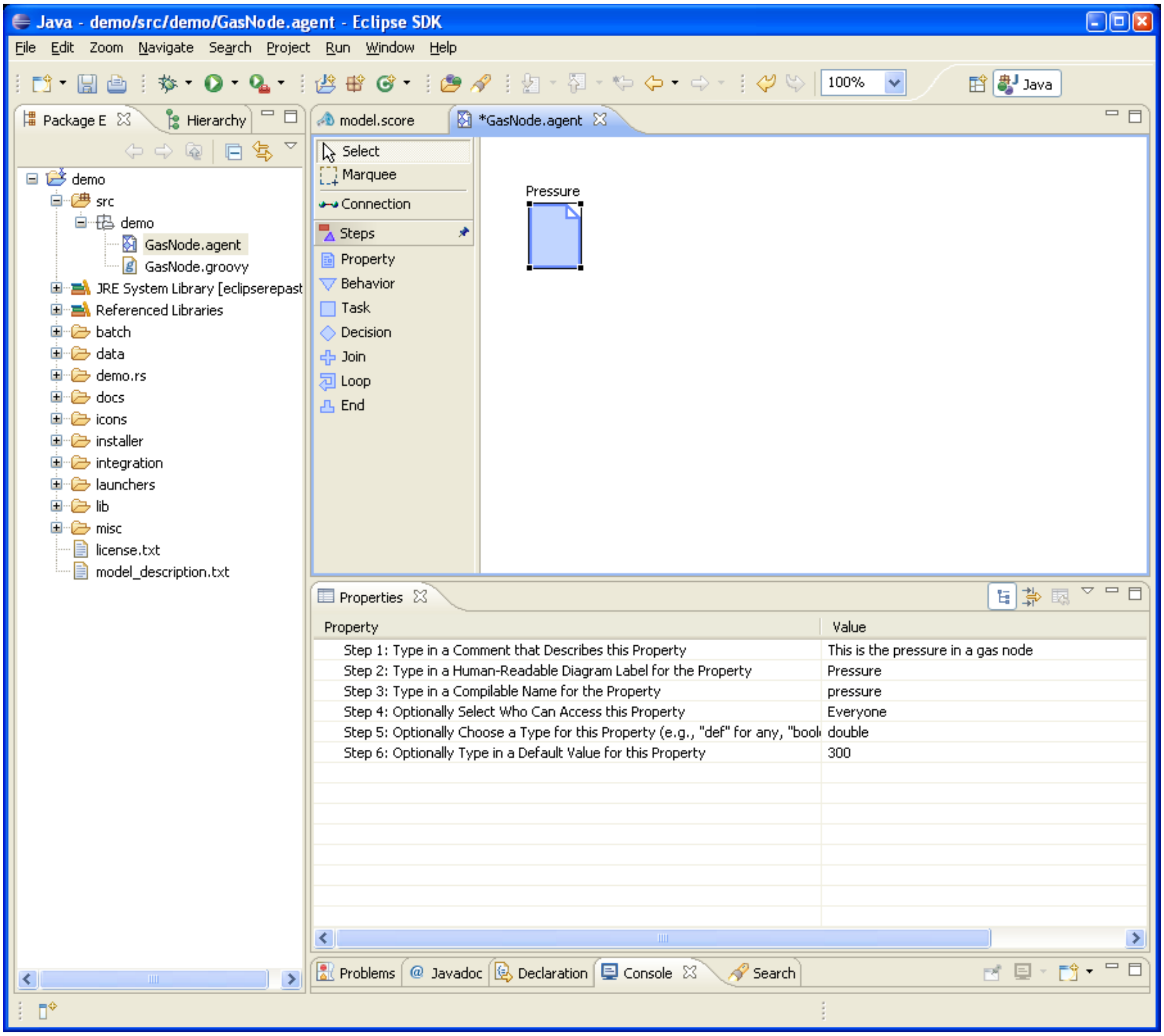


FIGURE 6 Repast agent editor showing creation of an agent property

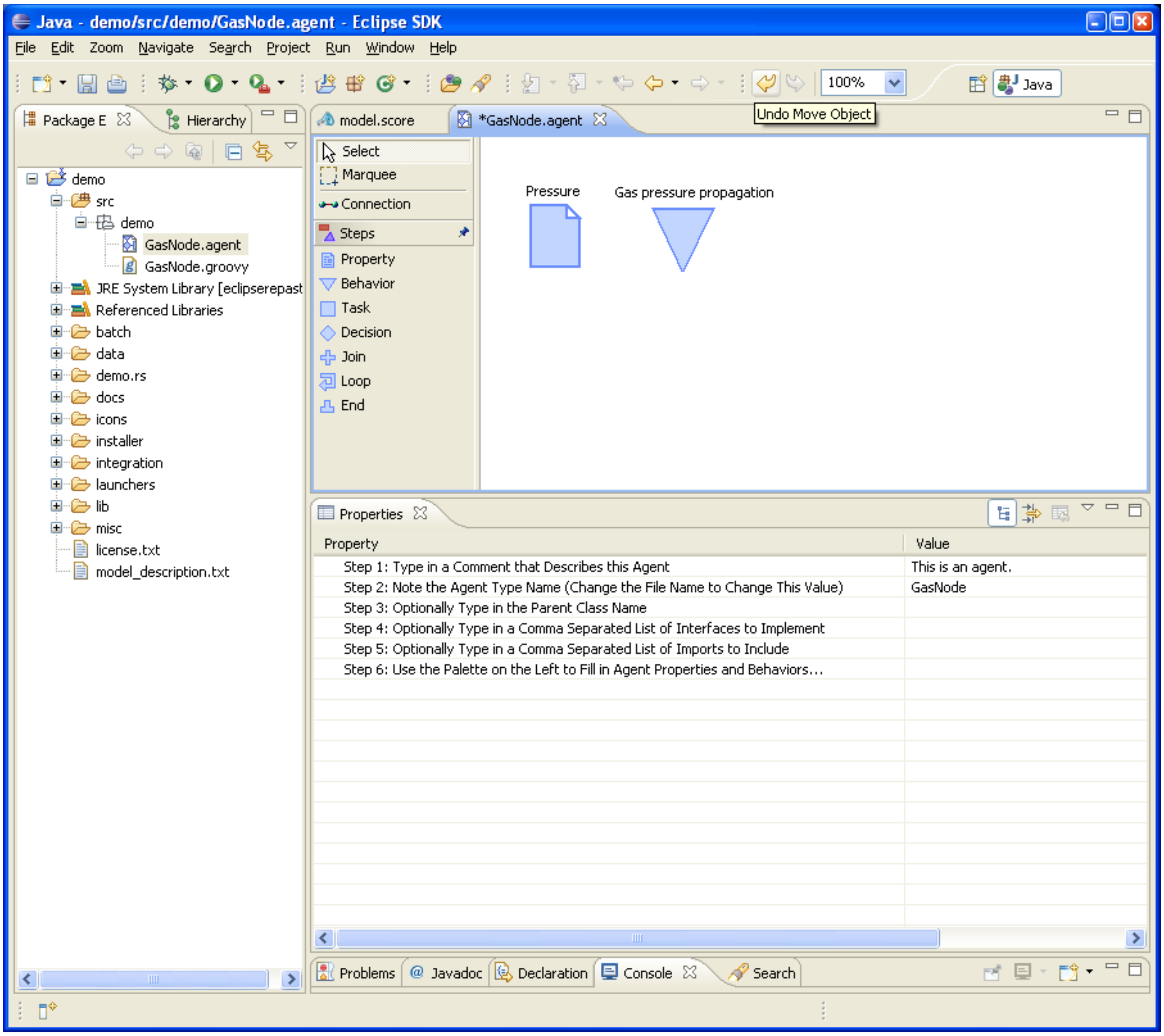


FIGURE 7 Repast agent editor showing creation of an agent behavior

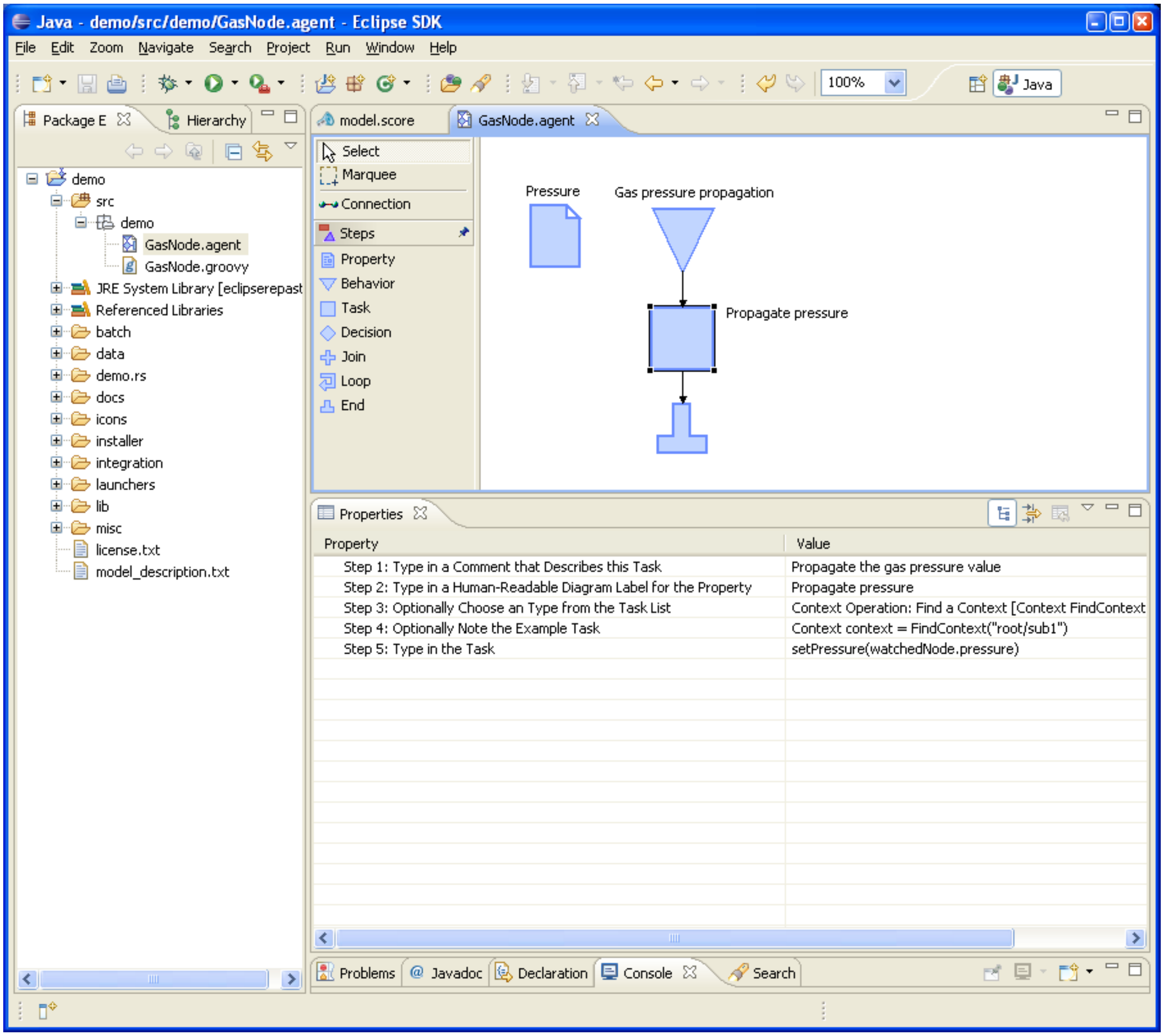


FIGURE 8 Repast agent editor showing creation of an agent task and its linkage with behavior

RUNTIME AGENT CREATION AND DISPLAY DESIGN

After the agents have been created in the Repast agent editor, the runtime may be started from the Repast IDE. Creation of user-specified data collection, output, and display may be performed through wizards in the Repast S runtime. The runtime window contains a scenario tree with contexts that branch from the main model context. The user may access each of the underlying wizards by selecting any component in the tree. Generally, one would create the components in the order of dependency: displays first, followed by data collection and data output components. The runtime graphical user interface (GUI) elements have been discussed in detail by Tatara et al. (2007a), and only the visualization elements will be discussed here.

Displays for two and three-dimensional spatial projections are created by selecting the Displays branch in the scenario tree. The user must specify at least one spatial projection and optionally one or more network projections or value layers. This demo uses a single two-dimensional (2D) grid projection on which the physical network elements are placed. The appearance of the agents is fully customizable and may be specified by the Agent Style wizard in a display item as shown in Figure 9.

The network style editor (Figure 10) allows the user to specify how the network links are visualized, including the line style, color, and width. Additionally, the line width and color may be optionally specified by the agent properties. For example, a high-pressure node may be dark blue in color and a low-pressure node may be light blue.

Additionally, the agent style may be defined by the agent style editor shown in Figure 11. The agent style editor provides options in addition to color and size, such as 2D shape, label data, and label font properties. The agent icon size may also be scaled according to a user-selected agent property. Options for 3D styles include the ability to set the 3D shape, wrap texture maps around the 3D shape, and load third-party 3D model files.

At this point, although the agent display styles have been created, no agent instances exist yet in the runtime, and thus there is nothing to visualize. The user has several options in creating agent instances, including loading from delimited or database files or by using the runtime Repast agent editor shown in Figure 12. The Repast agent editor provides the flexibility and power to create agents and arrange agents in the projections defined by the model. Agents may be created, cloned, and deleted, and agents in the projection may be freely arranged in space. The agent editor uses the styles defined for the display so that each agent type may be easily distinguished from one another.

If the model contains network projection, as with this demo, the agent editor may be used to connect the agents in the network by dragging connections between agents (Figure 13). Multiple network types are supported, and the network being edited is selected from a drop-down box. The network connections are styled on the basis of the specified styles for the display. In the case where a 3D display is to be edited, the agent editor tool will re-project the 3D display onto multiple 2D displays for editing. Figure 14 shows the connected infrastructure networks in 2D and 3D projections in the Repast runtime display.

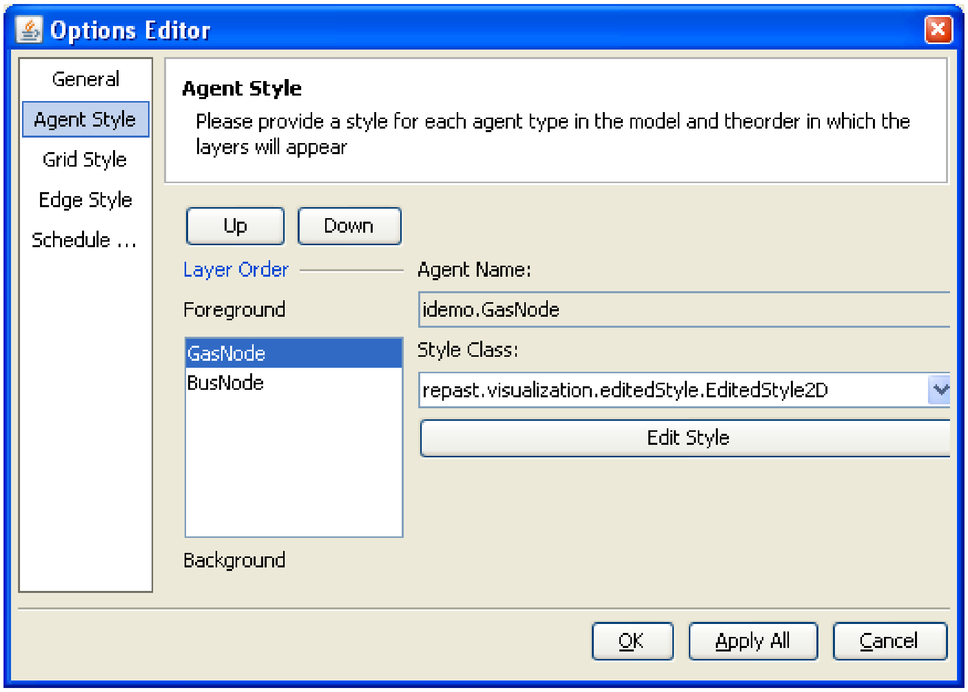


FIGURE 9 Runtime display creation wizard

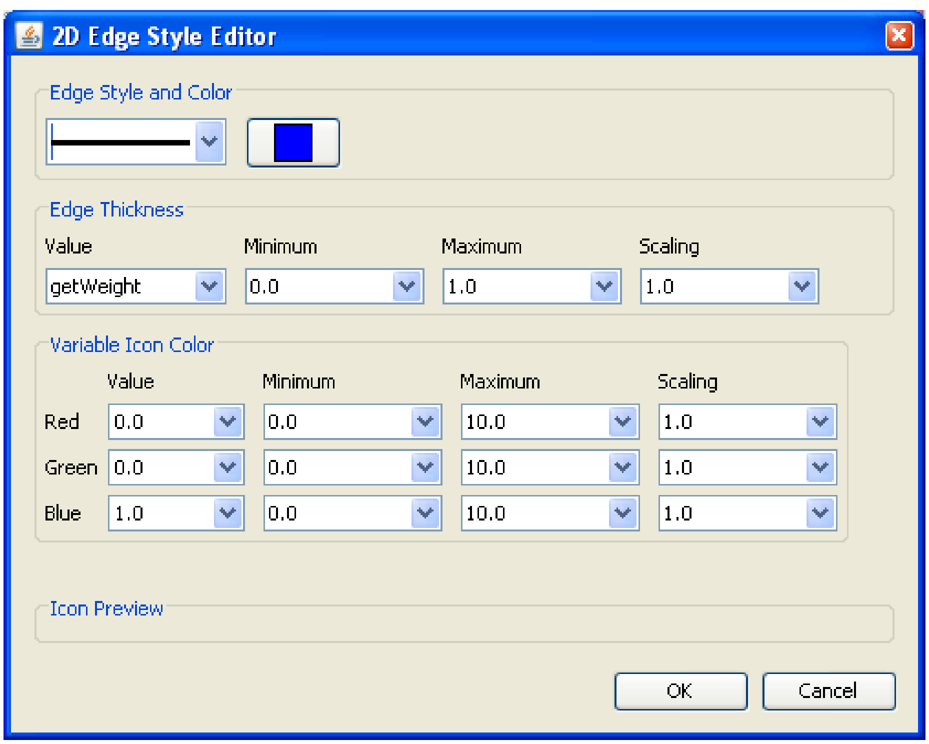


FIGURE 10 Network display style editor tool

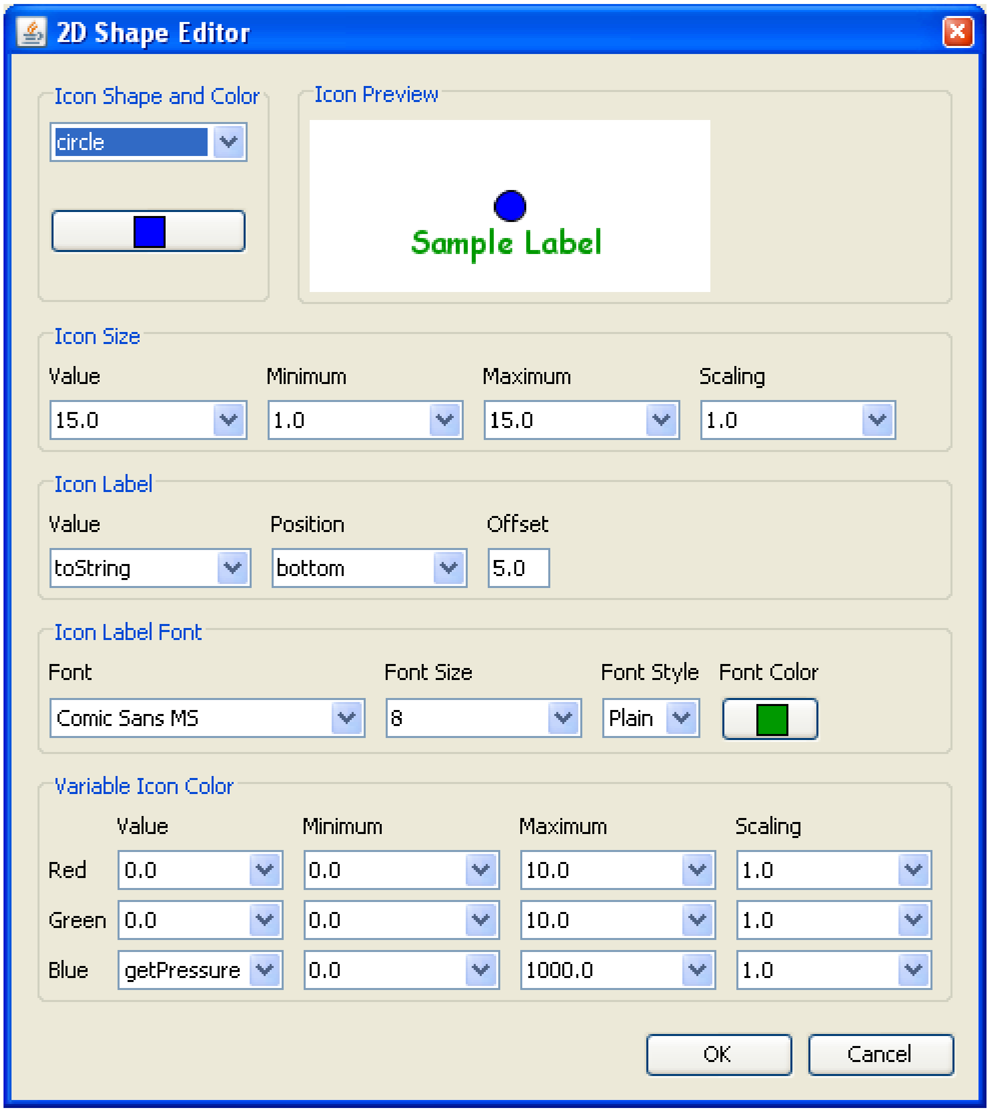


FIGURE 11 Agent display style editor tool

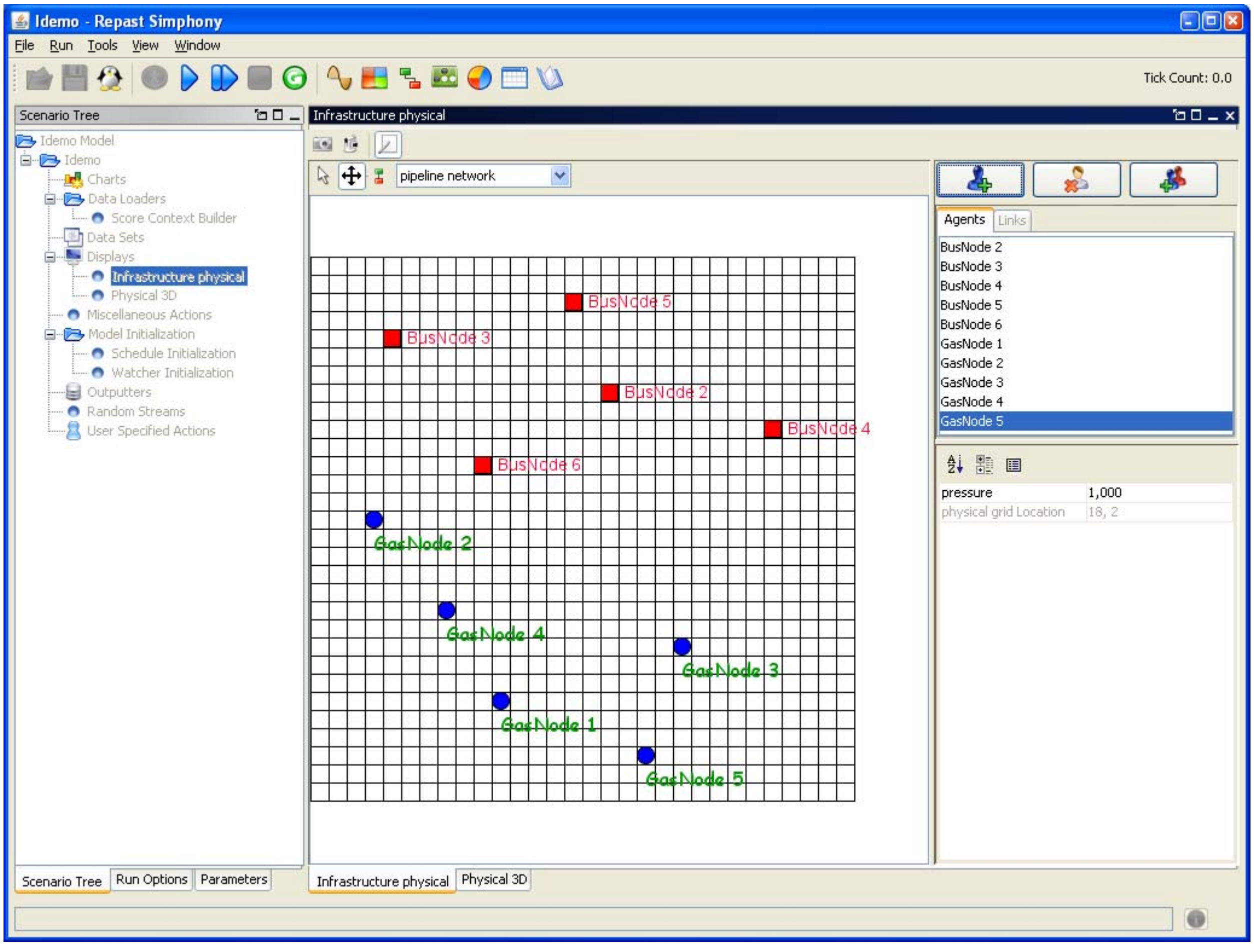


FIGURE 12 Runtime agent instance editor

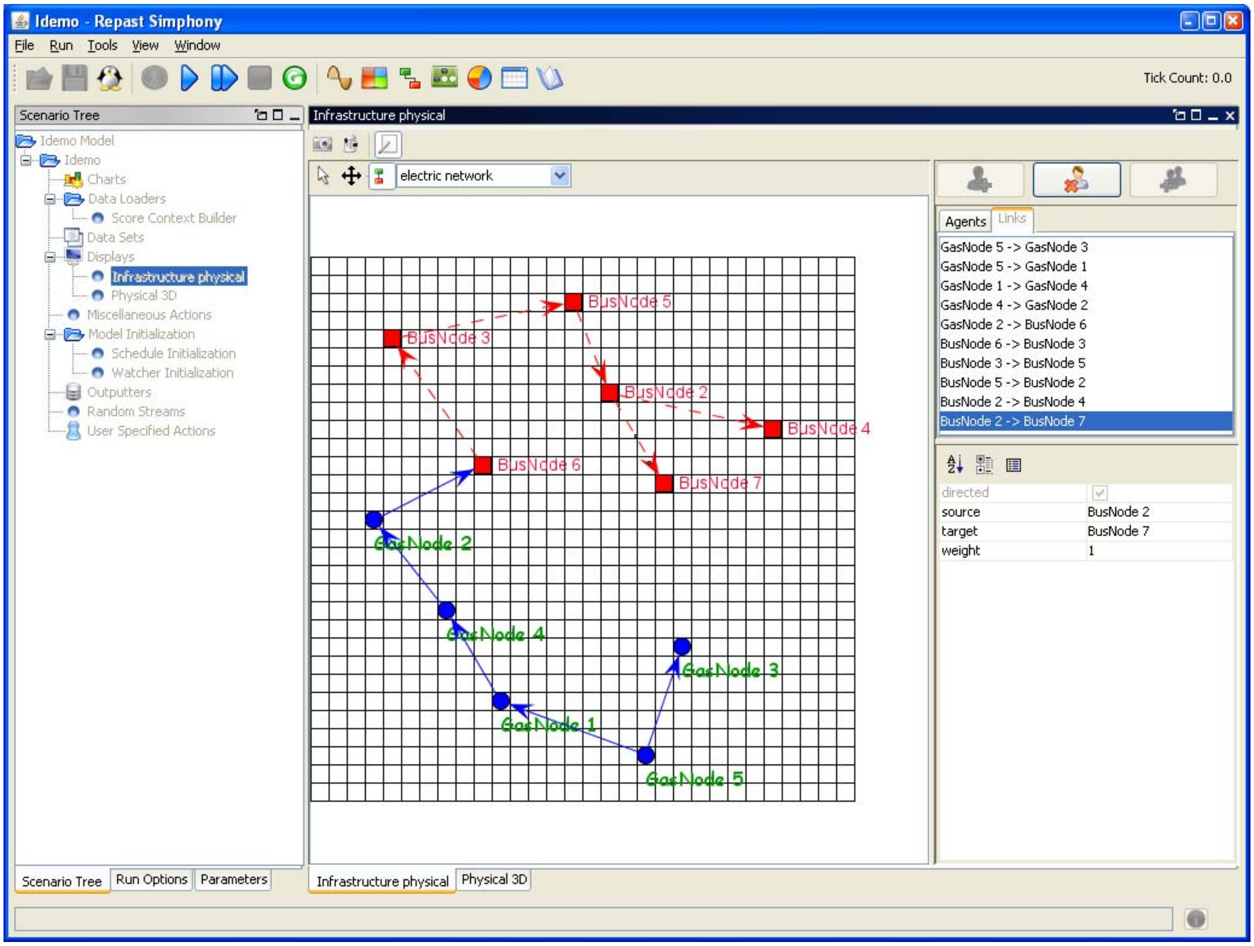


FIGURE 13 Runtime agent instance editor with network links

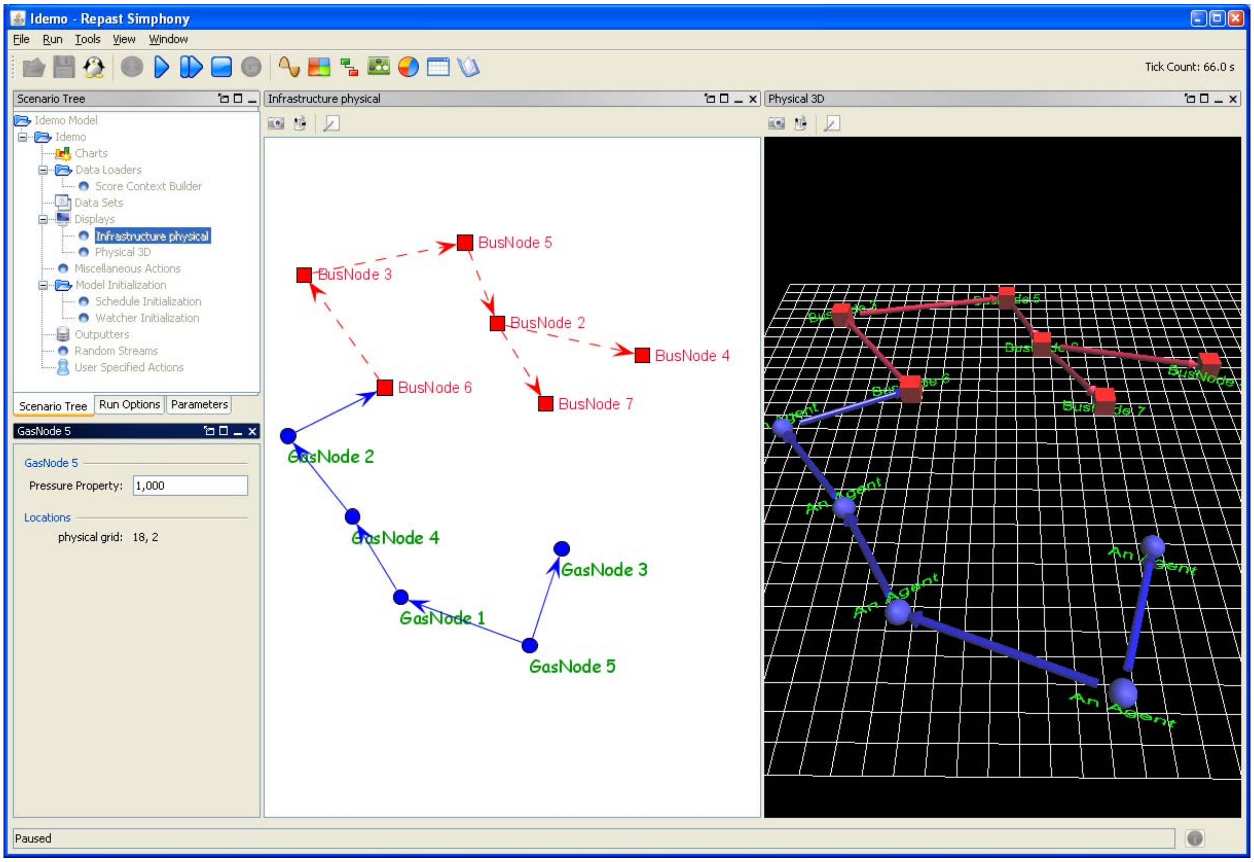


FIGURE 14 Runtime displays showing 2D and 3D visualization of networks

CONCLUSIONS

The Repast S runtime is a pure Java extension of the existing Repast portfolio. Repast S extends the Repast portfolio by offering a new approach to simulation development and execution. The Repast S development environment is expected to include advanced features for agent behavioral specification and dynamic model self-assembly. Any plain old Java object can be a Repast S model component. This paper presents an introductory tutorial and illustration of the visual modeling capabilities of Repast S by using a simple model of interconnected physical infrastructure networks.

ACKNOWLEDGMENT

The authors wish to thank David L. Sallach for his visionary leadership in founding the Repast project and Charles M. Macal for his sustaining involvement in the project. Also, the authors wish to thank Alexander Greif for contributing Flow4J-Eclipse to the software development community. This work is supported by the U.S. Department of Energy, Office of Science, under contract number DE-AC02-06CH11357.

REFERENCES

Greif, A., 2006, *Flow4J-Eclipse Home Page*; available at http://flow4jeclipse.sourceforge.net/.

Howe, T.R., N.T. Collier, M.J. North, M.T. Parker, and J.R. Vos, 2006, "Containing Agents: Contexts, Projections, and Agents" in D. Sallach, C.M. Macal, and M.J. North (eds.), *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects*, ANL/DIS-06-7, co-sponsored by Argonne National Laboratory and The University of Chicago, September 21–23.

North, M.J., T.R. Howe, N.T. Collier, and J.R. Vos, 2005a, "The Repast Simphony Development Environment," in C.M. Macal, M.J. North, and D. Sallach (eds.), *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms*, ANL/DIS-06-1, co-sponsored by Argonne National Laboratory and The University of Chicago, Oct. 13–15.

North, M.J., T.R. Howe, N.T. Collier, and J.R. Vos, 2005b, "Repast Simphony Runtime System," in C.M. Macal, M.J. North, and D. Sallach (eds.), *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms*, ANL/DIS-06-1, co-sponsored by Argonne National Laboratory and The University of Chicago, Oct. 13–15.

North, M.J., N.T. Collier, and J.R. Vos, 2006, "Experiences Creating Three Implementations of the Repast Agent Modeling Toolkit," *ACM Transactions on Modeling and Computer Simulation* 16(1):125, ACM (January): New York, NY.

Ozik, J., M.J. North, D.L. Sallach, and J.W. Panici, 2007, "ROAD Map: Transforming and Extending Repast with Groovy," *Proceedings of the Agent 2007 Conference on Complex Interaction and Social Emergence*, co-sponsored by Argonne National Laboratory and Northwestern University, Nov. 15–17.

Parker, M.T., T.R. Howe, M.J. North, N.T. Collier, and J.R. Vos, 2006, "Agent-Based Meta-Models," in D. Sallach, C.M. Macal, and M.J. North (eds.), *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects*, ANL/DIS-06-7, co-sponsored by Argonne National Laboratory and The University of Chicago, September 21–23.

ROAD (Repast Organization for Architecture and Design), 2005, *Repast Home Page*, Chicago, IL; available at http://repast.sourceforge.net.

Tatara, E., M.J. North, T.R. Howe, N.T. Collier, and J.R. Vos, 2006, "An Introduction to Repast Simphony Modeling Using A Simple Predator-Prey Example," in D. Sallach, C.M. Macal, and M.J. North (eds.), *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects*, ANL/DIS-06-7, co-sponsored by Argonne National Laboratory and The University of Chicago, September 21–23.

Tatara, E., M.J. North, T.R. Howe, N.T. Collier, and M.T. Parker, 2007a, "Building Models in Repast Simphony: A Predator-Prey Example," *Proceedings of the North American Association for Computational Social and Organizational Sciences 2007 Conference*, June 7–9, 2007, Emory University, Atlanta, Georgia.

Tatara, E., M.J. North, J. Dolph, J. Kavicky, and E. Portante, 2007b, "The Simphony Integrated Simulation Framework for Infrastructure Interdependency Modeling," *AIChE Annual Meeting*, Salt Lake City, UT, November 4–9, 2007.

ROAD MAP: TRANSFORMING AND EXTENDING REPAST WITH GROOVY

J. OZIK,* Argonne National Laboratory, Argonne, IL
and The University of Chicago, Chicago, IL
M.J. NORTH, Argonne National Laboratory, Argonne, IL
and The University of Chicago, Chicago, IL
D.L. SALLACH, Argonne National Laboratory, Argonne, IL
and The University of Chicago, Chicago, IL
J.W. PANICI, Northern Illinois University, DeKalb, IL

ABSTRACT

This paper discusses the integration of the dynamic object-oriented programming language Groovy into the Repast Simphony platform. It shows how the integration of Groovy benefits agent-based modeling in a number of ways, including its ability to simplify agent behavior specifications, create expressive and human-readable code, generate complex and adaptive agent behavior, and move between agent-modeling environments.

Keywords: Repast Simphony, Groovy, agent-based modeling, dynamic languages

INTRODUCTION

Dynamic languages have gained popularity in recent years. Groovy (König et al. 2007) is such a dynamic language with the added benefit of tight integration with Java and, hence, the ability to integrate into the Repast (ROAD 2005) agent-modeling platform. Repast Simphony (Repast S) is the latest extension to the Repast portfolio, a widely used, free, and open-source agent-based modeling and simulation (ABMS) toolkit (North et al. 2005a,b). Repast S offers a variety of approaches for developing and executing simulations, and it includes many advanced features for agent storage, display, and behavioral activation and new facilities for data analysis and presentation. This paper explores a few aspects of Groovy integration in Repast S.

A dynamic language such as Groovy is vital for implementing the next generation of ABMS models, such as interpretive agent (IA) models. Although in most agent models, meaning is stipulated by the model designer, interpretive agents are designed to discern and attribute meaning from within the model and shape their communications and actions accordingly.

Groovy capabilities can help achieve such a design goal in multiple ways. Tasks can be defined as closures and passed among agents as needed. Examples of such uses are commands or directives, skills acquired through training, and the imitation of rituals. Each of these can convey a closure to achieve its effect, and furthermore, the closure can be customized by context, arguments, agent state, composition with other closures, or additional embedded closures. In whatever way the customization is achieved, the effect is to calibrate the way in which the

* Corresponding author address: Jonathan Ozik, Argonne National Laboratory, Bldg. 900/DIS, 9700 S. Cass Ave., Argonne, IL 60437; e-mail: jozik@anl.gov.

directive, specialty, or ritual is carried out in a particular situation. Similarly, the creative use of iterators allows diverse agents to receive closures customized to their history and/or circumstances.

The existence of metaobjects allows routine activities to proceed as expected while possible changes in circumstances that would require the construction of a more innovative response are monitored. The metaobjects allow agents to have unique or path-specific capabilities or action orientations.

As such implementations of these dynamic capabilities are incorporated into Repast S, the toolkit will be able, through Groovy, to provide systematic support for IA models. An IA reference application (Interpretive Heatbugs) has been ported to Groovy, to both explore such capabilities and illustrate their use. All such innovations contribute to the forward potential of social agent simulation and the tools necessary to its realization.

As an added example of Groovy/Repast S integration, we introduce ReLogo, a pathway from the NetLogo (Wilensky 1999) "multi-agent programmable modeling environment" to Repast S via Groovy. NetLogo promises easy entry into agent-based modeling for anyone: from elementary school children, for whom it was originally designed, to advanced researchers.

ReLogo involves the creation of a NetLogo lexer and parser in ANTLR (Parr 1989) and the emission of Groovy translations of NetLogo code. The interpreted nature of the NetLogo language makes it especially useful to use a dynamic language such as Groovy as the target language. The expressiveness of Groovy results in easily readable code that retains a lot of the structure of the original agent code. As a consequence, the code can be easily extended and modified within the Repast S environment.

This paper is structured as follows. Section 2 gives a brief introduction to the Groovy dynamic programming language, focusing on the dynamic features that are most relevant to agent modeling. Section 3 presents Groovy integration with Repast S, specifically focusing on some aspects of the Repast S Agent wizard and the visual agent behavior editor, as well as the use of Groovy categories to carry out a variety of mathematical operations, including automatic unit conversions and matrix and calculus operations. The Groovy/Repast S port of Interpretive Heatbugs is discussed in Section 4, which highlights some of the dynamic Groovy capabilities. Finally, ReLogo is briefly presented in Section 5, and conclusions are given in Section 6.

It is important to note that Repast S and its related tools are still under development. This paper presents the most current information at the time it was written. However, changes may occur before the planned final release.

GROOVY

Dynamic languages (e.g., Ruby, Python, JavaScript, or Groovy) have gained popularity in recent years. Features such as dynamic typing enable individuals and small teams of programmers to rapidly develop applications and engage in prototype exploration, while test-driven development mitigates the loss of type checking when moving away from a statically typed environment (e.g., Java, C, or C++). Although it is not a characteristic of every dynamic language, there is usually a great reduction of boilerplate code, making the code much more expressive and readable by humans.

An important point to note, though, is that a lot of the ongoing debate that pits dynamic languages against static languages ignores the reality that some tasks are better done in a dynamic environment and others in a static one. Moreover, when two languages share object orientation, integration issues are more easily localized. Groovy steps into this “sweet spot,” bringing advanced dynamic language features, such as closures, dynamic typing, and the metaobject protocol, to one of the most widely used, robust, and well-supported (statically typed) language, Java, via seamless integration. In fact, since JSR-241 (Java Specification Request), Groovy is the second standard language for the Java platform (the first one being Java).

According to the Groovy Web site (http://groovy.codehaus.org/), Groovy:

- “is an agile and dynamic language for the Java Virtual Machine”
- “builds upon the strengths of Java but has additional power features inspired by languages like Python, Ruby and Smalltalk”
- “makes modern programming features available to Java developers with almost-zero learning curve”
- “supports Domain Specific Languages and other compact syntax so your code becomes easy to read and maintain”
- “makes writing shell and build scripts easy with its powerful processing primitives, object-oriented structure and an Ant DSL”
- “increases developer productivity by reducing scaffolding code when developing web, GUI, database or console applications”
- “simplifies testing by supporting unit testing “
- “seamlessly integrates with all existing Java objects and libraries”
- “compiles straight to Java bytecode so you can use it anywhere you can use Java”

There are a number of capabilities that Groovy brings to the realm of agent-based modeling. First of all, closures allow agent behaviors to be specified as first-class Closure objects. As taken from the Groovy Web site:

> A closure in Groovy is an anonymous chunk of code that may take arguments, return a value, and reference and use variables declared in its surrounding scope. In many ways it resembles anonymous inner classes in Java, and closures are often used in Groovy in the same way that Java developers use anonymous inner classes.

The closures can be customized by context, arguments, agent state, composition with other closures, or additional embedded closures. In whatever way the customization is achieved, the effect is to calibrate the way in which the agent behavior is carried out in a particular situation. In addition, closures can be passed around between agents, enabling agents to acquire capabilities or learn from each other.

The metaobject protocol in Groovy allows for the specification of agent metaclasses. Metaclasses can be used to interrupt routine agent activities when changes of agent circumstances require the agents to construct more innovative responses. Furthermore, with Expando Metaclasses, agent classes can be modified as necessary at run time. These capabilities allow agents to have unique or path-specific capabilities or action orientations.

Finally, Groovy's tight integration with Java provides the ability to (a) seamlessly incorporate Groovy into the sophisticated agent-based modeling platform Repast S and (b) take advantage of the vast amounts of existing Java libraries that support agent-based modeling.

GROOVY INTEGRATION IN REPAST S

Visual Agent Behavior Editor

Repast S includes a point-and-click agent editor. The visual agent behavior editor, the new project wizard, the new agent wizard, and related components are modified forms of Alexander Greif's free and open-source Flow4J-Eclipse components (Greif 2006) that have been adapted specifically for agent-based modeling. Greif (2006) has made the Flow4J-Eclipse system available under a BSD-style free and open-source license. The Repast project team has built on Greif's contribution to create the above-mentioned Repast S components. According to the Flow4J home page (Greif 2006):

> Flow4J-Eclipse is an Eclipse Plug-in for modeling process flows in a drag and drop manner. A process flow can contain process steps (I call them flowlets), which can be linked together [in]to a complex flow.

Flow4J-Eclipse used two types of "flowlets" (Greif 2006) — namely "Control Flowlets like Start-, Decision- and Jump- Flowlets which are configurable in Eclipse" — that "tell 'how' the process should flow" and "Task Flowlets [that] accomplish a specific task that is wrapped in a Java class," which can come from many sources, including Java itself or "scripting languages like Jython, Groovy, JavaScript, etc." Grief (2006) notes the following:

> After designing the flows in the Eclipse Plug-in, the flow's Java source code is automatically created, and is immediately ready for compilation and deployment in a Java web/application. The flow's generated Java source code is highly optimized and lightning fast…. Web/applications can execute flows in two ways: (1) from inside any Java code [and] (2) triggered by a HTTP request if the flow is deployed in a web application.

The Flow4J-Eclipse editor works by presenting users with a flowchart-drawing interface that is used to create a process diagram (Grief 2006). The process diagram is saved in a platform-independent XML file. Following the Eclipse development model, the Flow4J-Eclipse system registers a builder with Eclipse that is notified whenever a Flow4J-Eclipse XML file changes (Grief 2006). This builder creates a Java source file from the freshly saved XML file. Eclipse itself then automatically generates a Java binary class file from the new Java source code file. The Flow4J-Eclipse system also includes an Eclipse wizard for creating new Flow4J-Eclipse projects, an Eclipse wizard for creating new flowchart XML files, and supporting tools such as a project menu option to add the Flow4J-Eclipse nature to a standard project (i.e., activate the Flow4J-Eclipse builder for the project).

Repast agent editor adapts Greif's Flow4J-Eclipse to agent-based modeling in multiple ways:

1. The Repast agent editor's builder generates Groovy source code rather than Java source code to take advantage of a number of Groovy features.
2. The Repast agent editor has enhanced the Flow4J-Eclipse property views with step-by-step form-style inputs.
3. The Repast agent editor redefines the Flow4J-Eclipse flowlets to correspond to agent behavior primitives and properties rather than "process flows" (Greif 2006):
   a. Flow4J-Eclipse "Flows" have been redesigned to support Repast agent class definitions,including user input for super classes and implemented interfaces.
   b. Flow4J-Eclipse "Template Flowlets" have been redesigned as "Property Components" to support agent attributes.
   c. Flow4J-Eclipse "Start Flowlets" have been redesigned as "Behavior Components" to support Repast scheduling and watchers (Howe et al. 2006; Parker et al. 2006).
   d. Flow4J-Eclipse "Task Flowlets" have been redesigned as "Task Components" to support Repast agent activities.
   e. Flow4J-Eclipse "Decision Flowlets" have been redesigned to use Groovy logical conditions instead of Flow4J-Eclipse predicate lookup tables.
   f. Flow4J-Eclipse "Call Flowlets" have been redesigned as "Loop Components" to loops instead of method calls.
   g. Flow4J-Eclipse "Jump Flowlets" have been removed.
4. The Repast agent editor includes the ability to embed user-specified comments into the generated Groovy source code.
5. The Repast agent editor's XML storage schema has been modified versus Flow4J-Eclipse to reflect the above-cited differences in the agent behavior primitives and properties (Greif 2006). Furthermore, to differentiate the contents, the Repast agent editor uses a different file extension for its XML files than the Flow4J-Eclipse system.
6. The Repast agent editor automatically generates a set of supporting components within each agent. For example, each generated Repast agent class includes an attribute and method for automatically assigning the agent a human-readable run time identifier.
7. The Repast system has expanded the functionality of the Flow4J-Eclipse wizard for creating new projects by adding a variety of features, such as the following:
   a. Repast Score file input specification and generation has been added.
   b. A large number of Repast-specific supporting files and directories are now generated for tasks such as defining batch runs, defining legacy model descriptors, and building model installations.

c. Eclipse launch scripts are added for executing interactive model runs, executing batch model runs, and starting Repast in default mode.

d. Repast library dependencies are included in the generated project.

8. The wizard for creating new flowchart XML files now creates Repast-specific files.

9. The Flow4J-Eclipse nature has been modified to include Repast library dependencies.

Figure 1 shows an example flowchart in the agent editor, while Figure 2 shows the corresponding Groovy code. Please see North et al. (2007) for more details on how to use the agent editing system.

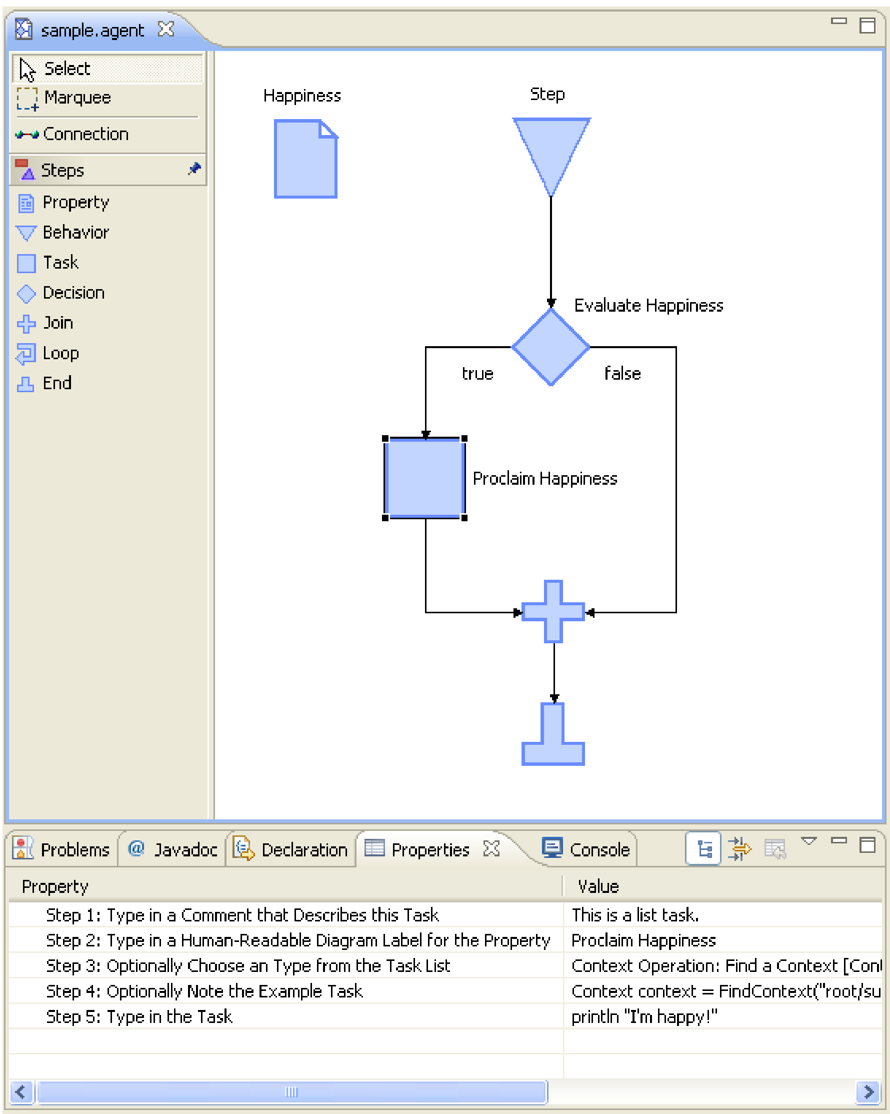


Figure 1 Example flow chart and Properties pane in the Repast S visual agent behavior editor

```
...
    /**
     *
     * This is an agent property.
     * @field happiness
     *
     */
    @Parameter (displayName = "Happiness", usageName = "happiness")
    public def getHappiness() {
         return this.happiness;
    }
    public void setHappiness(def newValue) {
         this.happiness = newValue;
    }
    public def happiness = 0;

    /**
     *
     * This is the step behavior.
     * @method step
     *
     */
    @ScheduledMethod(
          start = 1d,
          interval = 1d
    )
    public void step() {
         // Define the return value variable.
         def returnValue
         // Note the simulation time.
         def time = GetTickCountInTimeUnits();
         // Use the Repast Simphony Groovy math tools.
         use (MathOperations.mathCategories()) {
             // Make a decision.
             if (happiness > 0.5) {
                 // This is a list task.
                 println "I'm happy!";
             } else  {

             }
             // Exit this scope.
             return;
         }
    }
...
```

Figure 2 Condensed parts of the automatically generated Groovy code corresponding to the Repast S visual agent behavior editor flowchart in Figure 1

JScience and Groovy Categories

JScience is a Java library developed to “provide the most comprehensive Java library for the scientific community” (JScience 2005). Users developing a Repast S model could reference the JScience library like they would any other Java library, but, as will be demonstrated below, our intention was to make this process more user-friendly and create more human-readable code. To do this, we employed Groovy Categories and simplified the use of a subset of the JScience capabilities: the scientific unit and unit conversion modules and linear algebra modules.

Categories are useful for situations in which one would like to define additional methods on classes that are not under one’s control. Specifically, a Groovy category is a class that contains a set of static methods (called *category* methods). Each of these methods is made available on the class of the method’s first argument. The following example helps to clarify these concepts.

Figure 3 shows an example of the Java code needed to define Amount class objects *x* and *y* with the values 2 kilometers and 5,000 feet, respectively, along with the calculation of their sum, *z*, which properly accounts for the unit conversions by setting *z* to 3.524 kilometers.

```
Amount x = Amount.valueOf(2.0, SI.KILO(SI.METER));
Amount y = Amount.valueOf(5000.0, SI.FEET);
Amount z = UnitsOperations.addition(x,y);
```

Figure 3 Java code utilizing the Amount JScience class

Figure 4 shows the Groovy version of Figure 3 using categories.

```
def x = 2.kilometers
def y = 5000.feet
def z = x + y
```

Figure 4 Groovy code corresponding to Java code in Figure 3 (at end *z* = 3.524 km)

This simplification is possible for two main reasons. The first is that Groovy (unlike Java) allows one to override *operators*. Hence, in the example above, we were able to override the “+” and the “.” operators, resulting in intuitive and human-readable code. The second reason we can simplify is because we can enclose any lines of code within a closure and, by using the *use* keyword and referencing the appropriate Groovy category, [for example, `use(UnitsCategory) {`**`lines of code`**`}`], we can force the operators (or methods) within those lines of code to be overridden by the category’s [in this case, **`UnitsCategory`**’s] static methods. The lines of code in Figure 4 are located within such a closure.

In an analogous way, by using the **`MatrixCategory`**, which references the JScience linear algebra classes, matrix operations (e.g., matrix addition, subtraction, multiplication, and exponentiation) can be written in the simple syntax shown in Figure 5.

```
def c = a + b
def d = a – b
def e = a * b
def f = a ** 2
```

Figure 5 Matrix operations using Groovy categories (*a* and *b* are matrices)

In addition, we have also included a **`CalculusCategory`**, which allows for calculating derivatives and integrals of closures.

Such capabilities are built into Repast S, enabling the model developer to tap into these powerful functionalities and, at the same time, create human-readable and thereby more maintainable code.

INTERPRETIVE HEATBUGS

The purpose of the IA research program is to incorporate endogenous meaning attribution as a means of orienting agent communication and action selection into agent-based modeling (Sallach 2003). Interpretive Heatbugs (IHB) is an IA reference application in which interpretive mechanisms (prototype reasoning, situation definition, and orientation accounting) are developed, illustrated, and made available to other researchers (see Sallach and Mellarkod 2005; Mellarkod and Sallach 2005; Sallach and Ozik 2007). Similar to its widely known generic heatbug predecessor (Swarm Development Group 1997), it is designed to provide a simple introduction to the IA paradigm.

IHB uses the familiar heatbugs environment, in which heat-emitting bugs require temperate zones and flee from settings that are uncomfortably hot or cold. Because each bug emits a small amount of heat, congregations of bugs initially create the needed warmth, while overcrowding creates excessive heat. These competing influences give rise to the often-observed complex, churning patterns. To this dynamic of temperature fluctuations driving the bug movements, IHB adds the capability of bugs to ignore, engage in voluntary exchange with, or undercut each other, and it also adds ethnic and religious identities that mediate the decisions to help or hinder.

The IHB application explores the role of interpretation: in the use of force as a means of improving bug circumstances; in the bugs' decisions to request gifts of energy (a placeholder for health and/or wealth); and in the decisions to convey all, part, or none of the energy requested. These decisions draw on the projection of a bug's comfort levels on others, as well as the prototype categorization of all the bugs accessible for interaction. Of particular interest is the emergence of endogenous evaluative classification in bug orientation and action.

Consistent with the concept of a reference application, the interpretive mechanisms that support the acts of aggression, requests, and voluntary contributions are designed to support comparable decisions in a variety of complex social applications (e.g., models of cultural conflicts, such as genocide and ethnic cleansing, as well as diversifying markets and a range of extended cultural processes). The IHB application, which was initially implemented in the

J programming language (Iverson and Hui 1990), has been ported to the Repast S platform (see Figure 6) by using Groovy's dynamic capabilities and making these mechanisms available for the development of increasingly socially "thick" and interactive agent models. What follows is an illustrative set of examples of Groovy-isms used within IHB.

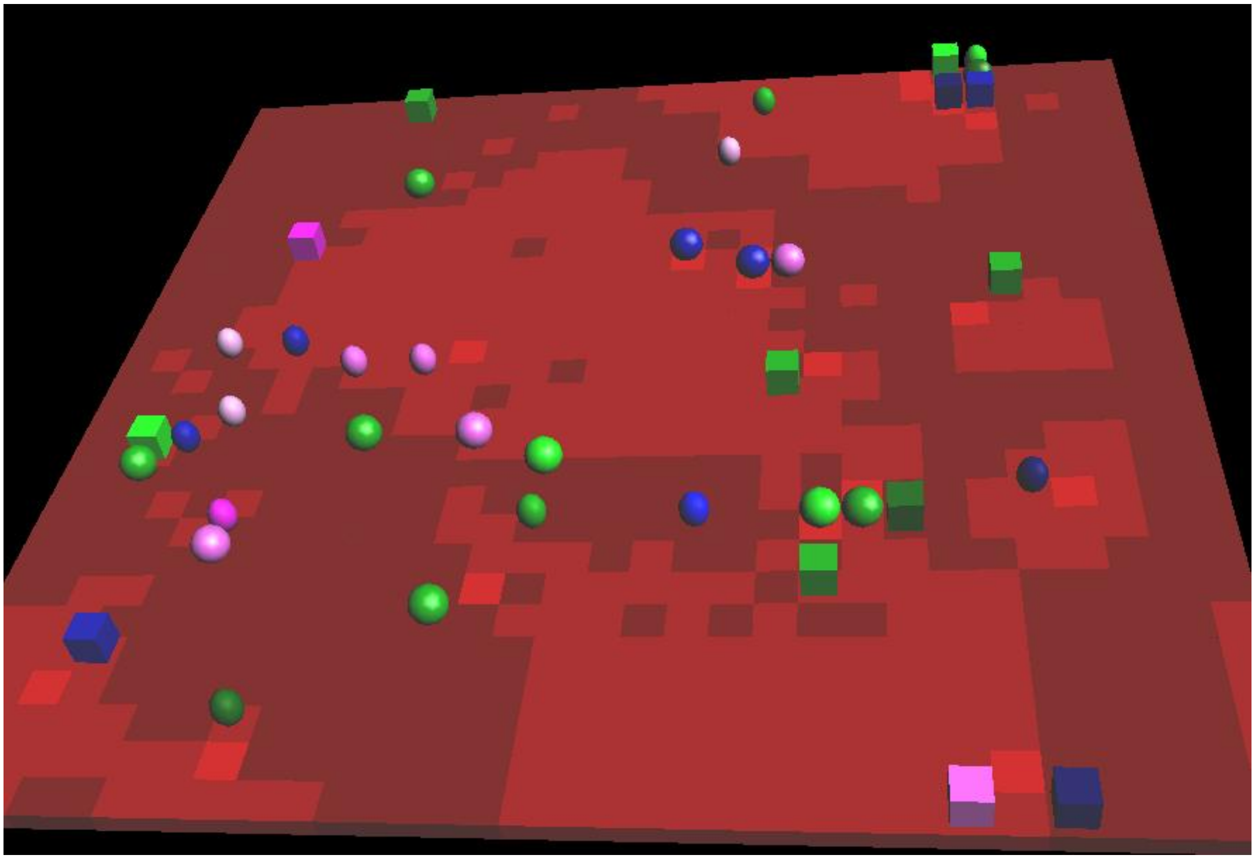


Figure 6 Screenshot of the 3D grid projection from the Groovy/Repast S port of IHB

The bugs make assessments of the bugs that they come into contact with. This involves creating prototype clusters that group a bug's known bugs into categories on the basis of each bug's idiosyncratic history and observations. The current implementation of IHB employs a hierarchical agglomerative clustering algorithm for these groupings and uses the standard Euclidean metric. However, it also includes the flexibility to introduce any user-defined metric via closures, as is shown in the Groovy code snippet in Figure 7.

```
protected def cluster(int maxClusters, def ptsToCluster, Closure metric) {
   …
}
```

Figure 7 Groovy code illustrating the `cluster` method taking a Closure object as a parameter

Depending on the type of bug and its situated context, a bug will adopt a particular set of behaviors for shoving, asking, and giving. As the bugs' shove, ask, and give rules are all implemented as closures, the bugs' behaviors can be assigned and modified easily by using rule dispatchers, which pass bug behaviors, specified as closures, to each bug as necessary. This allows for not only a flexible way in which to determine bug behavior but also an intuitive and straightforward way to encapsulate bug behavior, in a first-class Closure object, instead of requiring a lot of scaffolding code (for example, in a method within a first-class object).

One of the most powerful uses of closures in IHB has to do with a bug's **`more`** and **`all`** methods. In many situations, a bug's action is contingent on whether most or all of the bugs under consideration fit some set of criteria, where these criteria are implemented as closures. Instead of having to create different implementations of **`more`** and **`all`** corresponding to each combination of the many criteria that a bug can employ, the set of criteria are passed to the **`more`** and **`all`** methods as closures. Then Groovy's handy iterating mechanisms interate the bugs under consideration. This is a good example of using combinations of basic closures to implement complex agent behaviors. The closures can be composed, combined, and reordered to provide flexibility in creating new and adaptive agent behaviors.

RELOGO

ReLogo is a pathway from the NetLogo (Wilensky 1999) "multi-agent programmable modeling environment" to Repast S that uses Groovy. As previously mentioned, NetLogo promises an easy entryway into agent-based modeling (Wilensky 1999). Repast S is a very sophisticated agent-based modeling platform, offering many advanced features for agent storage, display, and behavioral activation and new facilities for data analysis and presentation, while also allowing for the integration of external (legacy) models. Thus, ReLogo offers the ability to go from an exceedingly user-friendly and intuitive environment for model development and exploration in NetLogo all the way to the enterprise-level models developed in Repast S.

ReLogo involves the creation of a NetLogo lexical analyzer or "lexer" and parser in ANTLR and the emission of Groovy translations of NetLogo code. The interpreted nature of the NetLogo language makes it especially useful to employ a dynamic language like Groovy as the translation target language. The expressiveness of Groovy results in easily readable code that retains a lot of the structure of the original agent code. As a consequence, the code can be easily extended and modified within the Repast S environment.

Although ReLogo is still under development, it is nonetheless possible to present a simple example to illustrate some of the benefits of using Groovy. One of the most common patterns observed in NetLogo models is that of sending a block of code to a set of agents. In the following NetLogo code (Figure 8), rabbit agents are created, and a block of code specifying the created rabbit's color, position, and energy is sent to each created rabbit. (Note that the block of code is interpreted by each rabbit individually; thus, each rabbit sets its own random position and energy.) Implementing this type of behavior in the static Java language would result in convoluted code that would be difficult to maintain or even read. Even if the resulting code was brought over to the Repast S environment, editing, modifying, or debugging it would become an unenviable task.

```
create-rabbits number [
      set color white
      setxy random-xcor random-ycor
      set energy random 10
]
```

Figure 8 Sample NetLogo code with code block (Rabbits Grass Weeds model) (Wilensky 1999)

On the other hand, Figure 9 shows how this agent behavior can be implemented with Groovy. The **`createRabbits`** method can take as parameters the *number* variable as well as the code block (as a closure). As one can see, the resulting code closely mirrors the original code in structure and in the function of the resulting code units, allowing the modeler to carry over many of the mental constructs that may have developed during model development in the NetLogo environment.

```
createRabbits(number){
      setColor(Color.white)
      setXY(randomXcor(),randomYcor())
      setEnergy(nextIntFromTo(1,10))
}
```

Figure 9 Groovy implementation of code example in Figure 8

CONCLUSIONS

This paper demonstrates the many benefits Groovy brings to the agent-based modeling realm. Groovy integration in Repast S simplified the use of the visual agent behavior editor as well as the incorporation of physical unit conversions, matrices and calculus operations into agent-based models. The Groovy and Repast S implemented Interpretive Heatbugs reference application demonstrated some of the flexibility and clarity achieved by the use of closures. Finally, this papers shows how having Groovy as the target language greatly improves the ability to move from the NetLogo to the Repast S environment via ReLogo. The authors believe that all such innovations will contribute to the forward potential of social agent simulation and the tools necessary for its realization.

ACKNOWLEDGMENTS

The authors wish to thank Alexander Greif for contributing Flow4J-Eclipse to the software development community. They also wish to thank Uri Wilensky, Seth Tissue, and Bill Rand for the helpful discussions and support received on the ReLogo project. In addition, they want to thank Eric Tatara for his helpful contributions. This work is supported by the U.S. Department of Energy, Office of Science, under contract DE-AC02-06CH11357.

REFERENCES

Greif, A., 2006, *Flow4J-Eclipse Home Page*; available at http://flow4jeclipse.sourceforge.net/

Howe, T.R., N.T. Collier, M.J. North, M.T. Parker, and J.R. Vos, 2006, "Containing Agents: Contexts, Projections, and Agents," in *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects*, ANL/DIS-06-7, co-sponsored by Argonne National Laboratory and The University of Chicago, Sept. 21–23.

Iverson, K., and R. Hui, 1990, *J Software Home Page*; available at http://www.jsoftware.com .

JScience, 2005, *JScience*, home page; available at http://jscience.org .

König, D., A. Glover, P. King, G. Laforge, and J. Skeet, 2007, *Groovy in Action*, Manning Publications.

Mellarkod, V.S., and D.L. Sallach, 2005, "Interpretive Heatbugs: Design and Implementation," in *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms*, ANL/DIS-06-5, co-sponsored by Argonne National Laboratory and The University of Chicago, Oct. 13–15.

North, M.J., T.R. Howe, N.T. Collier, and J.R. Vos, 2005a, "The Repast Simphony Development Environment," in *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms*, ANL/DIS-06-5, co-sponsored by Argonne National Laboratory and The University of Chicago, Oct. 13–15.

North, M.J., T.R. Howe, N.T. Collier, and J.R. Vos, 2005b, "Repast Simphony Runtime System," in *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms,* ANL/DIS-06-5, co-sponsored by Argonne National Laboratory and The University of Chicago, Oct. 13–15.

North, M.J., E. Tatara, N.T. Collier, and J. Ozik, 2007, "Visual Agent-Based Model Development with Repast Simphony," in *Proceedings of the Agent 2007 Conference on Complex Interaction and Social Emergence*, ANL/DIS-07-2, co-sponsored by Argonne National Laboratory and Northwestern University, Nov. 15–17.

Parker, M.T., T.R. Howe, M.J. North, N.T. Collier, and J.R. Vos, 2006, "Agent-Based Meta-Models," in *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects*, ANL/DIS-06-7, co-sponsored by Argonne National Laboratory and The University of Chicago, Sept. 21–23.

Parr, T., 1989, ANTLR, home page; available at http://www.anltr.org.

ROAD (Repast Organization for Architecture and Design), 2005, *Repast, News,* Chicago, IL; available at http://repast.sourceforge.net.

Sallach, D.L., 2003, "Interpretive Agents: Identifying Principles, Designing Mechanisms," in *Agent 2003, Conference on Challenges in Social Simulation,* Proceedings of the Conference, co-sponsored by Argonne National Laboratory and The University of Chicago, Oct. 2–4.

Sallach, D.L., and V.S. Mellarkod, 2005, "Interpretive Agents: A Heatbug Reference Simulation," in *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms,* ANL/DIS-06-5, co-sponsored by Argonne National Laboratory and The University of Chicago, Oct. 13–15.

Sallach, D.L., and J. Ozik, 2007, "Interpretive Heatbugs," Argonne National Laboratory, Argonne, IL, in preparation.

Swarm Development Group, 1997, *Swarm,* Albuquerque, N.M.; available at http://www.swarm.org/examples-heatbugs.html . Albuquerque, NM.

Wilensky, U., 1999, *NetLogo,* Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL; available at http://ccl.northwestern.edu/netlogo/.

MODEL EXPLORATION MODULE

[1]M.D. IVÁNYI, ELTE-IKKK Simulation Center, Budapest, Hungary
L. GULYÁS, AITIA International, Inc., Budapest, Hungary
R. BOCSI, AITIA International, Inc., Budapest, Hungary
G. SZEMES, ELTE-IKKK Simulation Center, Budapest, Hungary
R. MÉSZÁROS, AITIA International, Inc., Budapest, Hungary

INTRODUCTION

MEME stands for the Model Exploration Module of the Multi-Agent Simulation Suite developed at AITIA International, Inc. MEME is an agent-based simulation tool enabling modelers to design experiments, run models on clusters and grids of computers, and to manage, analyze and visualize the data produced. Development of the tool begun in mid 2006 after the realization that there was hardly any software available being able to run a large number of simulations, collect, organize and visualize their data for modelers with limited programming skills. This paper gives an insight into the motivations of the development and introduces the reader to the tasks and functions MEME is capable of.

Keywords: Agent-Based Modeling, Modeling Tools, Simulation, Experiment Design

MOTIVATIONS

Agent-based modeling and simulation based computational science demonstrates great promise, but as Bankes and Lampert (2004) argues, so far it has lacked the rigor that is needed in the scientific field and the robustness required in policy making. Models of complex social systems typically depend on a number of assumptions, quantified in the form of specific values to certain model parameters. Ideally, any such model should be tested with any meaningful combination of these parameters, in order to determine the validity of the model or ensembles of models. This process is often called the *parameter space search or parameter sweep*. Additionally the task of establishing the results' statistical validity also involves running the simulation with various random number generator seeds and analyzing the collected results.

* *Corresponding author address:* Marton D. Ivanyi, ELTE-IKKK Simulation Center, Pazmany Peter Setany 1/C, Budapest, Hungary, H-1117; e-mail: mivanyi@aitia.ai

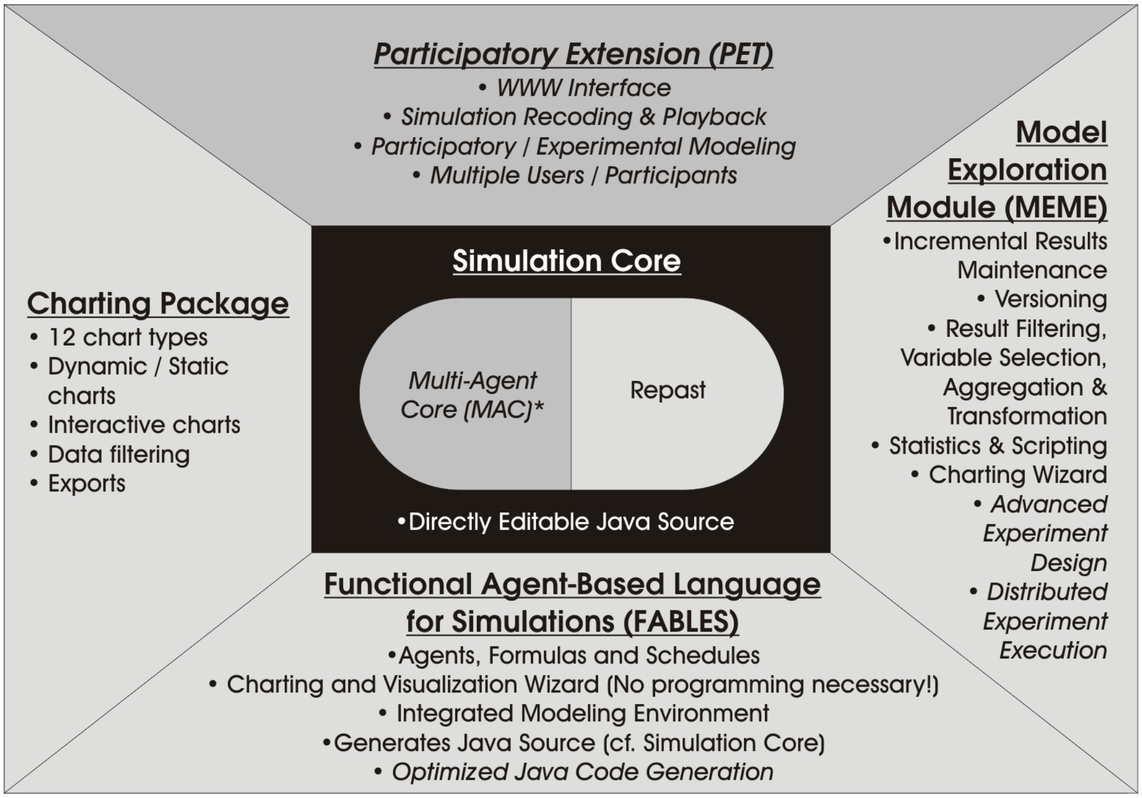


FIGURE 1 MEME is just an element of the Multi-Agent Simulation Suite (structure diagram above).

Running simulations at this scale and complexity is from a computational point of view a particularly intensive task. It is understood that it is essentially impossible to achieve complete parameter space exploration. This creates a special emphasis on the design and sequencing of the experiments, so that they allow for branching depending on earlier results and also for revisiting previously explored areas with greater 'resolution'. Even in case of carefully designed experiment plans, these tasks may exceed the abilities of today's PCs or workstations. Therefore it would be desirable to distribute simulations among several computers on the network, on a local cluster or a grid.

AVAILABLE APPLICATIONS

Standard ABM modeling packages like Swarm, Repast or NetLogo, all offer some support for collecting results in 'batch mode', as opposed to the primary, 'GUI mode' running of simulations. Swarm and Repast require (and support) modelers to create a distinct, 'batch mode' version of their models, offer a 'parameter language' to describe the regions of the parameter space the user wishes to explore, and collect the results in a specially formatted text file. The expressiveness of the offered parameter languages, however, limits the search strategies mostly to regular hyper-parallelepipeds. In particular, no dynamic branching of the exploration is possible, except by hand (i.e. processing the results of one batch and writing the next parameter file appropriately). Furthermore Swarm doesn't have a graphical user interface for designing parameter files, while the latest version of Repast has a simple GUI for parameter space explorations.

NetLogo, on the other hand, provides a GUI for model exploration, but no parameter file. None the less, the functionality of the tool is about the same as with Swarm and Repast. A significant difference is that NetLogo does not require the user the create a dedicated batch version of the model – at the price that by default NetLogo animates all displays and graphs during parameter space exploration, thus slowing down its execution. Lately turning off displays and graphs during batch runs has been introduced in NetLogo.

General-purpose, modeling package independent parameter space exploration toolkits also exist, like Drone and SweepOver. However, they also share most of the problems discussed above: they are typically highly technical, providing no easy-to-use user interface, but requiring a kind of programming. Moreover, they are suitable for exploring the regular regions of the parameter space (i.e. hyper-cubes and parallelepipeds), but not for adaptive branching, etc.

MODEL EXPLORATION MODULE

With the Model Exploration Module we want to provide a tool that makes the parameter space exploration experience as smooth and trouble-free, in other words user-friendly, as possible. In particular, modelers should be users and not programmers of this tool, who focus on the *modeling problem* and not on the technicalities. The goal with the MEME development is to provide easy means for the common parameter sweep tasks, while supporting model specific, complex task in addition. MEME offers easy-to-use graphical user interfaces for all its basic functions. The modeler can set up and run experiments from regular ABM models, execute batch runs on clusters of computers, collect, organize, import, visualize and do basic statistics from data through the GUI without having to write a line of code.

On the other hand the software offers advanced users optional scripting when setting up the parameter sweeps or importing and organizing results for example. Scripting requires more technical expertise from the user of course, as mentioned before it is optional, and was introduced for modelers whose needs exceed MEME's rich functionality.

The design and development of the tool is done in a modular way, with the components communicating through well defined APIs, allowing for the seamless integration of new or improved functionality. We have just recently started the Beta testing of a processing plug-in that will allow the user to run simulations on local clusters of computers and a number of existing grid solutions. We hope that this development will further reduce execution times, hence speed modeling work up without requiring the modeler to develop any special grid knowledge or skills.

Experiment Design

Agent based simulation is usually related to studying the behavior of certain real phenomena. In order to do scientific experiments various factors of long and large number of unattended simulations have to be observed. The simulation runs are executed on parameter spaces that are pre-defined or better yet, as discussed latter on, intelligently identified during the runs.

MEME currently offers solutions for designing experiments from models written in the popular [1]Repast environment and FABLES – a simple modeling language and its integrated modeling environment also by AITIA, see Gulyas and Bartha (2005) – without any additional programming. MEME offers a graphical user interface that fully assists the modeler in the process of setting up the experiments in a user-friendly way.

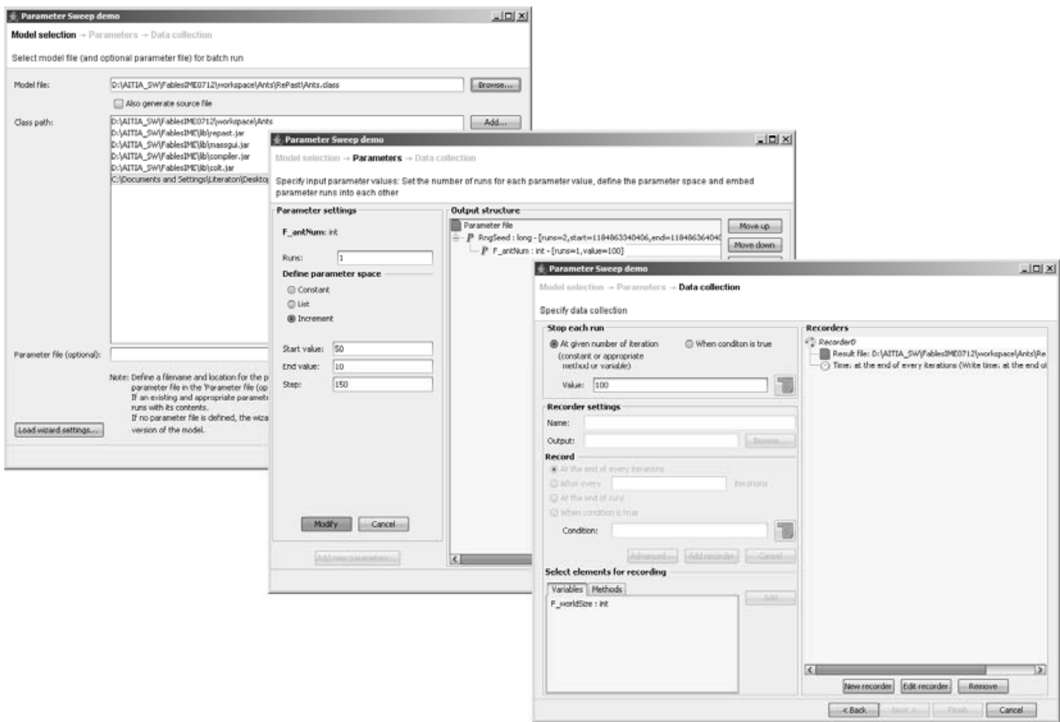


FIGURE 2 The parameter sweep wizard: selecting the model and required packages, setting up the parameter space and the data recording attributes.

Regular (i.e. not explicitly batch version) Repast and FABLES models can be imported into the application through a wizard that automatically identifies whether and which additional classes and jars that are needed to run the simulations. The same wizard explores the model for input and output parameters, lets the user create additional parameters if desired and provides the interface for declaring a parameter space for the experiment. The interface also offers capabilities to pick/add the variables whose values will be recorded and allows the user to define other data recording attributes. Measurements can be done at the end of simulations or at any desired tick count and in all cases measurements can record the actual value of the given variable, or the result of simple statistic operations. For advanced users MEME also offers scripting for declaring

[1] The tool is also developed to be able to deal with other Java-based modeling languages in the future.

entirely new variables that are results of various operations done on one or more of the original measurements.

Upon completing these tasks a new model and the corresponding parameter file are automatically generated from the original model with all the additional code necessary for running the simulation with the desired settings either on a local machine or to be distributed set as a set of simulation runs on a pool of available hosting computers.

Experiment Execution

The models and parameter files resulting from execution are forwarded to the respective engines for running Repast and Fables models in batch mode. In this phase of development, only 'brute-force' exploration of regular subspaces of the parameter space is supported. The engines are capable to run in a detachable, 'head-less' fashion. That is, after the GUI wizards have set up the exploration schemes, the engines require no user interactions and are capable to run in the background as separate applications, while providing means for the user to observe the incoming results and oversee the general state of the experiment while executing a high number of simulations.

Depending on the projected processing time of the batch runs the user has the option of launching the simulations on a single computer, a local or a remote cluster of computers. The module that distributes the simulation runs on the available computers was developed with extreme care in regard to collecting and storing the results of distributed runs. The module is prepared to re-instantiate runs to different computer once a host becomes unavailable or corrupted results are returned. As various systems with similar (albeit not simulation tailored) capabilities are already available, we did not intend to develop our own grid solution. To the contrary, we have and are developing a grid module that is capable of running simulations on an extendable list of distributed environments.

MEME currently is able to set up and run simulations with Inria's open source ProActive solution *(Caromel et al. 2006)* and the market leading Platform LSF family. These solutions enable the modeler to execute sets of simulation runs on clusters of computers running on [2]Windows or Linux/Unix operating systems, using both shared and separated storage.

Storing, Organizing and Visualizing Simulation Data

The MEME stores simulation results in a database, that includes all fix (constant) and changing parameters, and various additional information about the model (i.e. name, version, description, etc.). The software has a built-in Java-based database engine that can manage databases up to 8 GB in size, but the system is built in a way that it is independent of the

[2] Note, that ProActive is an SSH-based grid and as Windows platforms do not have a general SSH solution, preparing a Windows cluster for MEME carries some additional challenges.

particular SQL engine used, it supports professional database engines through the JDBC protocol. The program organizes raw data in a 3-level hierarchy (model, version and batch), the database-structure is created to be able to handle repeated exploratory runs, iterative, gradual import of results (i.e. new parameters being introduced and old ones deleted between versions of the same model).

MEME can obtain simulation results in two general forms. Results from batches of simulations designed and ran through MEME are automatically acquired. The other option is running simulations separately and then importing Repast result files or CSV files into the database. The program supports multiple file import, organizing data into different versions or batches under the same model. MEME distinguishes between input (i.e. values that do not change in during a run) and output parameters in result files. Various import settings can be saved if needed for later use, when importing data from a high number of simulations.

An important feature of model exploration is the processing of the available results. From the results obtained and stored in the database, subsets can be created. These computed tables, that we call 'views', can be described as tables, where variables (including parameters and measured variables) are in the columns and particular value-combinations are in the individual rows. Creating basic views for visualization and/or to be imported into other applications for more sophisticated analysis includes selection of the variables to be included from the results, plus specifying what model version(s) the variables be originated from.

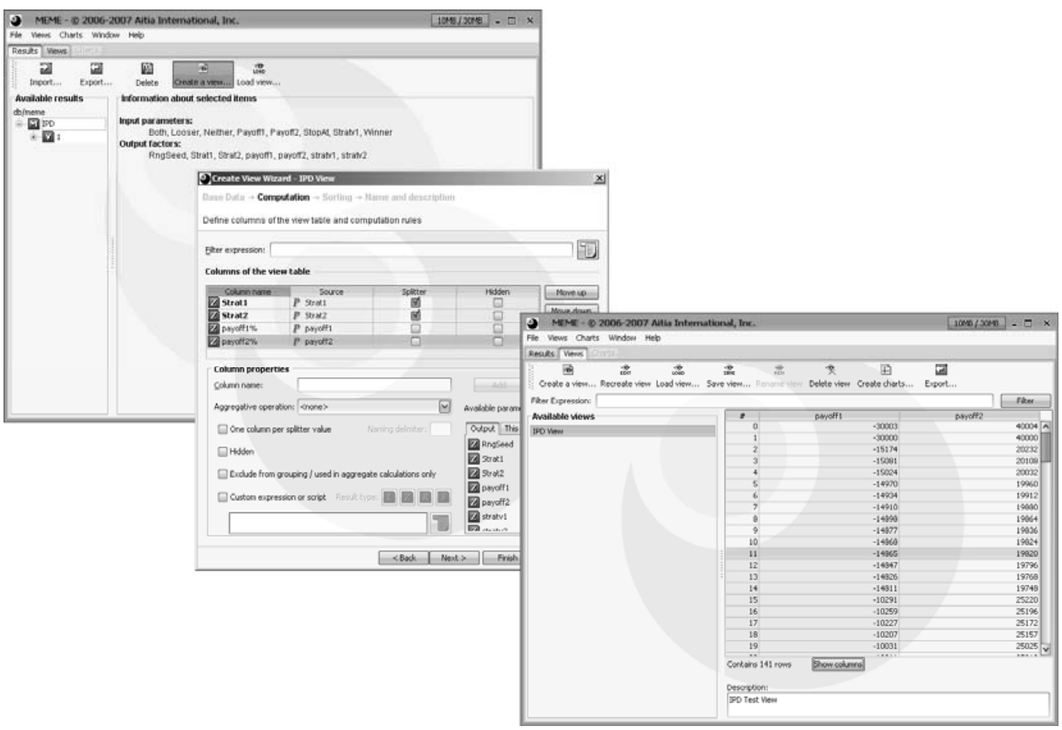


FIGURE 3 Creating a subset (views) of the original result data is done through a wizard. The modeler is presented with various filtering and computational options.

MEME provides the modeler with advanced functionality in creating subsets of the results database though. Conditional filtering, splitting (a generalized form of cross tabulation), aggregating and reorganizing data are supported, as well as the creation of derived statistics and custom computations typical of agent-based simulation, without any or with minimal coding. View creation settings can be saved as xml files for later use.

Once the data is organized into the desired form it can be exported for analysis in advanced statistical software of the user's choice, or it can be visualized through MEME's built-in Charting Wizard. The Charting Wizard is built on the Visualizations Package that is an ABM specific visualization tool developed by AITIA. It enables modelers to create visualizations for their simulations quickly and conveniently without any coding. The program offers visualizations[3] such as diagrams, grids and charts with both a vivid and rich design for presentation and a basic, black and white mode that suites scientific publication better.

Future Developments

MEME is not intended to be a sophisticated statistical tool. Although MEME already enables the user to export simulation results at any stage thus provide input as CSV files, further plans of development include implementing interoperation with such software.

Currently MEME is only capable to run parameter space explorations on a standard hyper-cube, thus allowing for interactive and incremental experiment design and execution. That is, a first set of runs is processed and their results are evaluated, which allows the user to determine the next set of runs. Ongoing developments include the introduction of intelligent parameter space exploration, where the explored sample of parameter space is irregular and is changing dynamically and adaptively.

The intellisweep capability is based on the meta-language we are currently working on, that describes what parameter combinations to explore next. The key to the workings of the intelligent parameter space exploration is the evaluation of results. Since this is always application dependent, general solutions are very limited. Therefore the advanced user is going to be provided with an integrated editor for developing call-back functions performing such evaluations in MEME.

Also general exploration routines are planned to be developed. The idea here is that the user defines a 'statement' about her results (i.e. the measured value converges/is always close to X, or depends linearly on Y, etc.). This is then converted into an evaluator that scores the measured value according its correspondence to the statement. The general exploration routines of MEME will then minimize this correspondence (maximize the error in the statement) using various and an extendable list of techniques (i.e. genetic algorithms, artificial neural networks, ant colony optimization, etc.) .

[3] List of currently available visualizations: Various 2D grids, bar charts, histograms, network and sub-graph visualizations, area-based chards, scatter plots, sequence visualizations and time series.

The Model Exploration Module is a modular system; all major components are decomposable and interchangeable. Hence in the future we would first like to extend MEME's import capabilities to reading result files from the MASON and JAS systems, then should demand arise we will also introduce parameter sweeps for the above mentioned, or any other Java-based modeling systems.

Acknowledgements

The work reported in this paper benefited from the partial support of the following grants:

- The GVOP-3.2.2-2004.07-005/3.0 (ELTE Informatics Cooperative Research and Education Center) by the Hungarian Government;

- The STREP project QosCosGrid (contract number 033883) by the European Commission;

- The AN FP6 Project Emergence in the Loop (EMIL) by the Information Society Technologies.

References

Axelrod, R. and Tesfatsion, L., 2006, "A Guide for Newcomers to Agent-Based Modeling in the Social Sciences", Appendix A in *Leigh Tesfatsion and Kenneth L. Judd (Eds.), Handbook of Computational Economics, Vol. 2: Agent-Based Computational Economics, Handbooks in Economics Series*, Elsevier/North-Holland, Amsterdam, the Netherlands.

Bankes, S. C. and Lempert, R., 2004, "Robust Reasoning With Agent-Based Modeling", *Nonlinear Dynamics, Psychology, and Life Sciences*, Vol. 8., Iss. 2, April, 2004, pp. 259-278.

Caromel, D., Delbé, C., di Costanzo, A. Ad Leyton, M., 2006, ProActive: an Integrated platform for programming and running applications on Grids and P"P systems, in *Computational Methods in Science and Technology*, Vol. 12., Iss 1, Polish Academy of Sciences, Poznan, Poland.

Gulyás, L., 2002, „On the Transition to Agent-Based Modeling: Implementation Strategies From Variables To Agents", *Social Science Computer Review*, Vol. 20, No. 4, Winter 2002, pp. 389-399.

Gulyás, L. and Bartha S., 2003, "How Much Is Too Much? -- What Programming Skills Are Really Needed to do ABM?" In *Proceedings of Seventh Annual Swarm Users/Researchers Conference*, Notre Dame, Indiana: Swarmfest 2003.

Gulyás, L. and Bartha S., 2005, “FABLES: A Functional Agent-Based Language for Simulations”, In *Proceedings of the Agent 2005 Conference on: Generative Social Processes, Models and Mechanisms*, Argonne National Laboratory, Chichago, IL, USA, October 2005.

Samuelson, D. and Macal, C., 2006, "Agent-based Simulation Comes of Age," *OR/MS Today*, Vol. 33, Number 4, pp. 34-38, Lionheart Publishing, Marietta, GA, USA.

Tatai, G., Gulyás, L., Laufer, L. and Iványi, M., 2005, “vBroker: Artificial Agents Helping to Stock Up on Knowledge”, In *Pechoucek, M., Petta, P. and Varga, L.Zs. (eds.). Multi-Agent Systems and Applications IV. Proceedings of the 4th International Central and Eastern European Conference on Multi-Agent Systems (CEEMAS 2005)*, pp. 336-345, Springer-Verlag.

Drone, http://drone.sourceforge.net/

NetLogo, http://ccl.northwestern.edu/netlogo/

Repast, http://repast.sourceforge.net/

Platform LSF, http://www.platform.com/Products/Platform.LSF.Family/Platform.LSF/

Swarm, http://www.swarm.org/wiki/Main_Page

SweepOver, http://www2.maths.ox.ac.uk/~challet/sweepover/sweepover.html

Friday, November 16, 2007

Computational Social Theory

Orientation and Action

MODELING COLLECTIVE COGNITIVE CONVERGENCE

H.V. PARUNAK,* T.C. BELDING, R. HILSCHER, S. BRUECKNER
NewVectors Division of TechTeam Government Solutions, Inc.

ABSTRACT

When the same set of people interact frequently with one another, they tend to think more and more along the same lines, a phenomenon we call "collective cognitive convergence" (C^3). In this paper, we discuss instances of this phenomenon and why it is advantageous or disadvantageous; review previous work in computational social science and evolutionary biology that sheds light on C^3; define a computational model for the convergence process and a quantitative metric that can be used to study it; report on experiments with this model and metric; and suggest how the insights from this model can inspire techniques for managing C^3.

Keywords: Groupthink, cognitive convergence, social simulation

INTRODUCTION

When the same set of people interact frequently with one another, they tend to think more and more along the same lines. We call this phenomenon "collective cognitive convergence" (C^3), since the dynamics of the collective lead to a convergence in cognitive orientation.

C^3 is seen in many different contexts, including research subdisciplines, political and religious associations, and even persistent adversarial configurations such as the cold war. Tools that support collaboration, such as blogging, wikis, and communal tagging, make it easier for people to find and interact with others who share their views, and thus may accelerate C^3. This efficiency is sometimes desirable, since it enables a group to reach consensus more quickly. For instance, in the academy, it enables coordinated research efforts that accelerate the growth of knowledge.

But convergence can go too far, and lead to collapse. It reduces the diversity of concepts to which the group is exposed and thus leaves the group vulnerable to unexpected changes in the environment. Here are two examples.

In academia, specialized tracks at conferences sometimes become unintelligible to those who are not specialists in the subject of a particular track, and papers that do not fit neatly into one or another subdiscipline face difficulty being accepted. The subdiscipline is increasingly sustained more by its own interests than by the contributions it can make to the broader community or to society at large.

* Corresponding author address: H. Van Dyke Parunak, NewVectors, 3520 Green Court, Suite 250, Ann Arbor, MI 48105; e-mail: van.parunak@newvectors.net.

In military operations, the force-on-force orientation developed during the Cold War left both the former Soviet Union and the United States ill-prepared to deal with insurgencies and asymmetric warfare.

Groups that have undergone cognitive collapse will only produce output conforming to their converged set of ideas, and will be unable to conceive or explore new ideas. In the worst case, collapse may lead a group to focus its attention on a cognitive construct with little or no relation to the real world. For example, highly specialized academic disciplines become increasingly irrelevant to people outside of their own circle.

We became interested in this phenomenon by observing the increasing balkanization of the research field of multi-agent systems.[1] Since we work in the area of multi-agent simulation, it occurred to us that some light might be shed on the phenomenon, and on how it can be managed, with a multi-agent model. This paper presents some preliminary results.

Section 2 discusses previous work related to our effort. Section 3 describes our model, and a metric that we use to quantify C^3. Section 4 outlines a series of experiments that exhibit the phenomenon and explore possible techniques for managing it. Section 5 suggests directions for further research, and Section 6 concludes.

PREVIOUS WORK

Our research on C3 builds on and extends two previous bodies of work, in computational social science and evolutionary biology.

In computational social science, our work merits comparison with Axelrod's adaptive culture model (Axelrod 1997) and its antecedents. What he calls "culture" corresponds to our notion of an agent's cognitive interests. Axelrod studied the transmission of cultural traits, represented as the elements of a numerical vector, between neighboring agents distributed on a 2-D lattice. Agents had a chance, proportional to their cultural similarity, of copying a trait from a neighbor. He found that a small number of large and stable homogeneous regions, or "cultures", would form. His work exhibits the emergence of disjoint regions of cultural (cognitive) homogeneity as agents interact with those who are adjacent to them spatially. Our model differs from his in several ways.

- His agents always interact with the same neighbors. Our agents can change their interaction partners as a result of the system's dynamics.

- His agents interact on the basis of spatial contiguity. Our model offers a much wider range of drivers for interaction.

- The nature of the interaction in his model is the same at every round. Our model modulates the strength of the interaction by the size and convergence of the emerging group.

---

[1] We are grateful to Simon Thompson for initial discussions that led to this project.

C3 can be considered a cultural analog of biological speciation, and so we look for insight to research in this field as well (see (Futuyma 1998) for a review). The most commonly proposed speciation mechanisms are allopatric speciation, sympatric speciation, and parapatric speciation. In allopatric speciation, genetic barriers gradually evolve between two or more geographically isolated species. This might happen for instance between organisms living on separate islands. These barriers could evolve either through natural selection or through other means such as the founder effect (i.e., differences in genes between populations due to the small sample sizes of the founding populations). This is analogous to different specialized communities developing in isolation from each other in C3. One specific type of natural selection that can cause speciation is sexual selection (Fisher 1930; Andersson 1994), a social process by which female mate choice influences the evolution of male traits. In extreme cases, this can become a runaway process that leads to extravagant features that are detrimental to survival (thus leading to a shorter lifetime and fewer opportunities to mate). Similarly, in C3 a social process can lead to the development of academic specializations with little practical relevance.

In parapatric speciation, there is no discrete barrier between populations; individuals are distributed along a geographic continuum and are separated by distance. Finally, sympatric speciation refers to instances where a single population with no gene flow barriers divides into separate species. While the relative importance and frequency of these speciation mechanisms in nature are still heavily debated, the mathematical prerequisites for each mechanism have been extensively studied; this work could be adapted to predict when and how C3 will develop, and how it can be prevented.

There has also been much theoretical work done to study the amount of gene flow or migration that is necessary to prevent isolated populations of organisms from diverging or losing diversity due to genetic drift, or sampling error (Hartl and Clark 1989). Sewall Wright argued in his Shifting Balance Theory that a subdivided population with intermittent migration could exhibit more rapid evolutionary change than a single cohesive breeding population (Provine 1986). The mathematical frameworks for studying migration could be applied to modeling the exchange of ideas or individuals between groups in C3, and the amount of exchange that is necessary to prevent intellectual isolation.

A MODEL AND METRIC

We have constructed a simple multi-agent model of C^3 to study this phenomenon. Our model represents each participant's interests as a binary vector. Each position in the vector corresponds to an atomic interest. A '1' at a position means that the participant is interested in that topic, while a '0' indicates a lack of interest. At each step, each participant

- identifies a neighborhood of other participants based on some criteria (which may include proximity between their interest vectors, geographical proximity, or proximity in a social network),
- learns from this neighborhood (by changing an interest j currently at 0 to 1 with probability $p_{interest}$ = proportion of neighbors having interest j set to 1), and
- forgets (by turning off an interest j currently at 1 to 0 with probability $1 - p_{interest}$).

One boundary condition requires attention. If an agent has no neighbors, what should $p_{interest}$ be? We take the view that interests are fundamentally social constructs, persisting only when maintained. Thus an isolated agent will eventually lose interest in everything, and in our model, a null community leads to $p_{interest} = 0$ for all interests. Alternative assumptions are certainly possible, and would lead to a different model.

We need a quantitative measure of agent convergence to study C^3 systematically. To derive our measure, we cluster the population hierarchically based on cognitive distance between agents (in our case, the Jaccard distance between their interest vectors). Each node of the resulting cladogram forms at a specific distance (the "diameter" of the cluster represented by that node). The root has the highest diameter. In a random population of agents, the distances at which lower-level nodes join the tree is not much less than the diameter of the root (Figure 1), while in highly converged populations, the diameters of lower-level nodes are much less than the diameter at the root (Figure 2, where agents grouped at diameter 0 have identical interest vectors). Thus we compute the ratio of node diameter to root diameter (the "min-max ratio") for each node, and use the median of this ratio as a measure of overall system convergence. A ratio of 0 (as in Figure 1) means that more than half of the agents belong to groups within which all interest vectors are identical.

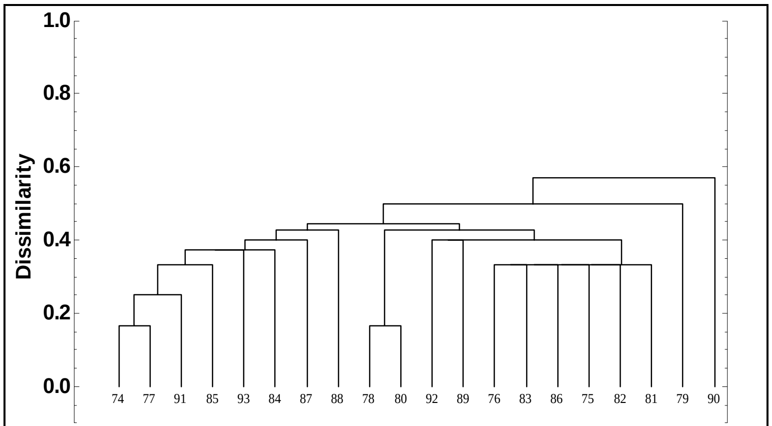


Figure 1 Cladogram of random interest vectors. The median ratio of the dissimilarity at which a node joins the tree to the dissimilarity of the root is 0.583.

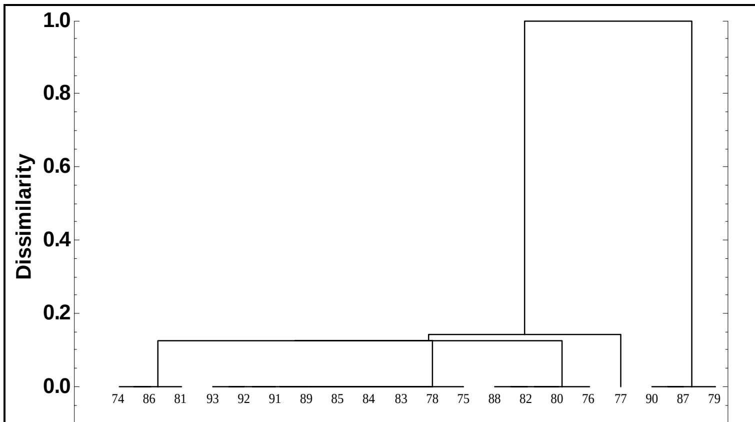


Figure 2 A highly converged population, whose median min-max ratio is 0

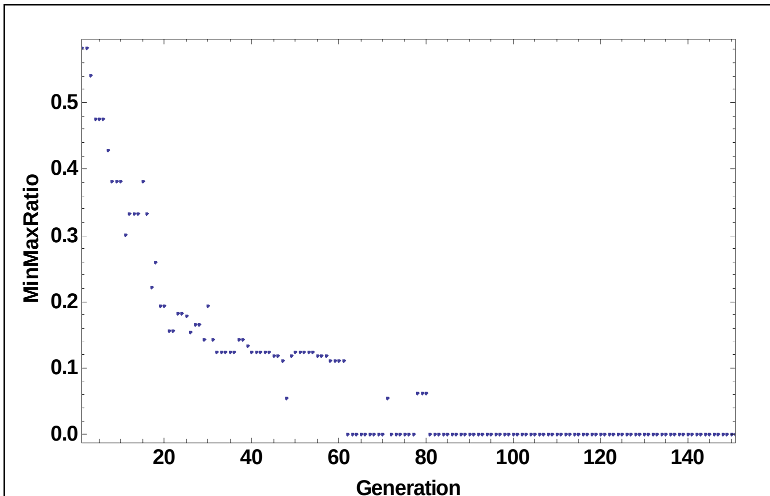


Figure 3 Evolution of 20 agents with length-10 interest vectors, neighborhoods defined by similarity > 0.5

Figure 3 shows the behavior of this measure over a sample run of the system with 20 agents and interest vectors of length 10, where the probability of learning and forgetting is equal, and where agents are considered to be in the same group if the similarity between their interest vectors (the similarity threshold) is greater than 0.5. It takes only about 80 generations for the median min-max ratio to reach 0. (A generation consists of selecting one agent, choosing its neighbors, choosing with equal probability whether it shall attempt to learn or forget, selecting a bit in its interest string at random, then if it is learning and the bit is 0, flipping the bit with probability p_{learn} * $p_{interest}$, or if it is forgetting and the bit is on, flipping the bit with probability p_{forget} * (1 $p_{interest}$.)) Figure 2 shows the state of this system at generation 300. By generation 370 it has collapsed into two groups of completely homogeneous agents of sizes 3 and 17 respectively.

SOME EXPERIMENTS

Armed with this model and metric, we can explore the dynamics of C^3 under a variety of circumstances. As we might expect, forming neighborhoods based on similarity of interest leads to rapid cognitive convergence. But surprisingly, other sorts of neighborhoods also lead to convergence.

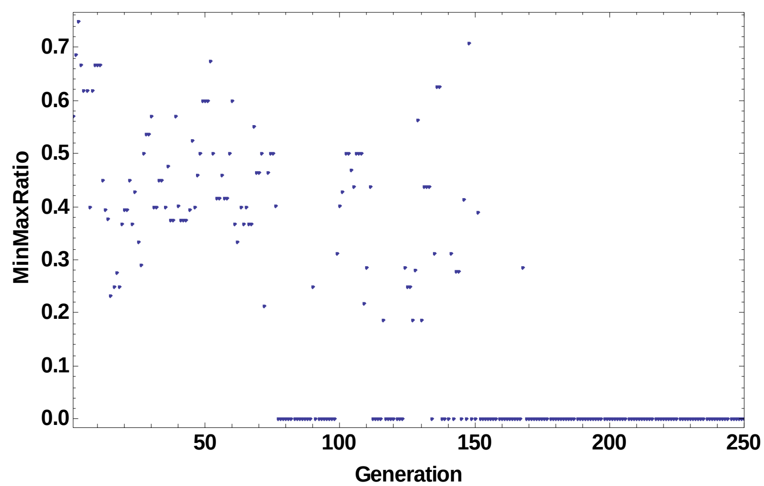


Figure 4 Evolution with similarity threshold = 0

Things that Don't Work

We might think that highly tolerant agents, those that consider all agents their neighbors, might be more robust to convergence. Figure 4 shows the evolution of the same population of agents when two agents consider one another neighbors if their similarity is greater than 0 (that is, they have at least one bit position in common). This configuration might be a model for a conference that has only plenary sessions. The population still collapses.

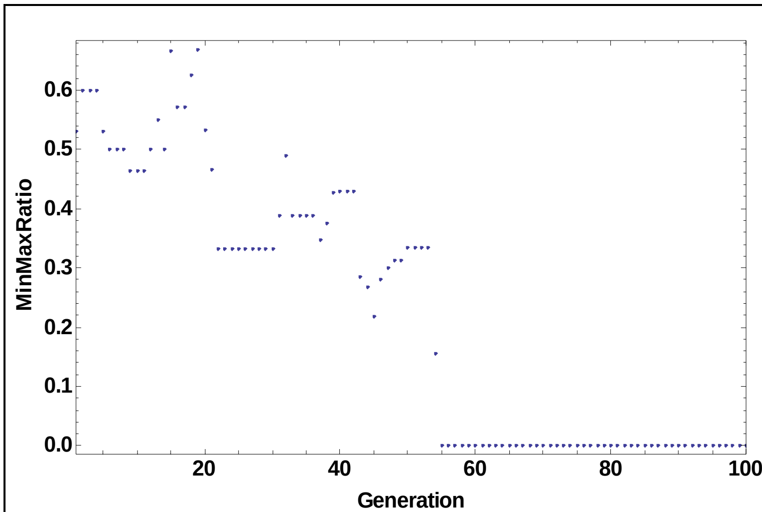


Figure 5 Fixed-size agent neighborhoods (four closest agents)

Perhaps the problem is that as agents converge, their neighborhoods increase in size. Figure 5 shows the effect of defining an agent's neighborhood at each turn as the group of four other agents that are closest to it. This configuration models a conference with separate tracks. Though agents base their adaptation at each turn on only 20% of the other agents, the population still collapses.

Figure 6 shows an even more radical approach. Here an agent's neighbors at each step are four randomly chosen agents. Imagine a conference at which papers are assigned to tracks, not by topic, but randomly. In spite of the mixing that this random selection provides, the population again collapses.

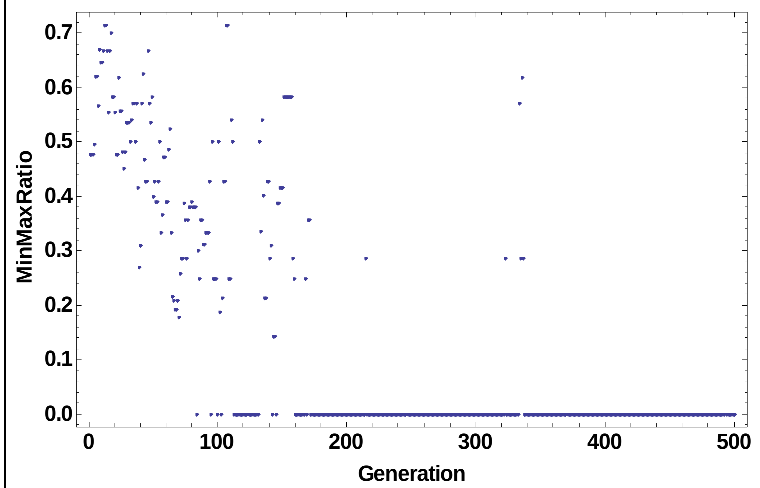


Figure 6 Neighborhoods of four randomly-chosen agents

These figures differ in how long it takes the system to converge to a min-max ratio of 0. The time to convergence is highly variable, even within a single configuration. Repeated runs show that we should not assume that because (say) Figure 5 converges faster than Figure 4, small groups will always lead to faster convergence than highly tolerant agents. The one constant across all runs is that the system does converge, in fewer than 500 generations (often far fewer).

Introducing Variation

The collapse of agent interests is due to the lack of any mechanism for introducing variation. Once the population loses the variation among agents, it cannot regain it. We have explored three mechanisms for adding variation to the population: random mutation, curmudgeons, and interacting subpopulations.

The simplest approach is mutation. At each generation, with some small probability p_{mutate}, after learning or forgetting, the active agent selects a bit at random and flips it. This mechanism models spontaneous curiosity on the part of agents. Figure 7 shows an extended run with parameters the same as in Figure 3 (neighborhoods defined by a similarity threshold of 0.5), but with $p_{mutate} = 0.03$. Mutation is certainly able to reintroduce variation, but the level is critical. If mutation is too low (say, 1%), it

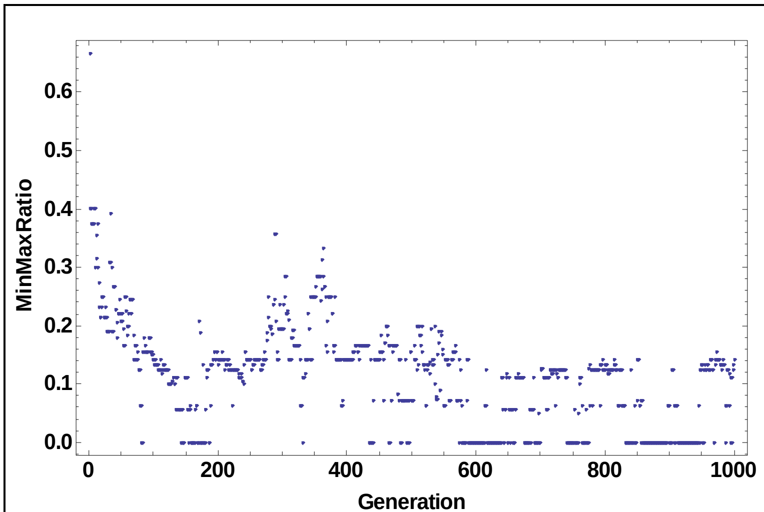


Figure 7 Adding 3% mutation

is unable to keep up with the pressure to convergence, while if it is too high (10%), the community does not exhibit any convergence at all (and in effect ceases to be a community). The nature of its contribution follows a clear pattern. When it is in the critical range, the system occasionally collapses to a min-max ratio of 0, but then discovers new ideas that reinvigorate it.

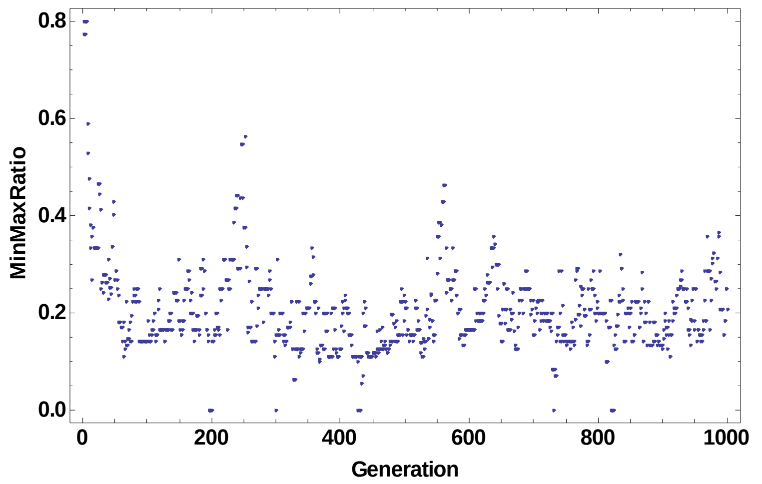


Figure 8 10% curmudgeons

A curmudgeon is a non-conformist, someone who regularly questions the group's norms and assumptions. Recall that ordinarily agents learn by flipping a 0 bit to 1 with probability $p_{interest}$, the proportion of neighbors that have the bit on, and forget by flipping a 1 bit with probability equal to $1 - p_{interest}$. To model curmudgeons, when an agent decides to learn or forget, with probability p_{cur}, it reverses these probabilities. That is, its probability of forgetting when it is curmudgeonly is $p_{interest}$ (instead of $1 - p_{interest}$ in the non-curmudgeonly state), and its probability of learning is $1 - p_{interest}$.

Figure 8 shows the effect of 10% curmudgeons, again with the baseline configuration of Figure 3. The system clearly converges, but seldom reaches a min-max ratio of 0. Furthermore, p_{cur} can achieve this balancing effect over a much wider range than p_{mutate}. As much as researchers may resent reviewers and discussants who "just don't get it," curmudgeons are an effective and robust way of keeping a community from collapsing.

The third source of variation is even more robust. So far, our agents have chosen a new set of neighbors at every step, based on their current set of interests. What would happen if we assign each agent to a fixed group at the outset, using a fixed similarity threshold that allows groups of various sizes to form?

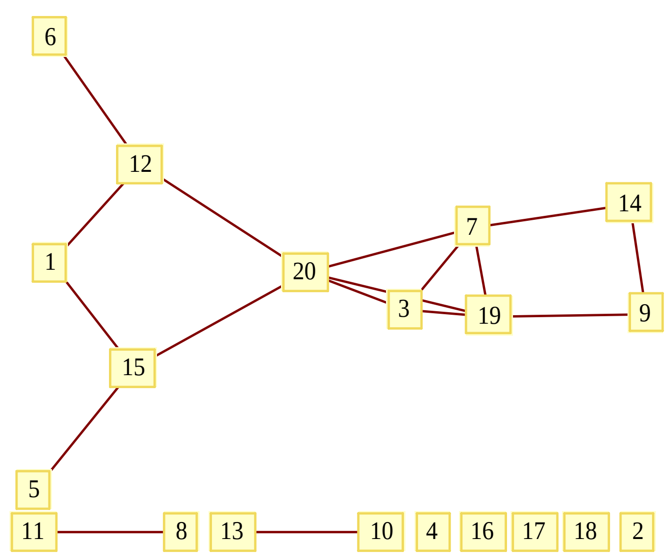


Figure 9 Neighborhood relations for threshold of 0.5

If the threshold is very high, each agent will initially be a group unto itself. With no neighbors to reinforce its interests, it will begin to forget them, and the agents will independently approach the fixed point of an all-zero interest string.

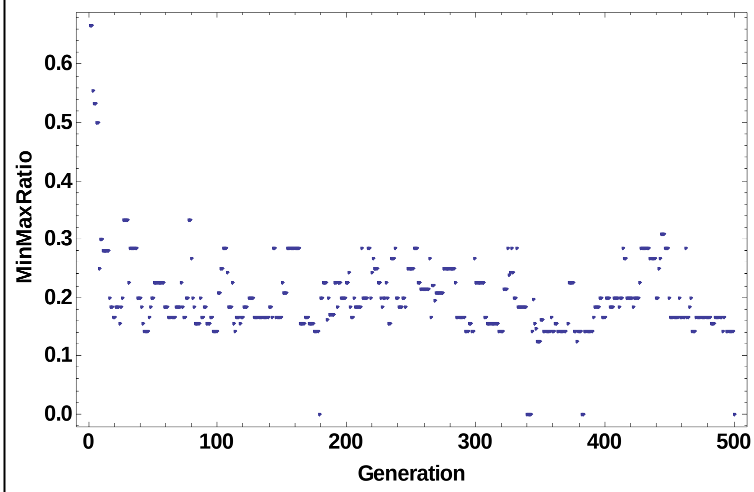


Figure 10 Fixed neighborhoods induced by threshold 0.5

If the threshold is very low, all agents will form one large group, and converge as in Figure 4.

For intermediate thresholds, the agents form a number of neighborhoods. Importantly, some agents ("bridging agents") belong to more than one neighborhood. Figure 9 is a graph of the agents, with an edge between two agents if those agents are neighbors of one another. Because neighborhoods are fixed over the run, each neighborhood can converge relatively independently of the others, but the bridging agents (in this case, notably agent 20) repeatedly displace each neighborhood's equilibrium with the emerging equilibrium of another group, a phenomenon noted by Page (Page 2007). As a result, the system shows convergence without collapse (Figure 10). This mechanism, like curmudgeons and unlike mutation, provides robustness against intermittent collapse. This system reflects a community with subdisciplines, but subdisciplines that recognize the value of members who bridge with other subdisciplines and exchange ideas between them. Such members are likely to be tolerated better by subgroups than would curmudgeons, because the source of the variation introduced by the bridging individuals is perceived as resulting from their multidisciplinary orientation rather than their orneriness

NEXT STEPS

Our simple model has shown a surprisingly rich space of behaviors. A number of directions for further work suggest themselves. For example:

- How can convergence be monitored in practice? Our metric, while effective for simulation, is impractical for monitoring actual groups of people. One might monitor the amount of jargon that a group uses, or lack of innovation, as indicators of convergence.

- We have suggested that convergence is a two-edged sword. What is the ideal degree of convergence, to allow the production of specialist knowledge without compromising the ability to escape collapse?

- How does convergence vary with group size? Recent work (Palla, Barabási et al. 2006) suggests that convergence in small groups requires specialized knowledge, while convergence in large groups requires a general knowledge base.

- We have assumed homogeneous tendencies to learn, forget, mutate, or behave curmudgeonly over all agents. How does the system respond if agents vary on these parameters? In particular, what is the impact of these parameters for bridging individuals in comparison with non-bridging individuals?

CONCLUSION

It is natural for groups of people to converge cognitively. This convergence facilitates mutual understanding and coordination, but if left unchecked can lead the group to collapse cognitively, becoming blind to viewpoints other than their own. Experiments with a simple agent-based model of this phenomenon show that seemingly obvious mechanisms do not check this tendency. In the domain of academic conferences, these well-intended mechanisms include plenary sessions, special tracks, or even random mixing. A source of variation must be introduced to counteract the natural tendency to converge. Mutation is effective if just the right amount is applied, but tends to let the system intermittently collapse. Curmudgeons are more robust, but socially distasteful. Perhaps the most desirable mechanism consists of bridge individuals who provide interaction between individually converging subpopulations.

Further understanding of C^3 could give important guidance in monitoring and managing collaboration. For example, consider a team of analysts searching for information.

- If a group's searches are sparsely distributed in search space, guide more analysts to join this group to cover more areas in this search space.
- If a group's searches are not specific enough, artificially promote the splitting of groups to create smaller, specialist groups (for example, by introducing specialists).
- If a certain convergence threshold is reached (perhaps because the search space has been exhausted), artificially introduce a curmudgeon to guide the group into a new area of the search space.
- If in a group only a few individuals drive convergence, artificially encourage less active individuals to participate more.
- If in a group the majority of people prevent the exploration of novel areas in search space, artificially encourage these people to be more adventurous.

REFERENCES

Andersson, M., 1994, *Sexual Selection*. Princeton, NJ, Princeton University Press.

Axelrod, R., 1997, "The Dissemination of Culture: A Model with Local Convergence and Global Polarization." *Journal of Conflict Resolution* **41**(2 (April)): 203-226. http://www-personal.umich.edu/~axe/research/Dissemination.pdf.

Fisher, R. A., 1930, *The Genetical Theory of Natural Selection*. Oxford, UK, Clarendon Press.

Futuyma, D. J., 1998, *Evolutionary Biology*. Sundarlund, MA, Sinauer.

Hartl, D. L. and A. G. Clark, 1989, *Principles of Population Genetics*. Sunderland, MA, Sinauer.

Page, S. E., 2007, *The Difference: How the Power of Diversity Creates Better Groups, Firms, Schools, and Societies*. Princeton, NJ, Princeton University Press.

Palla, G., A.-L. Barabási, et al., 2006, "Quantifying social group evolution." *Nature* **446**: 664-667. http://arxiv.org/abs/0704.0744.

Provine, W. B., 1986, *Sewall Wright and Evolutionary Biology*. Chicago, University of Chicago Press.

MODELING SITUATED ABSTRACTION: ACTION COALESCENCE VIA MULTIDIMENSIONAL COHERENCE

D.L. SALLACH, Argonne National Laboratory and the University of Chicago

ABSTRACT

Situated social agents weigh dozens of priorities, each with its own complexities. Domains of interest are intertwined, and progress in one area either complements or conflicts with other priorities. Interpretive agents address these complexities through: (1) integrating cognitive complexities through the use of radial concepts, (2) recognizing the role of emotion in prioritizing alternatives and urgencies, (3) using Miller-range constraints to avoid oversimplified notions omniscience, and (4) constraining actions to 'moves' in multiple prototype games.

Situated agent orientations are dynamically grounded in pragmatic considerations as well as intertwined with internal and external priorities. HokiPoki is a situated abstraction designed to shape and focus strategic agent orientations. The design integrates four pragmatic pairs: (1) problem and solution, (2) dependence and power, (3) constraint and affordance, and (4) (agent) intent and effect. In this way, agents are empowered to address multiple facets of a situation in an exploratory, or even arbitrary, order. HokiPoki is open to the internal orientation of the agent as it evolves, but also to the communications and actions of other agents.

Keywords: Situated agents, pragmatic hermeneutics, ontology

INTRODUCTION

When domains lend themselves to formal representation, the clarity and inferential power that becomes available is rewarding, even seductive. Unfortunately, the social domain manifests complexities that conventional formalism has been unable to capture successfully. Both natural language and situated action are heavily dependent upon context and based upon interactive interpretation by multiple participants. This may pose the greatest challenge confronting the development of effective (as opposed to suggestive or reconstructive) social models.

In part, the problem could be viewed as an issue of semantics, except that it concerns not only communication but action as well. Thus, the domain is one of semiosis (Hoffmeyer 1993), broadly conceived. That said, the issues that confront social action are very similar to those recognized as semantic. Some of the key issues are summarized in Table 1.

TABLE 1 Semiotic complexities

Issue	Description
Ambiguity (Empson 1966 [1930])	Words and actions have a range of meanings and implications, and their sense may differ from person to person, setting to setting, and time to time.
Vocality	Multiple actors bring multiple interpretations, intentions, and actions to a setting.
Fluidity	Comprehending and responding coherently to a multivocal state means that both situation and actor orientation are in dynamic flux.
Emergence	In discourse, whether interpersonal or collective, meanings and purposes evolve over time. Terms of the discourse are not fixed and, therefore, not definitive.
Thin coherence (Sewell 2005)	Complex and dynamic processes require that, while coherence is necessary for effective communication and action, it is a situated, provisional achievement that must be maintained.

The major premise of the present paper is that, in interpretive settings, formalization and abstraction may not succeed if they are overly general. On the contrary, in order for them to play the same clarifying roles that they do in natural domains, it may be necessary to identify abstractions that are strongly and inherently situated; their application may vary from case to case. They need to take forms that can be applied ‘from the inside out.’ That is, the situatedness of such abstractions can be utilized to refine and apply the insights that they carry. The paper discusses situated abstractions and then introduces a new structure that may be useful in social modeling.

ABSTRACTION

The efficacy of abstractions depends on their ability to be mapped to domains of interest in a regular way.[1] However, if a social domain is erratically dynamic, volatile, and/or self-organizing, it is difficult to identify a formalism that can exogenously represent its emergent patterns. Indeed, such a domain may require the use of *endogenous* abstractions, abstractions that evolve via the interaction of multiple self-organizing processes.

This insight is not new in sociology. Weber (1978) articulated formal organization (bureaucracy) as an ideal type that can be conceptualized but does not exist empirically. Subsequent generations of researchers found that the formalities of organization are immersed in a milieu of informal interactions that shape the results of organizational processes. Stinchcombe (2001) documents the role of interleaved informality in a wide range of domains, including: (1) construction blueprints, (2) civil law and procedures, (3) the commodification and liquidification of residential mortgage pools, (4) the

[1] This is not, of course, to deny that abstractions hold inherent interest, or that domains that are steeped in abstraction, such as mathematics, do not discover abstractions, the utility of which are not obvious and may not be known for some time.

classification of aliens at border crossings, and (5) the stratification of scientific knowledge.

In each domain, the informal and unscripted interaction determines how and in what ways the relevant formalism is to be applied. Rules can be defined formally, but decisions must still be made about when and under what circumstances it is appropriate to apply them.

FRAMING INDEXICALITY

The Interpretive Agent initiative addresses such complexities in multiple ways (Sallach 2003), including through: (1) integrating cognitive complexities through the use of radial concepts, (2) recognizing the role of emotion in prioritizing alternatives and registering urgencies, (3) incorporating Miller-range constraints in order to avoid oversimplified notions of agent omniscience, and (4) constraining actions to ‘moves’ in one of several prototype games.

Each of these mechanisms can be regarded as a facet of bounded rationality. However, it is also necessary to define how situated agent orientations are framed and refined such that they are dynamically grounded in pragmatic considerations and, at the same time, linked to internal and external priorities.

PRAGMATIC DECISIONS

Humans can be conceived of as boundedly rational agents acting within a complex world. While many models of cognition, intelligence, and/or action selection focus on a single task, situated social agents weigh dozens of priorities, each of which has its own complexities, urgencies, and timetable, with the latter sometimes being recurrent or recursive. Further, actors frequently find that the domains of interest are (or can be) intertwined, and that progress in one area either complements objectives from another area or, alternatively, conflicts with other priorities.

From within the web of complexities, decisions must necessarily be made on a pragmatic basis. This insight is not a new one. Graham (1989; see also Sallach 2007) indicates that ancient Chinese ethical thought follows an ethical form that approximates to [a] syllogism, applicable directly to concrete situations.

In awareness from all viewpoints, spatial, temporal, [social], and personal, of everything relevant to the issue, I find myself moved toward X; overlooking something relevant, I find myself moved toward Y.

> *In which direction shall I let myself be moved?*
> Be aware of everything relevant to the issue.
> Therefore, let yourself be moved toward X.

The quasi-syllogism is suggestive, but how might the web of complexities be modeled? Historically, pragmatist philosophy has derived its orientation amidst situated complexities (Joas 1993; Mead 1934; Peirce 1992 [1898]). One of major considerations is the *context* that frames the relevant communication and/or action. In his overview of semiotic relations, Peirce (1955 [1897]) describes context as follows:

> A sign, or *representamen*, is something which stands to somebody for something in some respect or capacity. It addresses somebody, that is, creates in the mind of that person an equivalent sign, or perhaps a more developed sign. That sign which it creates I call the *interpretant* of the first sign. The sign stands for something, its *object*. It stands for that object, not in all respects, but in reference to a sort of idea, which I have sometimes called the *ground* of the representamen, …

In this formulation, the ground defines the context of a semiotic process. Peirce continues (1955), stating that the task of the ground is "to ascertain what must be true of representamen… in order that they may embody any meaning." Drawing upon Wittgenstein (1958), Sheriff (1989) describes the ground as the 'language game' that the actor is playing.

Peirce and Wittgenstein have each made important contributions to the 'linguistic turn' of the twentieth century, and these have been brought into sharper focus by lines of research they have inspired. In a formulation that draws upon both of their contributions, and also follows the lead of Silverstein (2003), Blommaert (2005) introduces the concept of "orders of indexicality." This concept combines the idea that linguistic concepts and signs are ordered and that they occur in stratified complexes. The model construction of situated abstractions assumes a context defined by *orders of indexicality*.

Peirce's other semiotic elements (representamen, object, and interpretant) have also been deepened by subsequent work. In particular, the hermeneutic tradition (Gadamer 1989; Palmer 1969), which emphasizes the reciprocal dependency between the whole and the part has provided a new approach to the interpretive process. Shalin (2007) has developed an embodied semiotics that incorporates symbolic discursive, somatic-affective, and behavioral-component components, each with a particular relationship between signs and their objects. The resulting rich hermeneutic process has the potential to provide an effective focus for computational models of interpretation.

Now consider a social actor immersed in a wide range of relationships, opportunities, persistent and situated purposes, etc. Suppose that these salient social objects are represented using a Miller range of situated games (Sallach 2006b). Each game provides a means of expressing problems and possible solutions, sources of power and dependence, constraints and affordances, and possible actions and likely consequences. Since these pairs (which may not be exhaustive) (1) are intrinsically complex, (2) must be mapped by the actor to the relevant domain and the situated particulars of that domain, and (3) have the potential to dynamically change each other's state, there is clearly a need for an endogenous coherence-seeking process so that such games can be played with greater or lesser effectiveness. Designing a formalism to support such dynamics is a challenge, and it is to this task that we now turn.

HOKIPOKI

HokiPoki is a self-organizing model of situated abstraction designed to shape and focus agent orientations. The design integrates four pragmatic pairs: (1) problem and solution, (2) dependence and power, (3) constraint and affordance, and (4) intent and effect (agent action, as coalesced, lies between intent and effect) (see Figure 1). For each couplet, the first pair member tends to indicate current effect, while the second can often be seen in terms of prospective effects. As in the homophonous children's song, each of the four pairs can independently flip in and out of agent focus. In this way, the agent is empowered to address multiple facets of a situation in an exploratory, or even arbitrary, order.

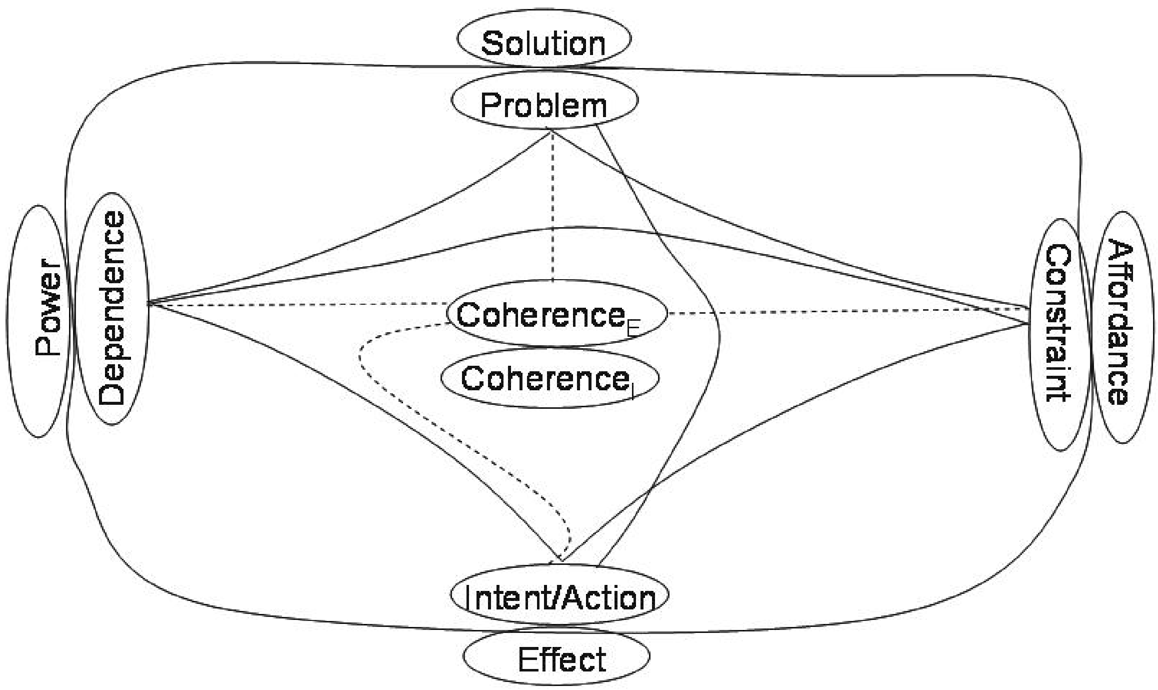


FIGURE 1 HokiPoki as a model of situated abstraction

Parenthetically, the designated pairs should not be viewed as definitive or exhaustive. On the contrary, the basic HokiPoki mechanism is designed to support exploration of diverse situated constructs, components that may be substituted exogenously by the designer or endogenously by agents within the model.

Notwithstanding the flexibility and expressiveness of the HokiPoki mechanism, a question remains as to how the multiple foci and priorities are to be integrated. The HokiPoki framework is guided by a dual internal/external coherence-calculation service that operates on lower-level ‘parts’ (cf., Sallach 2002; 2006a). The accomplishment of both internal and external coherence allows a Miller range of priorities to be addressed by toggling multiple current/prospective pragmatic pairs.

In application, each pragmatic pair will be defined relative to a particular problem domain. Each paired term is treated as a prototype concept (Sallach 2003) and grounded in a particular pragmatic domain. In interaction, agents are able to suggest (propose, etc.) alternate priorities, emotional valences, and/or conceptual structuring. The implication is that HokiPoki is open to the internal orientation of the agent as it evolves but also to the communications and actions of other agents. The agents engaged in such interaction are themselves represented within the orientation of each focal agent and, thus, their inputs will be integrated accordingly.

Implementations of the HokiPoki mechanism seem likely to benefit from the emergence of dynamic object-oriented programming languages such as Ruby and/or Groovy. Implementation considerations will be considered in greater depth during the presentation of this paper.

ARCHITECTURE AND MODELING ASSUMPTIONS

The HokiPoki architecture has two defining principles: (1) heterarchical, pragmatically focused self-organization and (2) openness to both internal and external interactions. The purpose of its framework is to allow these principles to be implemented in a flexible, responsive, and dynamic way.

The application of the HokiPoki model to a particular domain requires the structured coupling of components. Each of the pairs in the HokiPoki model is integral, where each component (problem and solution, power and dependencies, constraint and affordance, and intent/action and effect, respectively) is defined relative to its reciprocal. For most domains, pairs are likely to require stochastic components.

In addition, there are domain-specific relations between pairs. An affordance, if and when it appears, for example, may be part of a problem or a solution and will be situated and need to be specified. Depending on the domain, more detailed structural and/or data characterization may be necessary as well.

The integration of variegated dimensionality within a model has been described previously (Sallach 2007) and so will not be described here. Pair-wise operations and sequencing, however, will be described. On each turn, a pair examines both modes and determines which of the two will contribute to the greatest increase of coherence, a step which is then executed. Coherence-calculation is a service invoked by the activated pair. Exogenous changes may have modified overall coherence since the previous pair focus.

The progression from one pair to another may employ different rules, ranging from sequential to random order, to make-it/take-it, depending on the domain. Since HokiPoki has a problem-driven architecture, when the necessary solution(s) have been achieved (or the specified number of iterations have been completed), the process terminates.

DISCUSSION

The development of the HokiPoki model is an attempt to align social and computational realities in a different way. Social actors are immersed in complexities that bear directly or indirectly upon our condition. We must coherently organize communications and actions that improve, maintain, or (if nothing else) manage the deterioration of our circumstances. Our vantage point is from the midst of this field of propensities, and this field is flux. As a result, our orientation field must be immersed in complexity, and in flux, as well.

All of this is quite unwieldy from the perspective of most formal models. HokiPoki is an embryonic mechanism that supports the dynamic self-organization of agent orientation within complex and continually changing environments. It can be applied to many domains, which points to the work that lies ahead. The experience derived therefrom will provide the basis for further evolution of the mechanism.

ACKNOWLEDGMENTS

This work is supported by the U.S. Department of Energy, Office of Science, under contract number DE-AC02-06CH11357.

REFERENCES

Blommaert, Jan, 2005, *Discourse: A Critical Introduction*, New York: Cambridge University Press.

Empson, William, 1966 [1930], *Seven Types of Ambiguity*, New York: New Directions.

Gadamer, Hans-Georg, 1989, *Truth and Method*, New York: Continuum International Publishing Group.

Graham, A.C., 1989, *Disputers of the Tao: Philosophical Argument in Ancient China*, La Salle, IL: Open Court Press.

Hoffmeyer, Jesper, 1993, *Signs of Meaning in the Universe*, Translated by B.J. Haveland, Bloomington, IN: Indiana University Press.

Joas, Hans, 1993, *Pragmatism and Social Theory*, Chicago: University of Chicago Press.

Mead, George Herbert, 1934, *Mind, Self and Society*, Chicago: University of Chicago Press.

Palmer, Richard E., 1969, *Hermeneutics*, Evanston, IL: Northwestern University Press.

Peirce, Charles Sanders, 1955 [1897], "What is a sign? Three divisions of logic," pp. 98–101 in *Philosophical Writings of Peirce*, edited by J. Buchler, New York: Dover Publications.

—, 1992 [1898], *Reasoning and the Logic of Things*, Cambridge, MA: Harvard University Press.

Sallach, David L, 2002, "Situated social ecology: An integrated design hermeneutic," in *Agent 2002: Ecology, Exchange and Evolution*, edited by C. Macal and D. Sallach, Chicago: Argonne National Laboratory.

—, 2003, "Interpretive agents: Identifying principles, designing mechanisms," pp. 345–353 in *Agent 2003: Challenges in Social Simulation*, edited by C. Macal, M. North, and D. Sallach, Argonne: Argonne National Laboratory.

—, 2006a, "Coherence and interpretive agency: Mechanism design," in *North American Association for Computational Social and Organizational Science*, South Bend.

—, 2006b, "Complex multigames: Toward an ecology of information artifacts," pp. 185–190 in *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects*, edited by D.L. Sallach, C.M. Macal, and M.J. North, Chicago: Argonne National Laboratory.

—, 2007, "Logic for situated action," pp. 13–23 in *Advancing Social Simulation: The First World Congress on Social Simulation*, edited by S. Takahashi, D. Sallach, and J. Rouchier, Tokyo: Springer.

Sewell, William H., Jr., 2005, *Logics of History: Social Theory and Social Transformation*, Chicago: University of Chicago Press.

Shalin, Dmitri N., 2007, "Signing in the flesh: Notes on pragmatist hermeneutics," *Sociological Theory* 25:193–224.

Sheriff, John K., 1989, *The Fate of Meaning: Charles Peirce, Structuralism and Literature*, Princeton, NJ: Princeton University Press.

Silverstein, Michael, 2003, "Indexical order and the dialectics of sociolinguistic life," *Language and Communication* 23:193–229.

Stinchcombe, Arthur L., 2001, *When Formality Works: Authority and Abstraction in Law and Organizations*, Chicago: University of Chicago Press.

Weber, Max, 1978, *Economy and Society: An Outline of Interpretive Sociology*, Berkeley, CA: University of California Press.

Wittgenstein, Ludwig, 1958, *Philosophical Investigations*, translated by G.E.M. Anscombe, Upper Saddle River, NJ: Prentice-Hall.

NEXUS: AN INTELLIGENT AGENT MODEL OF SUPPORT BETWEEN SOCIAL GROUPS

D. DUONG, * US Office of the Secretary of Defense
R. MARLING, Marine Corps Combat Development Command
L. MURPHY, US Office of the Secretary of Defense
J. JOHNSON, US Office of the Secretary of Defense
M. OTTENBERG, US Office of the Secretary of Defense
B SHELDON, Marine Corps Combat Development Command
S STEPHENS, Marine Corps Combat Development Command

ABSTRACT

Nexus is an intelligent agent model focused on the support of social groups and organizations for each other and blame for departure from a social contract as evidenced by actions. It is a model of the attribution of blame for events based on trustworthiness. Agents representing leaders of groups look for breech of contract and keep track of a network of supporters. They choose whom to support depending on interpretations of past events. They perceive events depending on trust, reinterpreting past events in light of the present and visa versa. Nexus uses a model of the Boltzmann machine neural network, for the mind of each agent. It is based on interpretive social science, and the narrative paradigm in particular. Results are presented on a study of an insurgency using data collected from subject matter experts. This data includes support levels of ten relevant segments of population for each other, ideological similarity between groups, relevant historical events, and how groups might react towards the government if the US comes in to help during a natural disaster. The point of the simulation is to predict how disruptive the US aid would be if military personnel took an active role. The output data is the level of support for every group for every other group. It was found that, given historical events, a direct action by the US government only cause one group to like the insurgency a little more than they would have had the US government chosen an indirect approach to disaster relief..

Keywords: Interpretive Social Science, Narrative Paradigm, Boltzmann Machine, Constraint Satisfaction, Social Simulation, Neural Network, Irregular Warfare, Insurgency

---

* Corresponding author address: Deborah Duong, OSD/PA&E SAC1401 Wilson Blvd., Suite 300, Arlington, VA22209. email: Debbie.duong.ctr@osd.mil

INTRODUCTION

Nexus is an agent based model designed to simulate scenarios of irregular warfare. It is based on the narrative paradigm (Fisher), as agents look to relevant historical actions, current support networks, and ideological closeness to create a coherent view which calculates support levels for other agents. Agents try to make a story that is coherent with their historical context, and in doing so, may minimize apparent facts that don't make sense with the rest of the story, in accordance with cognitive dissonance theory (Festinger). Nexus is used for irregular warfare because the emphasis on contract keeping can be used for divisive strategies of nonviolent conflict. It takes into account higher orders of support, so that present enemies that are potential supporters, or present supporters that are potential enemies, may be identified.

THE MIND OF AGENTS

The Boltzmann Machine

The Boltzmann machine, a variety of the Constraint Satisfaction genre of neural networks, is used to represent the agent's minds, one for each agent. The Boltzmann machine is good at representing interpretations and reinterpretations of evidence. For example, the Boltzmann machine can model the interpretation of a Necker cube with a face as either in the front of the cube, or in the back of the cube, but as not both at the same time. It does this by making the belief that one vertice is in front evidence for believing or disbelieving that another vertice is in front (Simon). In Nexus, the Boltzmann machine is used to represent an evolving interpretation of evidence and blame, and its effect on levels of support. The paradigm shift, whether it be the shift that occurs when seeing a Necker cube in a different way, or the shift that occurs when facts are reinterpreted so that different parties are seen as responsible, is the same consonance seeking process. Constraint satisfaction networks have been used successfully to model how people see social situations (Duong and Reilly; Sallach; Thagard).

The basic agent of the Nexus model is the social group, which is a group of persons (whether they are organizations or not). Each social group has one Boltzmann machine that it uses to take all factors into account in its decision of whether it supports another social group.

The neurons in the minds of agents are of three types, as illustrated in figure 1.

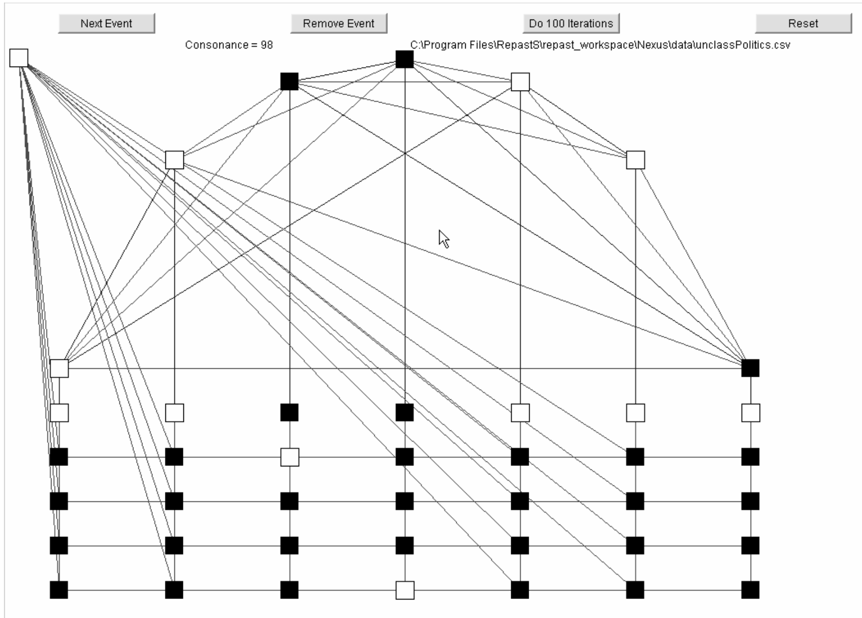


FIGURE 1 The Nexus GUI, of the neural mind of a single agent. The nodes along the top layer have been put into an arch so that their connections maybe seen. Connections are red for inhibitory, and blue for excitatory. The columns represent the social groups. The top layer contains the support nodes. The second layer (which does not have connections between nodes) contains the trust worthiness nodes. Each layer below that is for a single historical event, and contains blame nodes corresponding to the amount of blame a group is given for an event. To the left is an input node, that holds objective evidence for the blame of each group for events before the "spin" the mind will place on it.

The Nodes of the Network

1. Support : An node for the level of support for each other social group.
2. Trustworthiness: A node for how much each social group is perceived as a keeper of social contracts.
3. Blame: A node for the belief that this social group performed a particular event, for every (social group X event).

The Architecture of the Network

1. Support nodes. These nodes output whether the social group owning the net supports another social group or not. These nodes have mutual excitation with the nodes of groups that publicly support each other and mutual inhibition with the nodes of groups that publicly lack support for each other. The weights change in the network depending on changing public declarations of level of support between groups. The support nodes ensure that the groups that a group is supporting are taken into account in its decision of who to support (for example, the friend of my

friend is my friend, the enemy of my enemy is my friend), and enables agents to perceive accountability as something that is shared with their network of support.

2. Trustworthiness nodes. The trustworthiness node for a social group has a mutual inhibition with all of the blame nodes for that particular social group. That means, if a group performs an adverse action, it is not generally perceived as trustworthy, and if it is trustworthy, it does not tend to be perceived as performing adverse actions. Trustworthiness nodes have a mutual excitation with support node of the social group, meaning that if a social group is trusted it tends to be supported, and if it is supported it tends to be trusted. There is also an input node to the trustworthiness nodes that corresponds to ideological similarity, so that having a set of beliefs and agreed upon practices is taken into account into the estimation of whether a group is a contract keeper.

3. Blame nodes. There are (social group X event) blame nodes, with sets of blame nodes for individual events appearing in rows and for individual social groups appearing in columns. There is a constant excitation applied to each of the blame nodes through an input node. The blame nodes are lit in proportion to the degree of hard evidence for the fault of each social group, before the spin that the mind puts on it. The total energy the blame nodes for an event are lit is in proportion to the severity of the event. The constraint that an action tends to be performed primarily by one entity is expressed by negative inhibition between the blame nodes within a row that represents a single event. Because of this constraint, if an event is blamed on one party, it tends to let another party off the hook. The Boltzmann machine can measure cognitive dissonance, or the spin that the mind places on an event, pulling it away from where contrary evidence, through the calculation of "goodness" or consonance of the net. That is, even though hard evidence supports blaming one group for an event, relations of support and trust may cause another group to be blamed.

Blame for one event is connected to blame for another event only indirectly, through the trustworthiness node. If a new event is determined to be a group's fault, and the group's trustworthiness falls, then evidence in the past for events can be reinterpreted and blamed on that group even if they were blamed on another group before.

Running the Simulation

When an action happens, it has some blame attached to it that represents objective evidence that an event was caused by a group, in a magnitude that reflects how much the event hurt or helped the group doing the thinking. The leader thinks, taking into account the whole picture of all the groups behaviors, their affinities towards other groups, and then makes a new public declaration of support. This declaration increments the support level mappings in each leader's mind, preparing the leader for the next action when it will think and declare new support levels.

EMERGENT SOCIAL PHENOMENA

Nexus is a simulation based on first principles, and from which many types of tactics of irregular warfare may be modeled, including those discussed in Ackerman and Duvall's book, *A Force More Powerful*. For example, Nexus can model the fact that a group has to worry about the upholding of an ideology with its peers. Gandhi's revolution from India worked because Britain had to pay attention to its trustworthiness in keeping its ideology with international players. They had to worry about appearing hypocritical. Countries with ideologies that rationalize violence, whose allies support the same ideas, such as Nazi Germany, may not be so afraid of their reputation, and crackdown on a protest. A group could think that its peers would not support it if it did not uphold the binding ideology as a social contract between itself and other players. In the case of India, knowledge of the ideological break with an innocent, non enemy power, would affect the reputation of the British in the model, forcing it to keep an ideologically correct social contract with India as well. In Nexus, every agent has a model of its perceptions of the support networks of other agents, and making knowledge of ideological breaks public through a non violent warfare campaign, affects support levels as agents worry about the keeping the trust of their allies.

As in irregular warfare tactics, the network may be manipulated to separate a regime from its supporters. For example, if the analyst inputs into Nexus an IO campaign of adverse events to be blamed on the police (such as “Rodney King” style videos) , then the regime may break their support of the police, to keep the support of the citizens, but at the same time causing services to not be delivered. To input this into the model, the blame nodes would be lit against a police group for the event. This may cause the regime to cut ties with the police, creating another adverse event that the police would blame the government for.

The support for groups of similar ideology and ethnicity would tend to affect trustworthiness of a group when they break contract with particular groups and ethnicities. It may affect those with similar ideology more than those less similar, as in social theories which stress the importance of empathy in the success of irregular warfare. If groups tend to judge other groups by the same standards, and have the same opinions, it causes them to have similar friends and enemies and tend towards mutual support. “Cognitive Liberation” may be simulated by a change in the ideology by which one judges how adverse an event is. For example, to simulate nationalism in a majority ethnic group for securities sake in the face of a possible civil war with a minority ethnic group, the collateral damage caused by a violent insurrection would be seen as an adverse event, but the minority ethnic groups might not blame the insurgent group that caused the action directly, as much as the government, on the basis of similar support networks as well as an accumulation of blame for the government on adverse events in the past. However, if the government was kinder to the majority ethnic group in the past, they may put a spin on the interpretation of events against the insurgents.

EXAMPLE RUN

Nexus has been run on classified scenarios to study questions of irregular warfare at the Office of the Secretary of Defense. It has also been run on unclassified but sensitive scenarios at the Marine Corps Combat Development Command. The object of the Marine Corps scenario is to predict the effect that the greater presence of the United States would have on the support for the government and the insurgents in a country, in case it had to help that country during a natural disaster. The US could take a direct approach to assistance, or an indirect approach. The groups include displaced persons, the urban poor, the urban middle class, old money, illicit organizations, the police, the army, the church, the government, and the insurgency. Data was obtained through two subject matter experts (SMEs) that knew the details of the culture and the particular group’s history in a single province. SMEs estimated support levels, what those levels might be in both direct and indirect cases, and ideological similarities of the groups. The SMEs also described 22 historical events that were important to the groups in determining their present feelings for each other. These historical events are from different time periods, as cultures can consider something that happened thousands of years ago important. Historical events made up the cultural narrative and identity of the social groups. When Nexus was run, it was found that even though all groups except displaced persons were for the government and against the insurgents, the structure of their support for each other combined with historical events made the government somewhat vulnerable. All groups except the old money and the displaced persons changed their attitudes slightly more towards the insurgency when the US helped. It did not matter very much which kind of help. None of the groups, except for the urban middle class, had any different support levels for the government or the insurgents when direct and indirect action was compared. Only the urban middle class liked the insurgents a little bit more when the US action was direct than when it was indirect.

CONCLUSION

Nexus has been applied to real world scenarios of information operations and irregular warfare. It is one of the only tools that takes into account the historical consciousness of a people when explaining their actions. It is also unique in that it shows how new actions can influence a group to change their interpretations of the causes of their fortunes and misfortunes, and how these interpretations affect their alliances. Furthermore, it can reveal hidden vulnerabilities to changes in alliances due to higher orders of support levels and the entire historical picture of all parties. It has been suggested that Nexus will, in the future, be combined with Pythagoras for studies in stability operations.

REFERENCES

Ackerman P., and J. Duvall, 2001, *A Force More Powerful: A Century of Nonviolent Conflict,* New York: Macmillan.

Duong, D., and K. Reilly, 1995, “A System of IAC Neural Networks as the Basis for Self-Organization in a Sociological Dynamical System Simulation,” *Behavioral Science* Vol. 40, number 4, pp. 275-303.

Festinger, L., 1957, *A Theory of Cognitive Dissonance,* Stanford, CA: Stanford University Press.

Fisher, W., 1984, "Narration as Human Communication Paradigm: The Case of Public Moral Argument," *Communication Monographs 51*, pp. 1-22.

Sallach, D., 1988, “A Comparison of Parallel Architectures: Neural and Social Models of the Mind,” *Proceedings of the Second ACM Symposium on Artificial Intelligence,* Norman, OK.

Simon, Dennis. “The Boltzmann Machine Necker Cube Example,” http://www.cs.cf.ac.uk/Dave/JAVA/boltzman/Necker.html.

Thagard , P., 1999, *Coherence in Thought and Action*, Cambridge, MA: MIT Press.

HIGH-FIDELITY MATHEMATICAL MODELS OF SOCIAL SYSTEMS

Joe Jeffrey[1], Northern Illinois University, DeKalb, IL

ABSTRACT

We present a mathematical formalism, Entity Specifications, with sufficient rigor and expressive power to formalize a wide range of models of social systems. Entity Specifications are a combination of a rigorous formulation of the intuition of a frame and formal representation of relationships among constituents, as in mathematical logic. Entity Specifications are then used to formalize a particular conceptual framework that has been used as the basis for a number of computer systems, Communities and Intentional Action. This model articulates a broad range of social system phenomena, from individual actions to actions of groups at all levels, encompassing all types of phenomena actually encountered in human systems: biological, psychological, economic, sociological, political, and cultural. The formalism is then used to develop new mathematical formulations of concepts with broad applicability in the social sciences: complexity, similarity, and rate of change and rate of complexity change in social systems. The similarity measure is illustrated with an example in which the similarity between two pairs of intuitively similar families is calculated.

This paper presents a mathematical formalism for building high-fidelity models of the structure and dynamics of social systems. By "high-fidelity" we mean accurate representation of the entire range of the situations, processes, and events in the system, at every level of detail. The central intuition of the formalism is that a "thing" – an object, process, or state of affairs – is specified by giving a formal name, the logically necessary immediate constituents, and the relations between the constituents, all constituents and relationships also specified by formal name, as in mathematical logic. Constituents themselves may be further elaborated in the same way. The formalism is designed to handle incomplete knowledge and does not require reduction to atomic elements.

ENTITY SPECIFICATIONS

We need to be able to formally describe three kinds of "thing": objects (structures), processes (mechanisms), and states of affairs. Entities have parts – immediate constituents – that may be objects, processes, or states of affairs. An entity is described by giving its name and a description; the description consists of the names of the constituents and their type (object, process, or state of affairs), and their relationships.

Definition An *entity specification* (*ES*) consists of an ordered pair (N, D), where:

- N is the (formal) name of the entity including, optionally, a list of alternate names and/or a numerical ID.
- D is the set of *paradigms*, the major varieties or descriptions of the entity. In social systems, these are often structures or processes with little in common other than being

[1] H. Joel Jeffrey, Dept. of Computer Science, Northern Illinois University, DeKalb, IL 60115. Email: jeffrey@cs.niu.edu.

recognizable as varieties of the same thing. For example, in Western society celebrating a wedding anniversary has several paradigms: dining out, taking a cruise, buying a gift, etc.

Each paradigm of D is an ordered triple (C, R, E), where:

- C = $\{(C_{i,} T_i)\}$, in which C_i is the constituent and T_i is the constituent's classification, an element of the set {P, O, S}, representing "process,""object," or "state of affairs."
- R is the set of n-ary *relationships* that must hold between the named constituents. Any relationship may be named, not only those definable in terms of physical, mathematical, or computable quantities. (Equations are formal names.)
- The constituents and their relationships specify the structure of the entity. Additional information specifies particular instances of the entity. Identifying an actual instance requires the specification of which actual "things" (processes, objects, and states of affairs) that fill the roles named by the constituents. This information we term the *eligibilities* E for the entity: a set of ordered triples (c, i, r), in which
  - c is the name of the constituent;
  - i is the name of the individual;
  - r is the rule, or condition, under which i takes the role of c in this object.

ESs are a formalization and unification of the representation formats developed by P. G Ossorio[2], as a means of specifying objects, states of affairs, and processes at any level of detail.

A state of affairs' constituents may be any set of objects, processes, and other, smaller, states of affairs, in various relationships, i.e., any set C = $\{(C_{i,} T_i)\}$ and any set R, as defined above.

Processes

Processes are multi-step changes in objects and their relationships. Processes may occur in many ways, i.e., combinations of the steps. Therefore the $\{(C_{i,} T_i)\}$ for a process include:

1. Two constituents, the before-state and after-state.
2. A subset identifying stages, i.e., in which T_i = P. Some stages may be accomplished via two or more alternatives; these alternatives are included in this subset.
3. A subset identifying the objects, i.e., T_j = O.
4. A subset identifying the versions of the process. Each of these constituents C_k is a state of affairs, i.e., T_k = S, and the constituents of C_k are stages.

Relationships between stages specify the time relationships between them: sequential, parallel, overlapping, interspersed, etc.

Objects

Objects have only object constituents, and in that sense are simpler than entities in general or processes; each constituent of an object of Type O.

[2] P. G. Ossorio, *"What ActuallyHappens"*, University of South Carolina Press, Columbia, SC, 1978. Republished by Descriptive Psychology Press (www.descriptivepsychologypress.com), Ann Arbor, MI, 48104, in 2005.

Relationship to Frames

ESs are similar to frames, but differ in two important ways. First, ESs are considerably more rigorously defined. The constituents of an ES must be logically necessary for the thing being described to be what it is. By contrast, a frame simply represents “things commonly found together[3].” No distinction is made between things that are necessary to the definition of thing and those that are merely commonly present.

Second, frames do not include specification of the $R_1...R_m$ between constituents, although some frame-based systems allow specification of relationships. (Interestingly, while clearly a refinement of frames, Ossorio’s work predates the introduction of frames by several years[4].)

“Incomplete” Descriptions

Most real social systems are too complex to be specified completely, i.e., at the level of actions by individual persons. ESs are designed for handling incomplete specifications: a description and its elaborations are used in analysis or simulation by modeling the named entities and their interactions at that level. An ES set is not like a computer program, which is incomplete and not executable until all functions have been written in executable primitive statements.

For example, the state of affairs identified by the English sentence, “The rise of inflation in 1920’s Germany led to the rise of National Socialism,” identifies an entity (a state of affairs) with two constituents: “the rise of inflation in 1920’s Germany,” “National Socialism,” and the relationship “led to the rise of.” Each of these constituents has further Descriptions in terms of constituents and relationships, and a set of descriptions down to the level of the famous image of a woman with a wheelbarrow of Deutsche marks to buy a loaf of bread would be enormous. It is not, however, necessary for simulation or analytical treatment.

DESCRIBING SOCIAL SYSTEMS

Social systems, distinguished by having human “components,” require a conceptualization encompassing the specifically human aspects of action in a social context. The most comprehensive and systematic conceptualizations of this subject matter of which we are aware are the Community formulation, due to Putman[5] and Intentional Action, due to Ossorio[6]. They have been successfully used to build formal models of a number of social systems, and

---

[3] Minsky M (1975), “A Framework for Representing Knowledge”, in Winston P, ed., *The Psychology of Computer Vision*. New York: McGraw-Hill, NY.
[4] P. G. Ossorio, *State of Affairs Systems: Theory and Technique for Automatic Fact Analysis,* RADC-TR-71-102, Rome Air Development Center, 1971.
[5] A. O. Putman, “Communities,” in *Advances in Descriptive Psychology,* V. I, K. E. Davis ed., JAI Press, Greenwich, CT, 1981.
[6] P. G. Ossorio, *The Behavior of Persons,* Descriptive Psychology Press, Ann Arbor, MI, 48104, 2006.

software based on the models[7,8]. We illustrate using ESs to formalize social systems by formalizing Communities and Intentional Action. However, entity specifications are not particular to this conceptualization; formalizations of any model, no matter how abstract, physicalistic, or even fanciful, may be given.

A **Community** description is a 4-tuple <M, P, Cp, S>, where

- M (members) denotes all actual individuals in the community.
- P (Practices) denotes the set of social practices of the community. Practices encompass everything that a member of that community can do, *as* a member of that community.
- Cp (Choice principles) denotes the set of values or priorities specific to the community. The principles govern which practice is carried out and how, playing a large role in determining what actually occurs and how it occurs in the social system.
- S (Statuses) denotes the recognizable positions in the Community, whether formal and explicit or informal and implicit. “President,” “Senator,” “husband,” “child,” suicide bomber,” “respected leader,” “doctor,” “farmer,” etc., are examples. Each position has associated with it one or more *intrinsic practices*, practices a member engages in simply because they are in that position.

A practice is specified by giving a s*ocial practice description*, a quintuple <W, K, Kh, P, PC>, where

- W (want) denotes the goal.
- K (know) denotes the facts and concepts necessary for this action.
- Kh (know-how) denotes the skills needed to carry out the practice effectively.
- P (process) denotes the procedural aspect of the practice.
- PC (personal characteristics) denotes any relevant attitudes, traits, or abilities.

Finally, to specify the details of what happens in the system, the process parameter itself must be expanded with process entity description: the stages, constraints, eligibilities, and versions of the process.

What actually occurs is an *instantiated version* of the practice: a particular set of stages, with specific individuals filling the logical roles E, much as an actual production of *Hamlet* consists of the sequence of Act of the play, with actual persons and objects filling the roles of the characters and props.

To engage in an action is to engage in the practice of a community. The operation of the social system consists of members engaging in practices, in accordance with the choice principles of the community.

---

[7] “MENTOR: Replicating the Functioning of an Organization", in *Advances in Descriptive Psychology*, Vol. III, pp. 243-270, K. E. Davis, ed., JAI Press, Greenwich, Connecticut, 1983.

[8] Jeffrey, H. J., Schmid, T, Zeiger, H.P, and Putman, A. O., 1989, "LDS/UCC: Intelligent Control of the Loan Documentation Process" *Proceedings of the Second International Conference on Industrial & Engineering Applications of Artificial Intelligence and Expert Systems*, University of Tennessee Space Institute, Tullahoma, Tennessee, June, 1989, ACM Press, pp. 573-591.

ENTITY DESCRIPTIONS OF COMMUNITIES AND PRACTICES

Combining entity specifications with community descriptions gives a formal model of a social system. A community is an entity, with object, process, or event constituents: Members, a set of object names; Practices, a set of entity specications; Choice principles, a set of state of affairs entities; Statuses, a set of states of affairs entities.

Communities and intentional action, stated in Entity Specification form, provide a mathematical representation of human behavior in the human context, at any level of detail. A social system is a formal entity consisting of (formal) immediate constituents, with n-ary relationships between them, and elaboration of constituent entities, at any level, via entity specifications.

As noted above, the ES formalism may be used to formalize any model of human behavior; the community and practice model formulations are but one, albeit the most comprehensive and systematic we know of.

MEASURING SIMILARITY AND COMPLEXITY

Having a formalization of social systems allows us to give new mathematical formulations of the concepts of complexity of a social system and similarity of two social systems.

We first define the *structural complexity* of a social system A, with with N constituents $A_1, \ldots, A_N$ and K relationships, as:

$$SC(A) = \sqrt{N^2 + K^2 + \varepsilon \bullet \sum_{i=1}^{NA} SC(A_i)^2}$$

ε is an experimentally-determined multiplier modulating the impact of complexity of constituents, sub-constituents, etc. (Preliminary work indicates a value of approximately 0.7 for ε.)

We can now mathematically define the degree of similarity between any two entities based on their constituents and relationships. The definition is designed to correspond to the following intuitions:

1. The measure should take into account differences in attributes of the entities themselves.
2. The measure should take into account similarity of structure. Structure is formalized by relationships among constituents, in two ways: a) differences in the attributes of the constituents of A and B; b) if A and B have the same relationships among their respective constituents, but to different degrees, similarity should reflect the difference in degree; c) if A and B have different relationships, they should be less similar.

3. When the constituents of A and B themselves have ESs, the measure should recursively include the structural similarity of the constituents.

Accordingly, the distance between two entities is defined as follows:

1. Let A and B be any two entities, the properties of A and B be p_1, ..., p_M, and entity specifications comprised of constituents A_1, ..., A_{NA} and relationships r_1, ..., r_K, and B_1, ..., B_{NB} with the L relationships r_{K+1}, ..., r_{K+L}.

   The constituents A_1, ..., A_m with relationship r_j are ordered m-tuples. Denote the number of A-tuples by NAT, and the number of B-tuples by NBT.

2. Let the value of property i of a constituent be represented by $p_i()$, and $r_i(t)$ denote the value of the ordered tuple t of A- or B-constituents satisfying relationship r_k. For example, a strong love relationship between family members A_1 and A_2 is represented by loves(A_1, A_2) = 0.9 (on a 0 to 1 scale).

   Let P denote the matrix with M columns and NA+NB rows, whose values are the values of each property p_i.

   P:

	p_1	...	p_M
A_1			
...			
A_{NA}			
B_1			
...			
B_{NB}			

   The *property distance* between A_i and B_j is given by

   $$PD(A, B) = \sqrt{\sum_{i=1}^{M} (p_k(A_i) - p_k(B_j))^2}$$

   Let R be the matrix with K+L columns and NAT+NBT rows whose entries are the values $r_k(t)$. If a constituent does not have property p_i, or a tuple does not have relationship r_k, leave the corresponding entry of the matrix blank.

R:

	r_1	...	r_K	r_{K+1}	...	r_{K+L}
A-tuple_1						
...						
A-tuple_{NAT}						
B-tuple_1						
...						
B-tuple_{NBT}						

3. If any column of P or R contains a value < 0, re-scale the values of the column by adding the absolute value of the minimum value of the column to each value in it.

4. Normalize the values of P to the range 1 to 10, by setting

   $p_i(A_j) = 10 * (p_i(A_j) + 1) / pmax_i$), where $pmax_i$ is the maximum value of column i.

5. Set each empty entry of P to 0.

   The values of the property matrix P are now between 0 and 10, 0 indicating the component does not have the property of that column.

6. Similarly, normalize the values of R, the matrix with K+L columns representing relationship values for constituents of A and B, to the range 0 to 10.

7. When A and B have constituents, the similarity between A and B must include similarity of their constituents. That calculation is affected by the order of the constituents. For example, suppose A and B are organizations, and A has a large and complex marketing department and a small, simple shipping department, while B has a large and complex shipping department and small, simple marketing department. The calculated similarity between A and B will be quite different, depending on whether the two marketing departments and two shipping departments are compared, or whether A's marketing department is compared to B's shipping department. The constituents of each must be re-ordered so that the distance comparison has a consistent basis.

   Therefore, re-order the constituents of A and of B, from maximum SC (as defined above) to minimum.

8. The **distance between two entities A and B** is comprised of two components, the property distance and the structural distance:

   $$d(A, B) = \sqrt{PD(A, B)^2 + SD(A, B)^2}$$

   The *structural distance* SD(A, B) is defined recursively as follows:

   Let MC = max(NA, NB) and MT = max(NAT, NBT).

Then if both A and B have Descriptions, i.e., specified constituents and relationships, we define the structural distance SD as

$$SD(A, B) = \sqrt{(NA-NB)^2 + \sum_{i=1}^{MC} PD(A_i, B_{\kappa(i)})^2 + \sum_{j=1}^{MT} \sum_{i=1}^{L} (r_i(ta_j) - r_i(tb_{\kappa(j)}))^2 + \delta \bullet \sum_{i=1}^{MC} SD(A_i, B_i)^2}$$

where $B_{\kappa(i)}$ denotes the B-constituent closest to A_i, using Euclidean distance, and

$tb_{\kappa(j)}$ denotes the B-tuple closest to ta_j, using Euclidean distance between tuples.

If A NAT > NBT, $r_i(tb_j) = 0$ for NBT < j <= NAT, and similarly if NBT > NAT.

If NA > NB, then $d(A_i, B_i) = SC(A_i,)$, for NB < i <= NA, and similarly if NB > NA.

If either A or B have no Description, SD(A, B) = 0.

δ is an experimentally-determined discount factor reflecting the relative importance of the distance between constituents of A and B. (As with ε, preliminary work indicates a value of approximately 0.7 for δ.)

$PD(A_i, B_i)$ measures similarity of properties of each pair of constituents;

$\sum (r_i(ta_j) - r_i(tb_j))^2$ measures how much the constituents of A and B differ on relationship r_i; and the sum $\sum_{j=1}^{MT} \sum_{i=1}^{L} (r_i(ta_j) - r_i(tb_{\kappa(j)}))^2$ measures the total difference in structure between A and B, as articulated by the relationships r_i, $1 \le i \le L$.

The distance measure d(A, B) has the following properties:

- d(A, B) = 0 if A and B are the same except for differing only in names of constituents and relationships (mathematically, are isomorphic).
- The distance increases as the properties of A and B, the number of their constituents, the properties of the constituents, the structure of A and B, and the substructures of A and B diverge.

As with any mathematical definition intended to capture an intuition, this formulation must validated experimentally. This work is in progress.

Example: Structural Similarity of Two Families

Family A consists of a mother, father, and two children. The mother and father are married, and love each other. Both parents love both children; the children love each other. However, the children also compete with each other for success in school.

Family B consists of a mother, father, and three children. The mother and father are married. Both parents love all the children. The two younger children love each other, but both resent the eldest and compete with her for each parent's affection. The eldest child also has a significant responsibility in caring for the younger children

We suppose that the member attributes of interest in this case are age and health of the family members.

	Age	Health
M_A	40	0.8
F_A	42	0.7
AC_1	12	1.0
AC_2	10	1.0
M_B	35	0.9
F_B	36	0.8
BC_1	8	1.0
BC_2	6	1.0
BC_3	14	1.0

Table 1: P matrix for Families A and B

	Rom.. Love	Par. love	Sib. love	Aca. comp.	Resent	Affec. comp.	Care-taker
(M_A, F_A)	1.0						
(F_A, M_A)	1.0						
(M_A, AC_1)		1.0					
(M_A, AC_2)		1.0					
(F_A, AC_1)		1.0					
(F_A, AC_2)		1.0					
$(AC_1\ AC_2)$			1.0	1.0			
(AC_2, AC_1)			1.0	1.0			
(M_B, F_B)	1.0						
(F_B, M_B)	1.0						
(M_B, BC_1)		1.0					
(M_B, BC_2)		1.0					
(M_B, BC_3)		1.0					
(F_B, BC_1)		1.0					
(F_B, BC_2)		1.0					
(F_B, BC_3)		1.0					
(BC_1, BC_2)			1.0				
(BC_2, BC_1)			1.0				
(BC_1, BC_3)					1.0		
(BC_2, BC_3)					1.0		
(BC_1, BC_3)						1.0	
(BC_2, BC_3)						1.0	
(BC_3, BC_1)							1.0
(BC_3, BC_2)							1.0

Table 2: R matrix for Families A and B

The normalized values of the properties of the families A and B are (8.2, 10, 10) and (10, 10, 10), so PD(A, B) = $\sqrt{1.8^2 + (10\text{-}10)^2 + (10\text{-}10)^2}$ = 1.8.

$(NA–NB)^2 = (5\text{-}3)^2 = 4$.

The normalized property matrix P, with rows re-ordered so that the pairs A_i, $B_{\kappa(i)}$ are adjacent, is

	Age	Health
M_A	9.5	8
F_B	8.6	8
F_A	10.0	7
M_B	8.3	9
AC_1	2.9	10
BC_3	3.3	10
AC_2	2.4	10
BC_1	1.9	10
BC_2	1.4	10

Table 3: Normalized P for Families A and B

and the constituent property distance $\sum_{i=1}^{5} PD(A_i, B_{\kappa(i)})^2$ = 0.81+2.89+0.16+0.25+1.96

= 6.07

The normalized relationship matrix R, with rows re-ordered so that the pairs ta_j and the nearest tuple $tb_{\kappa(j)}$ are adjacent, is

	Rom.. Love	Par. love	Sib. love	Aca. comp.	Resent	Affec. comp.	Care-taker
(M_A, F_A)	10	0	0	0	0	0	0
(M_B, F_B)	10	0	0	0	0	0	0
(F_A, M_A)	10	0	0	0	0	0	0
(F_B, M_B)	10	0	0	0	0	0	0
(M_A, AC_1)	0	10	0	0	0	0	0
(M_B, BC_1)	0	10	0	0	0	0	0
(M_A, AC_2)	0	10	0	0	0	0	0
(M_B, BC_2)	0	10	0	0	0	0	0
(F_A, AC_1)	0	10	0	0	0	0	0
(M_B, BC_3)	0	10	0	0	0	0	0
(F_A, AC_2)	0	10	0	0	0	0	0
(F_B, BC_1)	0	10	0	0	0	0	0
$(AC_1\ AC_2)$	0	0	10	10	0	0	0
(BC_1, BC_2)	0	0	10	0	0	0	0
(AC_2, AC_1)	0	0	10	10	0	0	0
(BC_2, BC_1)	0	0	10	0	0	0	0
(F_B, BC_2)	0	10	0	0	0	0	0
(BC_1, BC_3)	0	0	0	0	10	0	0
(F_B, BC_3)	0	10	0	0	0	0	0
(BC_2, BC_3)	0	0	0	0	10	0	0
(BC_1, BC_3)	0	0	0	0	0	10	0
(BC_2, BC_3)	0	0	0	0	0	10	0
(BC_3, BC_1)	0	0	0	0	0	0	10
(BC_3, BC_2)	0	0	0	0	0	0	10

Table 4: Normalized R for Families A and B

The structural-difference sum $\sum_{j=1}^{MT} \sum_{i=1}^{L} (r_i(ta_j) - r_i(tb_{\kappa(j)}))^2 = \sqrt{6\bullet 0^2 + 2\bullet 10^2 + 8\bullet 10^2} = 31.6$

In this example, the immediate constituents are individual persons. Customarily one considers persons to be indivisible, so in this case $SD(A_i, B_i) = 0$. (In a case in which it is considered appropriate to model parts of a person, such as aspects of personality, $SD(A_i, B_i)$ may be non-zero.)

Thus $SD(A, B) = \sqrt{4 + 6.07 + 31.6}$ and $d(A, B) = \sqrt{1.8^2 + 4 + 6.07 + 31.6}$
$= 6.7$

Consider now the distance between A and B', a family identical to B except that $B'C_1$ and $B'C_2$ did not resent and compete for affection with $B'C_3$. The rows of R representing those relationships are now missing, so the structural-difference sum is 24.5, and d(A, B') = 6.1.

Rate of Social Change

Thus, we can now define rates of change in a social system:

The **rate of social change**, as a system goes from S_1 to S_2, in time Δt, is $\dfrac{d(S_1) - d(S_2)}{\Delta t}$

and the **rate of social complexity change** of S is $\dfrac{SC(S_1) - SC(S_2)}{\Delta t}$.

REFERENCES

Jeffrey, H. J., 1983, "MENTOR: Replicating the Functioning of an Organization", in *Advances in Descriptive Psychology*, Vol. III, pp. 243-270, K. E. Davis, ed., JAI Press, Greenwich, Connecticut.

Jeffrey, H. J., Schmid, T, Zeiger, H.P, and Putman, A. O., 1989, "LDS/UCC: Intelligent Control of the Loan Documentation Process," *Proceedings of the Second International Conference on Industrial & Engineering Applications of Artificial Intelligence and Expert Systems*, University of Tennessee Space Institute, Tullahoma, Tennessee, June, 1989, ACM Press, pp. 573-591.

Minsky, M, 1975, "A Framework for Representing Knowledge", in Winston P, ed., *The Psychology of Computer Vision*. New York: McGraw-Hill, NY.

Ossorio, P. G., 2006, *The Behavior of Persons*, Descriptive Psychology Press, Ann Arbor, MI 48104 (www.descriptivepsychologypress.com).

Ossorio, P. G., 1971, *State of Affairs Systems: Theory and Technique for Automatic Fact Analysis*, RADC-TR-71-102, Rome Air Development Center.

Ossorio, P. G., 2005, "*What Actually Happens*", University of South Carolina Press, Columbia, SC, 1978. Republished in by Descriptive Psychology Press, Ann Arbor, MI 48104. (www.descriptivepsychologypress.com).

Putman, A. O., 1981, "Communities," in *Advances in Descriptive Psychology*, V. I, K. E. Davis ed., JAI Press, Greenwich, CT.

Network Dynamics

THE DYNAMICS OF NETWORK-EFFECTS IN TWO-SIDED AND MULTI-SIDED MARKETS: AN AGENT-BASED APPROACH

W. GRANIGG,*
Martin-Luther-University Halle-Wittenberg, Germany

ABSTRACT

Two-sided and multi-sided markets and the dynamics of their network-effects have become an active research area in recent years. In this paper a general formulation of an agent-based duopoly-model of a two-sided market is presented that seems very promising to study and to analyze under which starting conditions two-sided markets tend (just because of the dynamics of the involved network-effects) to a winner-takes-all-situation and under which starting conditions both platforms can survive within the market. The results of the simulations of this model can be used for analysing real existing two-sided (and multi-sided) markets.

Keywords: Two-sided markets, multi-sided markets, network-effects, agent-based model

INTRODUCTION

Many modern markets try to tie together two or more distinct groups of individuals (users) in a network whereas the individuals of each group are interested in interacting with the individuals of the other group. These markets are called ‘two-sided’ or ‘multi-sided markets’ (depending on the number of different groups) and can roughly be defined as “markets in which one or several platforms enable interaction between end-users, and try to get the two (or multiple) sides ‘on board’ by appropriately charging each side” (Rochet/Tirole 2005).

If someone thinks about that definition, a lot of markets rapidly come into one’s mind: Newspapers for example have to attract readers and advertisers, videogame platforms have to attract consumers and software developers, TV networks have to attract viewers and advertisers, credit cards have to attract cardholders and merchants and so on (Rochet/Tirole 2005; Evans 2003). The actual importance of markets like these is quite enormous, since these markets have redefined and changed the global business landscape rapidly and fundamentally in the last few decades (Eisenmann/Parker/Van Alstyne 2006).

One interesting thing of two-sided and multi-sided markets is that frequently they cannot be analyzed by means of the traditional and well-known economic rules. Many of them are not working correctly in these markets any more and so the use of these traditional rules can sometimes lead to severe mistakes (and losses) (Wright 2004).

The key issues to understand the way how two-sided and multi-sided markets work seem to be the so-called ‘same-side’ and ‘cross-side’ network effects. Simply spoken, network effects

* Corresponding author address: Wolfgang Granigg, Institute of Economics, Department for Microeconomics and Public Economics, Martin-Luther-University Halle-Wittenberg, Universitätsring 3, 06108 Halle an der Saale, Germany; e-mail: wolfgang.granigg@wiwi.uni-halle.de.

in general describe the possibility that the derived utility for a user by the consumption of a certain good increases with the number of other users consuming the good as well (Katz/Shapiro 1985; Katz/Shapiro 1986; Tirole 2002). Since in two-sided and multi-sided markets (where in a simple model the 'good' is just to join or stay at a platform) two or more distinct groups of users are involved, two kinds of network effects can be distinguished: While same-side network-effects describe the network effects within each group of users, cross-side network effects describe the network effects between the two or more groups of users (respectively between the individuals of the two or more market sides).

RELATED WORK

Since two-sided and multi-sided markets frequently cannot be analyzed by means of the well-known economic rules, the implication for academic research seems to be quite clear: It is necessary to find adapted models to understand the nature of two-sided and multi-sided markets and to understand how the mechanisms of these markets work. In order to do so, especially in the last few years a lot of research has been done in this field in various directions. In the next paragraphs two major research directions are briefly introduced.

In a first important research field the new mechanisms of adding value are explored: While in traditional markets value moves from the left to the right side (where to the left side is cost and to the right side is revenue), in two-sided and multi-sided markets, cost and revenue are both to the left and the right, because the platform has to incur costs while serving both groups, but can also collect revenue from each side (Eisenmann/Parker/Van Alstyne 2006). As a result, the traditional model of the linear value-added chain is not very suitable for two-sided and multi-sided markets and has to give way for models with value-added helices, which are more qualified for describing the feedback effects and dependencies between the market sides (Dietl/Frank/Royer 2006).

A second research direction deals with the analysis of the optimal price setting in two-sided and multi-sided markets: In traditional (one-sided) markets firms have to choose the price level by which the sold quantity normally can be influenced (assuming a typically sloped demand function). Opposed to that, in two-sided or multi-sided markets firms (respectively platforms) must choose a price structure, which means that they have to find a price level for each market side and not just a general price level (Rochet/Tirole 2003). That fact and the mutual dependence of the different prices on each market side yield to the problem that setting prices in two-sided and multi-sided markets is a highly complex issue. The same holds for the analysis of the (dynamic) price competition between platforms and their pricing behaviour in different model environments (e.g. in models with product differentiation). General models of two-sided and multi-sided markets dealing with price competition and pricing behaviour can be found in Armstrong (2005) and in Rochet/Tirole (2005). Further (and sometimes more specific) models can be found (for example) in Armstrong/Wright (2007), Caillaud/Jullien (2001), Caillaud/Jullien (2003), Parker/Van Alstyne (2005), Rochet/Tirole (2002), Rochet/Tirole (2003) and Yoo/Choudhary/Mukhopadhyay (2002). Some empirical results concerning pricing behaviour (among other areas) in two-sided and multi-sided markets can be seen in Evans (2003).

RESEARCH QUESTION

Although a lot of efforts have been made to understand the way how two-sided and multi-sided markets work and although this research topic is being processed intensively at the moment, there is still a lot to do in order to understand the dynamics of the involved network effects. While analytical approaches in the field of two-sided and multi-sided markets very often have to face the general problem of an exploding complexity of the models, an agent-based approach seems to be an ideal method in this research area. It has to be mentioned that some authors have already used the agent-based approach in the field of two-sided and multi-sided markets, but with a different focus (e.g. Chen/Mäikö 2006 and Kabadjova/Tsang/Krause 2006). However, it seems that the use of the agent-based method can make it especially possible to understand this described key issue, namely the dynamics of the network effects, more accurately and more generally.

To be more precise, the agent-based approach allows to answer the research question, under which starting conditions (for example the number of individuals/agents at each platform on each side, the size of the weighting-factors for the size of the same-side and cross-side-effects, the dullness of the decisions etc.) two-sided and multi-sided markets tend (just because of the dynamics of the network-effects) to a winner-takes-all-situation (where just one platform survives) and under which starting conditions two or more platforms can survive within the market. After simulating this model the results can probably be used to analyse real existing two-sided (and multi-sided) markets.

To simplify things, only two-sided markets are considered in the model that is introduced in this progress report. The more complex multi-sided markets are likely to be considered in future research projects. However, the presented model can easily be expanded to incorporate more than two market-sides and therefore to consider these multi-sided markets as well.

AGENT-BASED MODEL

In the following subsections a simple but general agent-based model of a two-sided market is introduced, where all agents (respectively individuals) on each market side are confronted with a duopoly (meaning two platforms). Though these agents cannot change their market side (respectively their group), these agents have to decide in each period, which of the two platforms they should choose on their market side (whereas just one platform can be chosen at any time). In this model, the mentioned agents' decisions are made according to the agents' derived net-utility of each platform (a concept which is for example also used by Rochet/Tirole (2005)) which depends on the demanded prices set by the platforms, on the path-depending size of the number of other agents at the same market side (at a specific platform) and the number of other agents at the other market side (also at that specific platform), each weighted with factors that determine the size of the same-side and cross-side network effects.

Individuals, Groups and Platforms

Let's assume there exists a two-sided market with two groups of individuals (two market sides) denoted by $L,M \in \Gamma$ (whereas $L \neq M$). The individuals $l \in L$ of any of these two groups $L \in \Gamma$ are interested in interacting with the individuals $m \in M$ of the other group $M \in \Gamma$, $L \neq M$.

Let's assume further on that there exist two platforms $i, j \in P$. Each platform can help to bring together individuals of the two groups at certain points of time if individuals of both groups decide to join this platform or remain at this platform at exactly those points of time.

It is assumed that an individual stays exactly at one of the two platforms at a certain point of time, but that he or she is in general able to change the chosen platform at any of those points of time without switching costs (Varian 2003). Therefore let $l_t^i \in L_t^i$ denote an individual $l \in L$, $L \in \Gamma$ that stays at platform $i \in P$ at time $t \in T$. Because each individual stays exactly at one of the two platforms at any point of time $t \in T$, because no individual is able to change his or her group and because neither existing individuals can disappear nor new individuals can appear in the model, it follows that (whereas $i, j \in P$, $i \neq j$, $L \in \Gamma$ and $t \in T$):[1]

$$(1) \qquad \left|L_t^i\right| + \left|L_t^j\right| = \left|L_t\right| = \left|L_0\right| = \left|L\right| .$$

Let further denote $N(L_t^i)$ as the number of individuals of a group $L \in \Gamma$ that stay at platform i at time t, which means that $N(L_t^i)$ is defined as (whereas $i \in P$, $L \in \Gamma$, $t \in T$):

$$(2) \qquad N(L_t^i) = \left|L_t^i\right| .$$

Net-utility functions

First of all it is necessary to mention that all individuals in the model have static expectations, which means that they expect all observable variables at $t \in T$ to stay the same in the next period $t+1 \in T$.

Let $\alpha^{l,M}$ denote a utility-weighting-factor for a certain individual $l \in L$, $L \in \Gamma$ that determines how much utility a certain group of individuals $M \in \Gamma$ creates for this individual l. Let further $p_t^{i,L}$ denote the price (set by platform $i \in P$) that individuals of the group $L \in \Gamma$ have to pay at time $t \in T$ if they want to join the platform $i \in P$ or remain at this platform at time $t \in T$.

The net-utility of platform $j \in P$ for an individual $l \in L$, $L \in \Gamma$ that stays at platform $i \in P$ at time $t \in T$ is denoted by $u^{l_t^i}(j)$ and depends on the price set by the platform $j \in P$ that individuals of the group $L \in \Gamma$ have to pay at $t \in T$. Further on, this net-utility depends on the gross-utility that individuals of group $M \in \Gamma$ (whereas $L \neq M$) generate for this individual at time $t \in T$ ('cross-side effect') and the gross-utility that individuals of group $L \in \Gamma$ generate for this individual at time $t \in T$ ('same-side effect').

More specifically spoken, each individual $l \in L$, $L \in \Gamma$ that stays at platform $i \in P$ at time $t \in T$ calculates two net-utilities: one for platform $i \in P$ (where the individual stays at time $t \in T$) and one for the other platform $j \in P$ (whereas $i \neq j$). Concretely that means that for individual $l \in L$, $L \in \Gamma$ that stays at platform $i \in P$ at time $t \in T$ the net-utility of this platform $i \in P$ is then given by (whereas $L, M \in \Gamma$ and $L \neq M$):

[1] Mathematical remark: If A is a set, $|A|$ denotes the number of elements that are contained in A.

$$(3) \qquad u^{l_t^i}(i) = \underbrace{\alpha^{l,L}(N(L_t^i)-1)}_{\substack{gross\ utility\ generated \\ by\ group\ L \\ (same-side\ effect)}} + \underbrace{\alpha^{l,M} N(M_t^i)}_{\substack{gross\ utility\ generated \\ by\ group\ M \\ (cross-side\ effect)}} - \underbrace{p_t^{i,L}}_{price} .$$

In a similar way for the same individual at time $t \in T$ the net-utility of the other platform $j \in P$ is then given by (whereas $i \neq j$, $L, M \in \Gamma$ and $L \neq M$):

$$(4) \qquad u^{l_t^i}(j) = \underbrace{\alpha^{l,L} N(L_t^j)}_{\substack{gross\ utility\ generated \\ by\ group\ L \\ (same-side\ effect)}} + \underbrace{\alpha^{l,M} N(M_t^j)}_{\substack{gross\ utility\ generated \\ by\ group\ M \\ (cross-side\ effect)}} - \underbrace{p_t^{j,L}}_{price} .$$

According to the static expectations of all individuals these two derived utilities are used to determine the behaviour of the individuals at the beginning of the next period $t+1 \in T$. The concrete behaviour of the individuals is explained in a later subsection.

Behaviour of the platforms

To simplify the model, it is assumed that both platforms $i, j \in P$ (whereas $i \neq j$) set identical price structures over time. That means that at all periods of time $t \in T$ both platforms charge the same price p on the same market side $L \in \Gamma$. Mathematically this means that

$$(5) \qquad \forall t \in T : p_t^{i,L} \equiv p_t^{j,L}$$

or equivalently

$$(5a) \qquad \forall t \in T : p_t^{i,L} - p_t^{j,L} = 0 .$$

Although this assumption can be identified as being drastic in some respects, there are at least two reasons why this assumption makes sense in that model: Firstly, this assumption allows to concentrate on the pure dynamic effects that are caused by the same-side and cross-side effects (see next subsection) and secondly, in many two-sided markets price competition and elastic prices don't allow platforms to charge any price from the individuals – which means that they have to finance their business by means of different ways (e.g. by advertising for other companies).[2] Mathematically this means that

$$(5b) \qquad \forall t \in T : p_t^{i,L} = p_t^{j,L} = 0 .$$

This is completely compatible to the assumption made in equation (5). Nevertheless it has to be mentioned that this assumption is very radical, since there is no possibility for platforms to act and react in the model. They are in some sense completely passive, which can be criticised as being unrealistic. However in later research projects it is planned to weaken this assumption to face these claims.

---

[2] Of course, if a two-sided market finances this platform by means of advertising for other companies, this platform can in fact also be seen as a three-sided (respectively multi-sided) market.

Behaviour of the individuals

Let's assume $h(x-y)$ to be a classical Heaviside step function (whereas x and y are real numbers):

$$(6) \qquad h(x-y)=\begin{cases}0 & if \quad (x-y)\leq 0\\ 1 & if \quad (x-y)>0\end{cases}.$$

Let's assume further that z is a monotonically increasing function whose image set is assumed to be bounded such that $0\leq z\leq 1$. The probability that an individual $l\in L$, $L\in\Gamma$ (that stays at platform $i\in P$ at time $t\in T$) changes to the other platform $j\in P$ (whereas $i\neq j$) at the beginning of time $t+1\in T$ is then assumed to be

$$(7) \qquad w_{t+1}^{l_t^i}=w_{t+1}^{l_t^i}(\underbrace{u_t^{l_t^i}(j)-u_t^{l_t^i}(i)}_{utility\ \ difference})=h_{t+1}^{l_t^i}(\underbrace{u_t^{l_t^i}(j)-u_t^{l_t^i}(i)}_{utility\ \ difference})\cdot z_{t+1}^{l_t^i}(\underbrace{u_t^{l_t^i}(j)-u_t^{l_t^i}(i)}_{utility\ \ difference}).$$

In this equation the term $h_{t+1}^{l_t^i}(utility\ difference)$ secures the rationality of the individuals ('rationality term'): If the utility difference[3] is negative (which means that there is no incentive for an individual to change the platform) this term $h_{t+1}^{l_t^i}(utility\ difference)$ becomes 0. As a result the whole probability that an individual changes to the other platform $w_{t+1}^{l_t^i}(utility\ difference)$ becomes 0 as well. If however the utility difference is positive (which means that there is an incentive for an individual to change to the other platform) then the term $h_{t+1}^{l_t^i}(utility\ difference)$ becomes 1 and the probability that an individual changes to the other platform is just depending on the term $z_{t+1}^{l_t^i}(utility\ difference)$. $z_{t+1}^{l_t^i}(utility\ difference)$ is kind of a 'dullness term' (respectively 'dullness function') which ensures that if the utility difference is (although positive) only relatively small, the probability that an individual switches to the other platform $w_{t+1}^{l_t^i}(utility\ difference)$ is also relatively small. If however the utility difference is relatively high, the term $z_{t+1}^{l_t^i}(utility\ difference)$ ensures that the probability that an individual changes to the other platform $w_{t+1}^{l_t^i}(utility\ difference)$ is relatively high as well.

In this general model z is not specified exactly. In concrete agent-based simulations several specifications of the term (or function) z are possible, as long as z is monotonically increasing and the image set of this term (or function) is bounded such that $0\leq z\leq 1$ (as mentioned earlier).

Let's assume $\Psi_{t+1}^{l_t^i}$ to be a random number drawn from a continuous uniform distribution out of the interval between 0 and 1 for individual $l\in L$, $L\in\Gamma$ (that is at platform $i\in P$ at time $t\in T$) at time $t+1\in T$. The switching function W for this individual is then defined as (whereas h is again a classical Heaviside step function):

$$(8) \qquad W_{t+1}^{l_t^i}=h_{t+1}^{l_t^i}(w_{t+1}^{l_t^i}-\Psi_{t+1}^{l_t^i}).$$

This switching function allows transferring the probability that an individual switches to the other platform into a clear choice of switching: If $w_{t+1}^{l_t^i}-\Psi_{t+1}^{l_t^i}$ is greater then 0 (which is more

---

3 Because of the assumption made in equation (3), the utility difference is independent of the prices set by the firms in this model.

likely if $w_{t+1}^{l_t^i}$ is relatively big) then $W_{t+1}^{l_t^i}$ becomes 1, which means that the individual is switching at the beginning of time $t+1 \in T$. If however $w_{t+1}^{l_t^i} - \Psi_{t+1}^{l_t^i}$ is smaller then 0 (which is more likely if $w_{t+1}^{l_t^i}$ is relatively small) then $W_{t+1}^{l_t^i}$ becomes 0, which means that the individual remains at his or her platform at the beginning of time $t+1 \in T$.

Number of individuals

As defined in equation (2), $N^{L_{t+1}^i}$ denotes the number of individuals $l \in L$ of a group $L \in \Gamma$ that stay at platform $i \in P$ at time $t+1 \in T$. However, because of the introduction of the switching function W, $N^{L_{t+1}^i}$ can also be defined as a recursive function (whereas $l_t^i \in L_t^i$, $l_t^j \in L_t^j$, $L \in \Gamma$, $i, j \in P$, $i \neq j$, $t, t+1 \in T$):

$$(9) \qquad N(L_{t+1}^i) = \left|L_{t+1}^i\right| = N(L_t^i) + \sum_{l_t^j} W_{t+1}^{l_t^j} - \sum_{l_t^i} W_{t+1}^{l_t^i} .$$

With this equation the model is closed: Each individual $l \in L$, $L \in \Gamma$ that is at platform $i \in P$ at time $t \in T$ calculates two net-utilities for the two platforms at time $t \in T$ according to equations (3) and (4). Further on, at the beginning of time $t+1 \in T$ each individual calculates a probability of switching to the other platform according to equation (7). Equation (8) helps to transfer this probability into a clear choice of switching. Finally equation (9) gives the new values of $N^{L_{t+1}^i}$ at the beginning of time $t+1 \in T$ (in a recursive way) so that individuals can calculate their two net-utilities again at time $t+1 \in T$ and so on. The platforms are passive in this model according to equation (5).

PARAMETERS AND PLANNED SIMULATIONS

The general agent-based model presented above seems to be a promising model to answer the question under which starting conditions two-sided markets tend to a winner-takes-all-situation and under which starting conditions both platforms can stay within the market in the long run. It is planned to transfer this general agent-based model into the NetLogo environment (Wilensky 1999), where the outcomes of the model should be received through computerised simulations, given different starting conditions (respectively parameters).

Starting conditions (or parameters) that have to be defined (and that can be varied) can be divided into four groups: Firstly, the initial number of individuals on each side ($|L|$, $|M|$ whereas $L, M \in \Gamma$ and $L \neq M$) have to be defined (and can be varied) as well as the absolute fractions of individuals on both sides that stay at each of the two platforms initially ($\left|L_0^i\right|$, $\left|L_0^j\right|$, $\left|M_0^i\right|$, $\left|M_0^j\right|$ whereas $\left|L_0^i\right| + \left|L_0^j\right| = |L|$, $\left|M_0^i\right| + \left|M_0^j\right| = |M|$, $L, M \in \Gamma$, $L \neq M$, $i, j \in P$ and $i \neq j$).

Secondly, the ‘dullness term’ (respectively the ‘dullness function’) $z(\mathit{utility\ difference})$ has to be defined. As mentioned earlier, a lot of curve progressions of that function are possible, as long as $z(\mathit{utility\ difference})$ is defined as a monotonically increasing function with an image set that is bounded such that $0 \leq z \leq 1$. One meaningful possibility (among many others) would be defining $z(\mathit{utility\ difference}) = z(\Delta u)$ as a convex function capped by 1 (whereas e is Euler’s number and y has to be defined reasonably):

$$(10) \qquad z(\Delta u) = \begin{cases} 0 & if \quad \Delta u \leq 0 \\ e^{(\Delta u - y)} & if \quad 0 < \Delta u \;\; and \;\; e^{(\Delta u - y)} < 1 \\ 1 & if \quad 0 < \Delta u \;\; and \;\; 1 \leq e^{(\Delta u - y)} \end{cases}.$$

Thirdly, the utility-weighting-factors for all individuals $\alpha^{l,M}$ (whereas $l \in L$, $L,M \in \Gamma$) have to be defined and should be varied. The systematic variation of these utility-weighting-factors for the individuals is the key to answer the research question, when – just because of the dynamics of the network-effects – two-sided markets tend to a winner-takes-all-situation and when both platforms can survive within the market. Basically, it seems reasonable (among other possibilities) to assign real numbers out of the range between -1 and 1 to all same-side utility-weighting-factors for all individuals $\alpha^{l,L}$ (whereas $l \in L$, $L \in \Gamma$) and to assign real numbers out of the range between 0 and 1 to all cross-side utility-weighting-factors for all individuals $\alpha^{l,M}$ (whereas $l \in L$, $L,M \in \Gamma$, $L \neq M$). The reason why only positive values are assigned to cross-side utility-weighting-factors is that negative values would in some sense contradict the definition of a two-sided (and multi-sided) market, which says that two-sided or multi-sided markets tie together two or more distinct groups of users in a network, whereas the individuals of each group are interested in interacting with the individuals of the other group.[4] Therefore if negative numbers were assigned to the cross-side utility-weighting-factors, the individuals of each group wouldn't be interested in interacting with the individuals of the other group.

To be more specific, it seems meaningful (among many other reasonable possibilities) to draw the same-side utility-weighting-factors for all individuals $\alpha^{l,L}$ for a distinct group $L \in \Gamma$ (whereas $l \in L$) from a defined continuous uniform distribution out of several, for example out of five defined sub-ranges (out of five possibilities), e.g. out of the ranges $[-1.0,-0.6]$, $[-0.6,-0.2]$, $[-0.2,0.2]$, $[0.2,0.6]$ and $[0.6,1.0]$.

In the same way, it seems reasonable to draw the cross-side utility-weighting-factors for all individuals $\alpha^{l,M}$ for a distinct group $L \in \Gamma$ (whereas $l \in L$, $L,M \in \Gamma$, $L \neq M$) from a defined continuous uniform distribution out of several, for example out of two defined sub-ranges (respectively possibilities), e.g. out of the ranges $[0.2,0.6]$ and $[0.6,1.0]$.

Given the two groups (respectively market sides) $L,M \in \Gamma$ (whereas $L \neq M$) and given that the same-side and cross-side utility-weighting-factors are assigned to the individuals of each group according to the defined possibilities above, to answer the described research question this setting would yield $2 \times 2 \times 5 \times 5 = 100$ possible utility-weighting-factors-combinations for each combination of the other starting conditions (respectively parameters) that have to be varied in the simulations.

Finally, the number of time periods $|T|$, the numerous simulations with the different configurations of the parameters will run until they stop, has to be defined. Since the research question refers to the outcomes of the model in the long run, it seems meaningful to define $|T|$ such that $|T| \geq 1000$.

---

4 It has to be mentioned that this is not generally true for every two-sided or multi-sided market one could think of. For example, imagine a media platform (e.g. a television program) where the platform brings together viewers and advertisers. In such a case cross-side network effects and therefore cross-side utility-weighting factors may be positive in just one direction and zero or negative in the other direction and yet it is a two-sided market (Peitz/Valletti 2005; Reisinger 2004).

CONCLUSION AND OUTLOOK

Although no computerised simulations have been done so far in a simulation-environment like NetLogo, the presented general formulation of an agent-based model of a two-sided market seems very promising to study and analyze under which starting conditions (given a duopoly) two-sided and multi-sided markets tend (just because of the dynamics of the network-effects) to a winner-takes-all-situation and under which starting conditions both platforms can survive within the market. The use of an agent-based approach seems to be an appropriate and useful method in this field, since traditional approaches often have to face the problem of an exploding complexity of the created models.

Nevertheless, the model presented in this paper is relatively simple: Firstly, only two-sided markets are considered. Secondly, platforms have to be more or less passive in the model and thirdly, just one platform can be chosen from each agent at any time – which means that 'multi-homing' is not possible. In future versions of the model it is planned to weaken these limitations, especially the passivity of the platforms.

REFERENCES

Armstrong, M., 2005, "Competition in two-sided markets," RAND Journal of Economics, Vol. 37, No. 3, pp. 668–691.

Armstrong, M., J. Wright, 2007, "Two-sided markets, competitive bottlenecks and exclusive contracts," Economic Theory, Vol. 32, No.2, pp. 353–380.

Caillaud, B., B. Jullien, 2001, "Competing Cybermediaries," European Economic Review, Vol. 45, No. 4-6, pp. 797–808.

Caillaud, B., B. Jullien, 2003, "Chicken and Egg: competition among intermediation service providers," RAND Journal of Economics, Vol. 34, No. 2, pp. 309–328.

Chen, X., J. Mäkiö, 2006, "A Simulation Approach for Analyzing the Competition among e-Auction Marketplaces," Proceedings of IEEE International Conference on e-Business Engineering, Shanghai, China, pp. 17-24.

Dietl, H., E. Frank, S. Royer, 2006, "Strategische Wettbewerbsvorteile und Wertschöpfung-sorganisation – Konzeptionelle Überlegungen und Gestaltungsüberlegungen für zwei- und mehrseitige Dienstleistungsmärkte," Working Paper, Institute for Strategy and Business Economics, University of Zurich; available at http://ideas.repec.org.

Eisenmann, T., G. Parker, M. W. Van Alstyne, 2006, "Strategies for Two-Sided-Markets," Harvard Business Review, October 2006.

Evans, D. S., 2003, "Some Empirical Aspects of Multi-sided Platform Industries," Review of Network Economics, Vol. 2, No. 3, pp. 191–209.

Kabadjova, B. A., E. Tsang, A. Krause, 2006, "Competition among Payment Cards: An Agent-Based Approach," Working Paper, Centre for Computational Finance and Economic Agents, University of Essex.

Katz, M. L., C. Shapiro, 1985, "Network Externalities, Competition, and Compatibility," The American Economic Review, Vol. 75, No. 3, pp. 424–440.

Katz, M. L., C. Shapiro, 1986, "Technology Adoption in the Presence of Network Externalities," Journal of Political Economy, Vol. 94, No. 4, pp. 822–841.

Parker, G. G., M. W. Van Alstyne, 2005, "Two-sided Network Effects: A Theory of Information Product Design," Management Science, Vol. 51, No. 10, pp. 1494–1504.

Peitz, M., T. M. Valletti, 2005, "Content and advertising in the media: pay-tv versus free-to-air," CEPR Discussion Paper, No. 4771; available at http://ssrn.com/abstract=666825.

Reisinger, M., 2004, "Two-Sided Markets with Negative Externalities," Discussion Paper, University of Munich; available at http://epub.ub.uni-muenchen.de.

Rochet, J.-C., J. Tirole, 2002, "Cooperation among competitors: some economics of payment cards associations," RAND Journal of Economics, Vol. 33, No. 4, pp. 549–570.

Rochet, J.-C., J. Tirole, 2003, "Platform Competition in Two-Sided Markets," Journal of European Economic Association, Vol. 1, No. 4, pp. 990–1029.

Rochet, J.-C., J. Tirole, 2005, "Two-sided Markets. A Progress Report," IDEI Working Paper; available at http://idei.fr/doc/wp/2005/2sided_markets.pdf.

Tirole, J., 2002, "The Theory of Industrial Organization," The MIT Press, 13th edition.

Varian, H. R., 2003, "Intermediate Microeconomics," W. W. Norton & Company, 6th edition.

Wilensky, U., 1999, "NetLogo," Center for Connected Learning and Computer-Based Modeling. Northwestern University; available at http://ccl.northwestern.edu/netlogo.

Wright, J., 2004, "One-Sided Logic in Two-Sided Markets," Review of Network Economics, Vol. 3, No. 1, pp. 44–64.

Yoo, B., V. Choudhary, T. Mukhopadhyay, 2002, "Pricing Strategies of Electronic B2B Marketplaces with Two-Sided Network Externalities," Proceedings of the 35th Hawaii International Conference on System Sciences, Vol. 7.

AXELROD'S METANORM GAMES ON COMPLEX NETWORKS

J. GALAN, University of Burgos
M. LATEK[1], George Mason University
M. TSVETOVAT, George Mason University
S. RIZI, George Mason University

ABSTRACT

In this paper we adapt Axelrod's metanorms model to run on arbitrary interaction structures. We propose a hybrid analytical methodology combining computer simulations with a novel mathematical approach that abstracts some details of evolutionary dynamics while retaining explicit and exact representation of underlying network topology. We construct simple network statistics based on the clustering coefficient and first order degree distribution of the interaction network as strong predictors of the emergence of cooperative norms.

Keywords: Metanorms, evolutionary processes, multi-person games on networks, agent-based social simulation

INTRODUCTION

Axelrod's metanorms model, (Axelrod 1986), is an *n*-person game that captures the essence of several social dilemmas, stemming from the fact that any one individual is better off defecting irrespective of the other players' decisions, but universal cooperation is preferred to universal defection. Axelrod's model illustrates mechanisms that explain the emergence or collapse of cooperative social norms when punishing norm deviants is costly. The model assumes that agents face potential punishment not only when they deviate from the norm, but also when they fail to punish norm deviants. A simple evolutionary principle drives the model: the more successful a strategy has been so far, the more likely agents are to adopt it in the future. While it provides insights into the process of emergence, the model adopts the generally false assumption that each player interacts with all other players. This assumption leads to an extreme payoff structure and an exponential growth in the number of interactions that quickly exceed any reasonable cognitive constraints as number of players grows.

To highlight this weakness of Axelrod's model, consider the following *n*-person prisoner's dilemma: imagine some students taking a test in a large lecture hall with a proctor who fails to exercise strict vigilance. Each student has some incentive to cheat; however, a student's cheating imposes two costs on the rest of the students taking the

[1] Corresponding author: Maciej M. Latek, Center for Social Complexity, Krasnow Institute for Advanced Study, George Mason University, Fairfax, VA 22030; e-mail: mlatek@gmu.edu.

exam. First, given how much they have studied for the test, the rest of the students will not obtain as good a grade as the cheater; second, if cheating diffuses among the students in the class and possibly among various classes, the prestige of the institution will fall; students will have paid for studies without much real value. Suppose a student (*S*), who may or may not cheat himself, notices that another student (*Cheater*) is cheating. *S* can punish *Cheater* by informing the proctor, but punishing *Cheater* is costly for *S*: usually no one wants to be seen as a "snitch". If *S* refuses to punish *Cheater* because of the punishment cost involved, another student (S^*) who has observed the situation between *Cheater* and *S* can either punish *Cheater* by informing the proctor that he has cheated or punish *S* by informing the proctor that *S* has refused to punish *Cheater*. Punishing *S* is also costly for S^*. When this simple example is played repeatedly, it captures the essence of Axelrod's metanorms game.

In some social contexts the idea of complete interaction among individuals on a fully connected graph is realistic; in many other cases, it is not. For instance in the above example of cheating during the test, the probabilities of detecting a cheater depend on the size and topology of the lecture hall that can be reasonably approximated by a lattice-based topology. In other social contexts a lattice model is not empirically relevant. Recent work on social network analysis (Newman 2003) has shown that social interaction structures may differ markedly from the stylized regular grids or fully connected graphs used in the extant versions of Axelrod's model. In this paper we extend the metanorms model to arbitrary interaction topologies. By doing so we hope to contribute to a better understanding of the metanorm mechanism as a norm inducer and further the literature on evolutionary game theory and social dilemmas in networks (Santos *et al.*2006; Szabo and Fath 2006).

The structure of the paper is as follows: in the next section we give some background to Axelrod's metanorms model and describe how it is adapted to be played in networks; we then proceed to the analytical abstraction used to understand the dynamics of the metanorms model. Next, we provide simulation results and discuss them. Finally, we present the conclusions.

METANORMS MODEL IN NETWORKS

Axelrod's metanorms model in networks begins with a setup phase. Initially 50 agents are embedded in a network structure developed by a predetermined network generation algorithm.[2] We have analyzed the effect of two different algorithms: the Barabasi-Albert algorithm that generates networks with power law degree distributions (Barabasi and Albert 1999) and the Watts algorithm (Watts 1999) with different values of rewiring probability (β) that smoothly interpolates between two extreme cases of a regular lattice and a random network, traversing the small world networks category (Watts and Strogatz 1998). A link between two agents represents an opportunity for direct interaction between them. A set of all links for an agent is called neighborhood of the agent.

Once agents and the underlying network structures are created, agents play a repeated game that consists of three decisions or stages:

[2] We used 50 instead of 20 agents used in Axelrod's default setting to make high-order statistics more interpretable.

1. Agents decide whether to cooperate or defect. A defecting agent obtains Temptation payoff (T = 3) and inflicts on each of the remaining agents in the population Hurt payoff (H = -1). If agents cooperate, no one's payoff is altered. Here we assume that the spillover cost of defection is global.
2. Agents observe other agents in their neighborhood who defected in stage 1 with a certain probability. For each observed defection, agents decide whether to punish the defector or not. Punishment is costly: one must pay Enforcement cost (E = -2) to impose Punishment cost (P = -9) on the defector. Notice that the opportunity to observe defection; hence the possibility to punish it, is conditional on the existence of a link connecting both agents.
3. The third step includes the concept of metanorm. The metanorm here is that agents who fail to punish observed defection should be punished. Similar to the previous step, an agent who fails to punish an observed defection may not be caught. The probability of being seen not punishing a defection given that the defection has been seen is the same as the probability of observing such defection. Network topology plays a critical role in this step: it determines who can see unpunished defection (See Figure 1). Observing a defection requires links among the defector, un-punishing agent and metapunisher. As for payoffs, a metapunisher pays Meta-enforcement cost (ME = -2) every time it Meta-punishes (MP = -9).

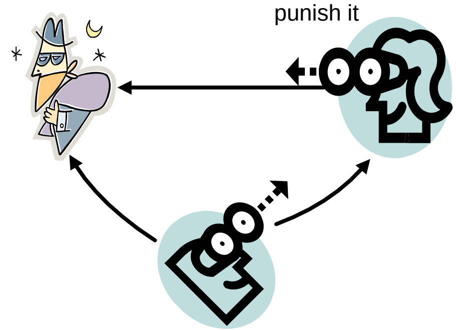


FIGURE 1 Links needed in the network to have a chance of meta-punishment. Links are undirected; the arrows simply clarify the logic of the dynamic.

The strategy of an agent depends on two of its characteristics: boldness and vengefulness. Boldness is the probability that an agent defects in the first stage; vengefulness is its propensity to punish agents that have been observed defecting in the second and third stages. Following Axelrod's implementation, we implement each of these probabilities as a 3-bit string denoting eight evenly-distributed values from 0 to 1 (0/7, 1/7, ..., 7/7). Initial values for each agent's strategy are determined randomly at the beginning of each simulation run and updated through an evolutionary mechanism.

Once agent payoffs are determined at the end of a generation, agents can change their strategies according to two evolutionary forces of selection and mutation. We have implemented four selection mechanisms where the most successful agents at a particular time have the highest probability of having their strategy copied, but with different variance (De Jong 2006), adapting them for local use within a network structure:

1. **Best neighbor**. Select the strategy used by the most successful agent in the neighborhood. Break ties randomly.
2. **Average selection**. Select agents in the neighborhood whose payoff are higher than the average plus a standard deviation of neighborhood payoffs. Select one strategy in this set randomly as new strategy. If the set is empty, continue with current strategy.
3. **Random tournament**. Compare a randomly selected strategy from the neighborhood with the current strategy. Select the strategy with higher payoff.
4. **Roulette wheel**. Assign an adoption probability to each neighborhood strategy proportional to the payoff other agent obtained using it minus the minimum payoff obtained in the neighborhood.

Whenever a bitstring-coded strategy is replicated, that is, when agents invoke a selection mechanism, every bit has a certain probability of being flipped called mutation rate. We have studied the effect of different selection mechanisms as well as different mutation rates.

RESULTS

We combined mathematical modeling and agent-based simulation to analyze the model. These two approaches have been shown to complement one another in obtaining a better understanding of complex systems (Galan and Izquierdo 2005; Izquierdo and Izquierdo 2006; Izquierdo *et al.* 2007).

Mathematical modeling results

Mathematical analysis of the original metanorms game is a challenging task; including realistic interaction structures makes it even more cumbersome. Therefore, we use simplifying strategies to arrive at closed form solutions amenable to mathematical analysis and graphical representation. We present a summary of the abstraction process of the model following the logic presented in Galan and Izquierdo (2005), with a number of substantial changes introduced. Given an undirected network $\mathbf{\Gamma} = (N, L)$ defined by a set of agents N and set of links, the payoff of an agent i playing the metanorms game is defined by:

$$Payoff_i = Def_i \cdot T + \sum_{\substack{j \in \mathrm{N} \\ j \neq i}} Def_j \cdot H + \sum_{j \in \mathrm{N}^i} Pun_{ij} \cdot E +$$

$$\sum_{j \in \mathrm{N}^i} Pun_{ji} \cdot P + \sum_{j \in \mathrm{N}^i} \sum_{k \in \mathrm{N}^i \cap k \in \mathrm{N}^j} ME \cdot Def_k \cdot (1 - Pun_{jk}) \cdot Pun_{ij} +$$

$$\sum_{j \in \mathrm{N}^i} \sum_{k \in \mathrm{N}^i \cap k \in \mathrm{N}^j} MP \cdot Def_k \cdot (1 - Pun_{ik}) \cdot Pun_{ji}$$

Where following notation is used:

1. T, H, E, P, ME, MP are the payoffs of the metanorms game and are set exogenously;
2. *n* is the number of agents;
3. $N_i \equiv \{ j \in N : ij \in L \}$ is the set of nodes connected to any given $i \in N$. This set defines the *neighborhood* of agent *i*. $z^i \equiv \left| N^i \right|$ denotes the degree of the agent *i*.

In addition, indicator functions *Def* and *Pun* have been defined as follows:

$$Def_i = \begin{cases} 1 & \text{If agent } i \text{ defects} \\ 0 & \text{If agent } i \text{ cooperates} \end{cases} \qquad \begin{array}{l} \text{Prob}\,(Def_i \equiv 1) = b_i \\ \text{Prob}\,(Def_i \equiv 0) = 1 - b_i \end{array}$$

$$Pun_{ij} = \begin{cases} 1 & \text{If agent } i \text{ punishes agent } j \\ 0 & \text{If agent } i \text{ does not punish agent } j \end{cases} \qquad \begin{array}{l} \text{Prob}\,(Pun_{ij} \equiv 1) = b_j \cdot \left(b_j / 2 \right) \cdot v_i \\ \text{Prob}\,(Pun_{ij} \equiv 0) = 1 - b_j \cdot \left(b_j / 2 \right) \cdot v_i \end{array}$$

We define the clustering coefficient of agent *i* with at least two neighbors as:

$$C^i \equiv \frac{\left| \{ jk \in L : ij \in L \wedge ik \in L \} \right|}{\frac{z^i (z^i - 1)}{2}}$$

Lastly, we define clustering coefficient for a degree in a given network as

$$C(k) = \frac{\sum_{\substack{i=1 \\ z^i = k}}^{n} C^i}{\left| \left\{ i \in N : z^i = k \right\} \right|}$$

Assuming continuity and homogeneity of population and abstracting the details of the network by selected network statistics, the expected payoff of an agent *i* in one round can be calculated as:

$$Exp(Payoff)_i = b_i \cdot T + H(n-1)B + E\frac{v_i}{2}B^2 \cdot \overline{k} +$$
$$P\frac{b_i^2}{2}V \cdot \overline{k} + ME\frac{v_i}{4}B^3(1-V)\sum_{d=1}^{n-1} p(k_i = d) \cdot C(d) \cdot d \cdot (d-1) +$$
$$MP\frac{1-v_i}{4}B^3 V\sum_{d=1}^{n-1} p(k_i = d) \cdot C(d) \cdot d \cdot (d-1)$$

In order to evaluate the dynamics and discover the evolutionary stable states of the system (Galan and Izquierdo 2005), we compute the differences between the payoff obtained by a mutant *m* and the payoffs obtained by a representative agent *i* of the incumbent population when the mutant changes its strategy. This difference may contribute to shifting the system by means of selection pressure. The results depend on the topological properties of the network, in particular on the clustering coefficient and first order degree distribution of the network. For random networks where the degree distribution can be modeled by a Poisson distribution, it is easy to prove that the clustering coefficient is independent of the node degree. So we can study the system mathematically in a clean and stylized fashion. In more complex networks we estimate the required statistics of the network (degree, degree-clustering) numerically so they can be plugged back into equations. Lastly, we will define Interconnectivity factor as follows:

$$InterConn = \sum_{d=1}^{n-1} \mathrm{p}(d)C(d)d(d-1)$$

Specifically if we examine the last equation, we observe that from a network point of view the different addends only depend on the number of agents, the average degree and interconnectivity factor influenced by the clustering degree distribution. The resulting gradient vector field of the system under different parameters topologies of networks consisting of 50 agents is presented in Figure 2. Arrows represent the expected movement of the system (tangent to the trajectories) and colors correspond to the module of the gradient (the speed of the trajectories), the more red the faster. The *x*-axis represents average boldness of population, *y*-axis average vengefulness.

In the first row we fix the average degree of the network to 4. Columns correspond to values of 2, 10 and 20 of the interconnectivity factor. The red dashed line is the boundary that separates the region of left-pointing arrows and the region of right-pointing arrows. The white line separates the area of top-pointing arrow and down-pointing arrows. In the first figure we observe a single evolutionary stable state, marked with a red dot, in the zone of norm collapse at the bottom-right area corresponding to high average boldness and low average vengefulness. Increasing the interconnectivity factor, that is, increasing clustering coefficient for a constant average degree shifts the white dashed line toward the left in the second and third figures. When the interconnectivity factor is large enough, dashed lines cross: a second evolutionary stable state emerges, this time in the zone of norm establishment at the top-left area corresponding to high average vengefulness and low average boldness. The purple line separates basins of attraction for both states.

In the second row we fix the interconnectivity factor while increasing the average degree of the network from 4 to 8 and 16. We observe that the second evolutionary stable state disappears. This result is mainly due to the fact that for fixed population sizes, average degree and the interconnectivity factors are not independent values. Regardless of that, as both lines were pushed to the right, the escape corridor from the zone of norm establishment toward the evolutionary stable state in the zone of norm collapse is very narrow indicating that with a high level of noise the system may spend a lot of time in the zone of norm establishment.

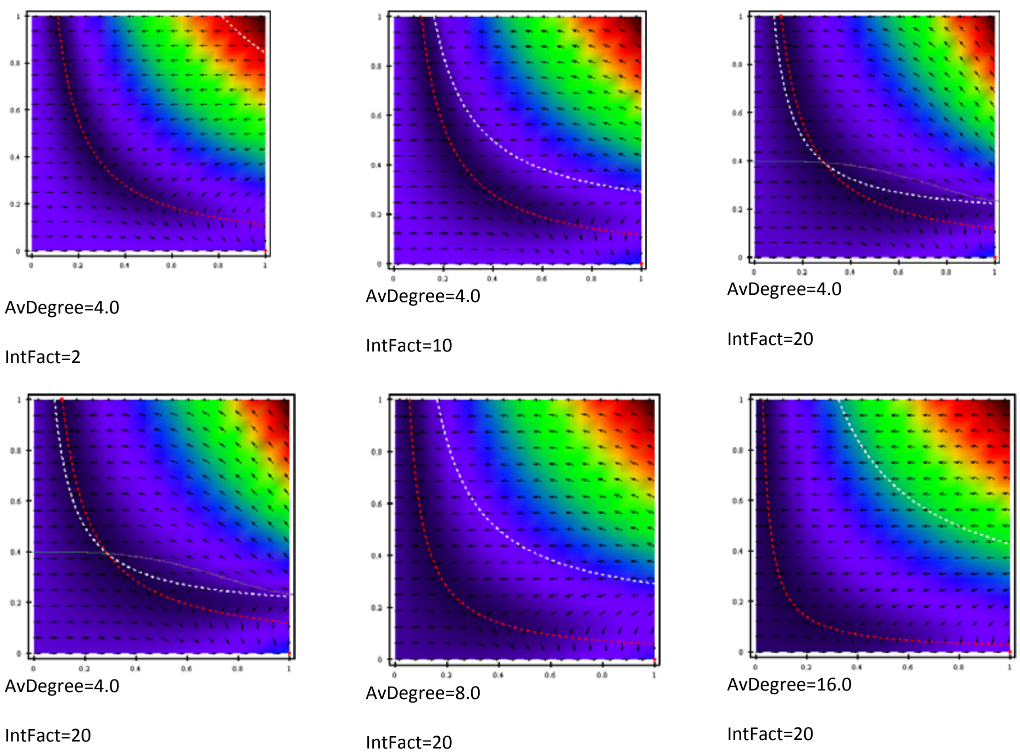


FIGURE 2. Expected dynamics of the metanorms model on networks. We use Axelrod's original parameter values and assume continuity of boldness and vengefulness. We also assume relative homogeneity of agents' strategies. The *x*-axis represents average boldness; *y*-axis represents average vengefulness. The meaning of the colors and lines are explained in the text.

Simulation results

We implemented the model in Java 2 using RePast 3 (North *et al.* 2006) and JUNG libraries[3]. We conducted several experiments, focusing on the role of the interconnectivity factor in the dynamics of the metanorms model. Bear in mind that the results presented in the previous section about the influence of the interconnectivity factor on the game and indirectly on the influence of the clustering coefficient and average degree of the network were derived from an abstraction of the game. To confirm these analytical results, we simulated the game for 10000 runs. Each run was initialized with different interaction topologies and produced time series of the

[3] http://jung.sourceforge.net/

proportion of time that the system, in the long run, is in the zones of norm collapse and norm establishment. We used networks generated by the Barabasi-Albert and Watts algorithms, randomizing the parameters of network generation and varying evolutionary details of the network. Figure 3 shows the percentage of time the system spends in the zone of norm establishment as a function of the interconnectivity factor and clustering coefficient of the network for networks with average degrees between 4 and 7.

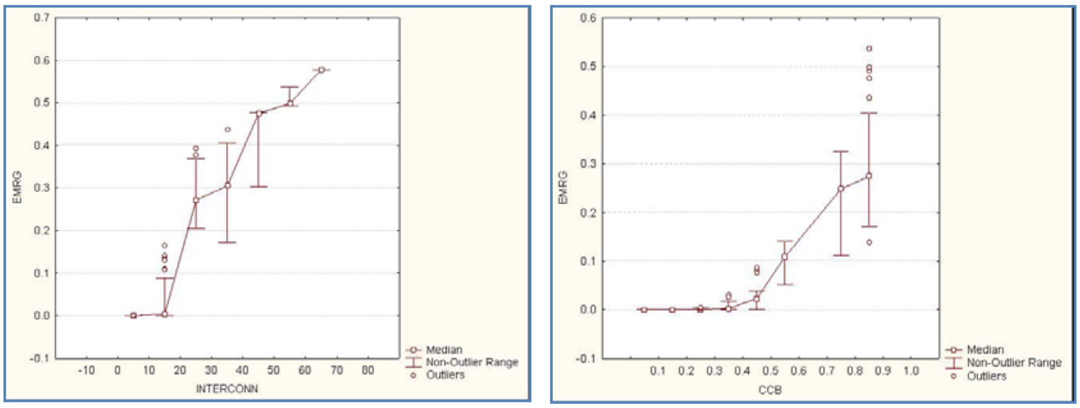


FIGURE **3.** Percentage of time the system spends in the zone of norm establishment (EMRG) depending on the interconnectivity factor (INTERCONN), on the left panel, and clustering coefficient (CCB), the right panel, of the network.

In the left diagram of Figure 3 we observe that the average amount of time spent in the zone of norm establishment increases monotonically with the interconnectivity factor. It also illustrates that there is an important variance and some outliers in the graph. This may be due to the influence of the evolutionary details of the game for each specific simulation, but more work is needed to confirm this hypothesis. The right diagram represents the same percentage of time but as a function of the clustering coefficient of the network: the higher the clustering coefficient of the network the more time spent in the zone of norm establishment. Also note that the variance of the average time for norm establishment clearly increases with its size. However, as the mathematical model suggests, the interconnectivity factor seems a better predictor.

CONCLUSIONS

We combined mathematical simulation approaches to analyze evolutionary games played on networks. We investigated the influence of the average degree and the interconnectivity factor of an interaction network on the process of norms emergence in the metanorms model by analytical and simulation techniques. Simulation results obtained confirmed the positive influence of interconnectivity factor on reinforcing the meta-punishment mechanisms.

ACKNOWLEDGEMENTS

This work was supported in part by the NSF Human and Social Dynamics Program, grant no. 0527471. and by the Spanish Ministry of Education and Science through Projects DPI2004-06590 and DPI2005-05676.

REFERENCES

Axelrod, R.M. (1986) An Evolutionary Approach to Norms. *American Political Science Review*, 80, 1095-1111.

Barabasi, A.L. and Albert, R. (1999) Emergence of scaling in random networks. *Science*, 286, 509-512.

De Jong, K.A. (2006) *Evolutionary computation. A unified approach*. MIT Press, Cambridge, Mass.

Galan, J.M. and Izquierdo, L.R. (2005) Appearances Can Be Deceiving: Lessons Learned Re-Implementing Axelrod's 'Evolutionary Approach to Norms'. *Journal of Artificial Societies and Social Simulation*, 8, 2.

Izquierdo, L.R., Izquierdo, S.S., Gotts, N.M., and Polhill, J.G. (2007) Transient and Asymptotic Dynamics of Reinforcement Learning in Games. *Games and Economic Behavior*, In press.

Izquierdo, S.S. and Izquierdo, L.R. (2006) On the Structural Robustness of Evolutionary Models of Cooperation. In: *Intelligent Data Engineering and Automated Learning - IDEAL 2006. Lecture Notes in Computer Science 4224* (Eds. E. Corchado, H. Yin, V.J. Botti, and C. Fyfe), pp. 172-182. Springer, Berlin Heidelberg.

Newman, M.E.J. (2003) The structure and function of complex networks. *SIAM Review*, 45, 167-256.

North, M.J., Collier, N.T., and Vos, J.R. (2006) Experiences Creating Three Implementations of the Repast Agent Modeling Toolkit. *ACM Transactions on Modeling and Computer Simulation*, 16, 1-25.

Santos, F.C., Pacheco, J.M., and Lenaerts, T. (2006) Evolutionary dynamics of social dilemmas in structured heterogeneous populations. *Proceedings of the National Academy of Sciences of the United States of America*, 103, 3490-3494.

Szabo, G. and Fath, G. Evolutionary games on graphs. http://www.citebase.org/abstract?id=oai:arXiv.org:cond-mat/0607344 . 2006. Ref Type: Electronic Citation

Watts, D.J. and Strogatz, S.H. (1998) Collective dynamics of 'small-world' networks. *Nature*, 393, 440-442.

Watts, D.J. (1999) *Small worlds: the dynamics of networks between order and randomness*. Princeton University Press, Princeton, N.J.

NETWORK FRACTURE: HOW CONFLICT CASCADES REGULATE NETWORK DENSITY

E. M. O'GRADY, George Mason University, Fairfax, VA
M. ROULEAU, George Mason University, Fairfax, VA
M. TSVETOVAT, George Mason University, Fairfax, VA *†

October 1, 2007

Abstract

The complexity of human social structures often masks the simplicity involved in their development. Social networks are a product of dynamic processes and feedback. In other words, the ties that people make affect the topology of a network and the form of a network affects the ties that people make. Therefore, social network structure evolves in a path-dependent manner. In this paper, we begin to sift through the complexity of social network ties in an effort to unearth the fundamental rules of social interaction and their impact on network formation and evolution.

Keywords: social networks, evolution, conflict propagation, avalanche, power law

1 Introduction

The complexity of human social structures often masks the simplicity involved in their development. Social networks are a product of dynamic processes and feedback. In other words, the ties that people make affect the topology of a network and the form of a network affects the ties that people make. Therefore, social network structure evolves in a path-dependent manner. In this paper, we begin to sift through the complexity of social network ties in an effort to unearth the fundamental rules of social interaction and their impact on network formation and evolution.

We have all witnessed social turmoil in our midst – or even have been involved in its very middle. A long-married couple decides on a divorce – and suddenly their friends are faced with difficult decisions. They may feel pressured to side with one partner or the other,

*Corresponding author: M. Tsvetovat *mtsvetov@gmu.edu*

†Authors appear in alphabetical order

potentially splitting long-standing friendships and dividing a formerly cohesive network into "his side" and "her side". As the wounds of the split-up heal, the space is opened up for creation of new friendships and romantic relationships, and the cycle starts again.

The example above illustrates several concepts. The first is the ability of change in network structure – particularly change of a destructive nature –

to propagate through a network, potentially affecting a large number of people. Second, the network reacts to addition and deletion of edges in qualitatively different fashions, depending on its density, undergoing a phase transition [8]. Finally, it has been observed [6] that networks in the real world settle to a certain density, suggesting the presence of a dynamic equilibrium [7]. We hypothesize that conflicts play a regulatory role in social networks and help establish and maintain this dynamic equilibrium.

Considerable research [10, 4, 9] has centered on the generative processes of network creation. Such processes range from purely random models [3] to generation of small-world networks [11]and scale-free networks [1]. While methods of network generation and results vary widely, these methods have one common property: they consider only processes that add or change edges of the network. Although the generative processes that create giant component networks are theoretically valid, in real-world networks the forces responsible for this formation are mitigated by an unknown resistance. As the generative and resistance forces attempt to balance, the system enters and oscillates around a dynamic equilibrium. We consider this counter-force to be node or edge deletion, which we describe as conflict. In this paper, we would like to consider the role of destructive processes, such as tie extinction and outright conflict, as an equally important influence upon network topology.

We implement a simple agent-based model to explore the complexities of network structures and the consequences of conflict upon these structures. We are interested in the micro-level mechanisms that produce the macro-level patterns observed in real-world networks. To achieve this, we employ a simple set of agent rules. These rules were selected for two reasons: 1) they address the micro-level or social aspect of network formation using structural balance theory; 2) they produce the macro patterns that are commonly observed in social networks. To show how these two simple rules function dynamically, the rest of the paper will develop these ideas in three stages. First, we describe the qualitative logic behind network formation. We then attempt to support these qualitative claims through a process of agent-based experimentation. Finally, we conclude with a synthesis of the above-mentioned qualitative and experimental undertakings.

We use structural balance theory [2], derived from Heider's balance theory [CITE], to model agent behavior. This notion is represented as a triadic structure where the triad represents an individual's attitude toward both physical and social objects. The links between the objects represents the affect the primary object has towards the other two. Balance theory suggests that people, from the perspective of the individual, have a preference for balance. Unbalanced structures occur when the primary individual perceives a difference between their affect and that of other objects. This difference results in a sense of discomfort for the primary individual, which they may address through a change in affect, thus, changing the structure. We use structural balance theory to examine social structure and measure the level of "discomfort" present in a network. In our model, "discomfort" is represented as conflict (or an enemy link) and is caused by opposing relationships among agents.

A simple triadic rule set can form the basis of a complex social network whose structure is dependent on the stability of the linkages: newly added actors or linkages act as generative processes (i.e. a couple's friends becoming friends) that percolate throughout the network. Similarly, conflict, which acts as a resistance process, may cause a relationship to reverse and can trigger a chain reaction, restructuring a network. The complexity of the network structure plays a considerable role in the consequences of these changes. The rate that change percolates throughout a social network is highly dependent upon the density of the network. Returning to the example of our couple, the number of peripheral actors drawn into a given linkage change is dependent upon how well the remaining actors are connected throughout the system. Thus, as the generative and resistance forces attempt to balance, the system enters and oscillates around a dynamic equilibrium.

2 Simulation Model

To study the effect of conflict on the topology and density of networks, we propose a simple computational model: an agent-based version of a highly simplified social network. The model serves as an experimental environment using four 'social rules' previously described. In its initial state, no social connections exist amongst agents. Subsequent stochastic agent interaction evolves the social network and serves as a starting point to examine network conflict. As conflict is introduced into the network, agents utilize the social rules to achieve or maintain balanced relationships.

The Poisson scheduler is responsible for agent activation. It ensures that, throughout the simulation, all agents are active on average but not necessarily active the same number of times in each round. This enabled the model to replicate the heterogeneous social environment faced by social agents within the real world.

The base model consists of a static set of 100 agents. The social environment changes for later simulations which adds new agents to the system at a fixed rate per round. When activated, an agent randomly chooses from the remaining strangers in the simulation and attempts to establish a new friend connection. If the active agent already possesses ties from a previous round, this agent will attempt to meet the friends of its current friends. Probabilities of making new friends among strangers or friends-of-friends have been parameterized for further study.

Agents react to changes to their friendships and conflicts with a set of simple rules of triadic interaction:

Rule 1 A friend of my friend is my friend (*Simmelian tie[5]*)

Rule 2A An enemy of my friend is my enemy (*social balance[2]*)

Rule 2B A friend of my enemy is my enemy

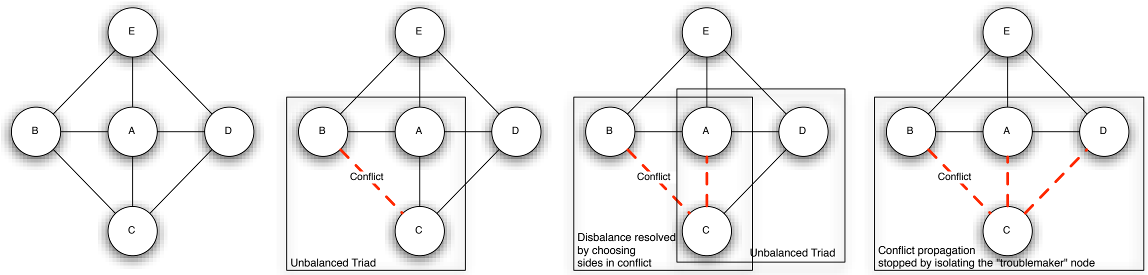


Figure 1: Propagation of Conflicts in a Dense Network

Rule 2C A enemy of my enemy is my friend[1]

Rules 2A, 2B, and 2C represent the same balanced triad containing two conflict links and one friendship link the only difference is the viewpoint taken by each rule. To maintain simplicity, Rules 2A, 2B, and 2C, referred to as Rule 2, is stated as follows:

> Each node shall seek to be embedded in balanced triads that are either fully connected *Symmelian triads* or *conflict-balanced triads*.

Conflict is introduced in the network at a constant probability, by changing a single friendship tie into an enemy tie. What happens then is illustrated on figure 1. In this simple example, a network consisting of 4 closed triads is struck by a conflict on a single edge. Triad $A - B - C$ becomes unbalanced due to a conflict between B and C; thus A is forced to take sides in the conflict by choosing to remain friends with either B or C, at random. Adding conflict to the $A - C$ edge forces another triad, $A - C - D$ to become unbalanced, thus drawing agent D into the conflict. If agent D then chooses to isolate C from the rest of the network, the propagation of the conflict can be stopped. However, if instead it separates from A, this will cause the conflict to propagate further and destroy more links.

Despite the simplicity of the micro-processes involved in our simulation, our agent-based model is able to replicate well-known macro patterns. This is largely due to the heterogeneity of social tie interpretation involved in the model. In other words, due to the stochastic nature of agent interaction, no two agents possess the same picture of the developing social network. Not all agents share a tie with all other agents in the network and this inequality in social ties provides the necessary condition for percolation.

Having more ties increases an agents probability of forming even more ties, but also increases the probability that a conflict between two agents spread throughout the network. Thus, percolation of both friendship and conflict is dependent upon the density and timing of agent connections.

[1]Attributed to: the Bible (Exodus 23:22 and Matthew 22), ancient Chinese and Arab proverbs. Most likely, as old as the world.

3 Simulation Experiments

The four simple agent decision-making rules of our model are able to mimic common characteristics of structural network growth and evolution. We first illustrate phase transition, specifically focusing on the growth of network friendship ties. Next we introduce conflict, highlighting the importance of conflict percolation and the inherent self-sustainability of the network. Finally, we end with a discussion of the sensitivity of the models parameters in relation to the realism of the models results.

3.1 Phase transition from linear to exponential growth

The model was run without conflict to assess its ability to create friendship ties. Initially, the network grows linearly with addition of ties. After a given threshold is passed, friend-of-friend triads begin to connect into balanced triads and the growth of network density accelerates dramatically. In absence of conflict, our model produces a sigmoid curve of network density growth (see Figure 1). This result is largely dependent upon the probability of an agent making friends with its friends friends. The more connections made in prior rounds increase the chances of producing exponentially more friendship ties in future rounds until most potential friendship ties have been exhausted. Beyond this exhaustion point, a decreasing probability of making new ties will then occur.

Although feedback is present in the first simulation result in terms of timing and friendship connections, we do understand that a network consisting of only growth produces results that are neither very realistic nor interesting in terms of network evolution, namely fully connected graphs. Nevertheless, ensuring that network growth occurs realistically within our simulation, we are then able to take the next step and begin to introduce conflict into our growing network.

Our model allows us to experiment with a number of conflict- and friendship-based probability parameters: 1) the probability of making a new friend, 2) the probability of meeting your friends friends, 3) the probability of conflict occurring between yourself and one of your friends, and 4) the probability of assessing your current friendships to check for conflict amongst your friends.

However, as one may assume, we find through our experimentation with these probabilities that not all of the possible values for each of the parameters produce realistic network structures this is something we highlight in more detail in the final subsection of this section. Therefore, in an effort to focus our attention upon the critical parameter values, our goal was to investigate which parameter settings produce realistic network structures and then determine the degree of sensitivity these parameters have in terms of their range and ability to maintain structural realism. The first step in this process was to search for fundamental characteristics of network structure, such as the Power Law distribution of network ties, which are often found within empirically observed networks. We use this characteristic node degree distribution as a guiding point for our attention and then looked to see if something in particular about this structural form helped to produce realistic network structures through the rules provided to our agents.

	Experiment		
Parameter	A	B	C
Probability of making new friends	0.90	0.33	0.50
Probability of meeting friends friends	0.90	0.33	0.50
Probability of assessing a friendship	0.90	0.10	0.10
Probability of conflict	0.01	0.01	0.01
Probability of conflict decay	1.00	0.75	0.50
Probability of friendship decay	0.00	0.75	0.50

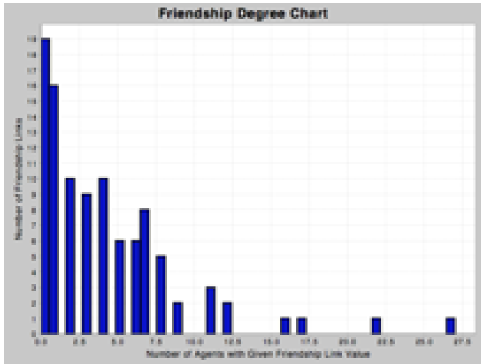
Table 1: Network Density Parameter Settings

The Power Law distribution of degrees marks a phase transition process within social networks because, with this critical characteristic intact, we are able to observe key percolation consequences in terms of structural realism. We find that a transition from a Power Law node degree distribution to a Normal or even a Log Normal distribution of friendship ties alters the percolation consequences of conflict within a network that also serve as realistic network evolution features.

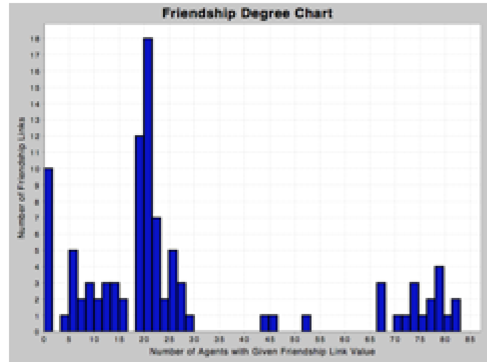
The above result is found by observing the node degree distribution at the point just before the onset of conflict. We see in figure 2 that, with conflict probability set quite low to allow the network to develop ties before conflict onset, a Normal or Log Normal node degree distribution before the onset of conflict results in a much greater drop in friendship ties and thus a larger percolation of conflict amongst the network. However, with the network in a Power Law distribution of network ties, the conflict is able to diffuse in most cases with infrequent but very large crashes occurring. Furthermore, if a network begins to develop friendship ties in a Power Law fashion, we also see that, through the help of minor conflicts, a network is able to develop a Normal or Log Normal distribution of friends, which is then followed by a crash and a return to a Power Law distribution of friendship ties. In other words, we begin to see a form of network self-sustainability in terms of node degree distribution and a fluctuation of network density around a critical point that is highlighted by the structural phase transitions.

3.2 Conflict Propagation

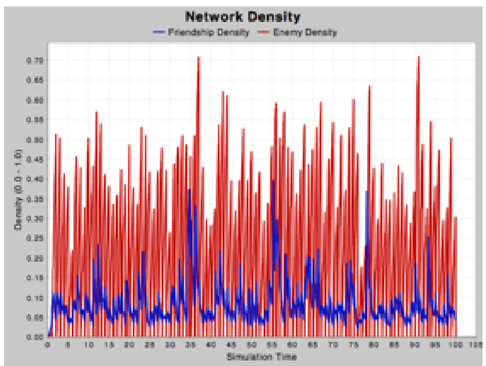
Now that idea of conflict percolation and phase transition has been introduced, we can now turn to experimenting with the conditions needed to produce realistic conflict percolation and realistic network structures as a result of this conflict percolation. In the results from figures 2 and 3 above, we see two important and realistic features of network evolution present within this run. Network First, we see the appropriate node degree distribution of friendship links, this being a Power Law distribution, on average throughout the simulation except for periods just before the spread of larger-scale conflicts. Second, we see a prominent critical density point. Upon reaching this point, the network settles in a dynamic equilibrium balancing introduction of new friendships and ongoing conflicts.



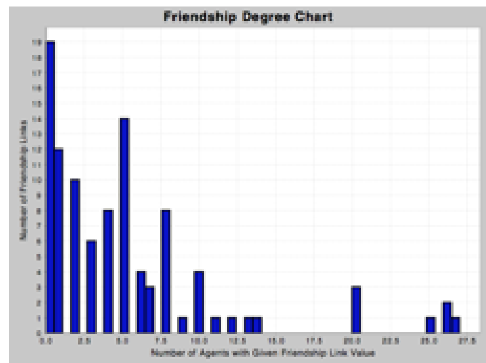
(a) Node degree distribution before minor conflict percolation



(b) Node degree distribution before medium to high conflict percolation



(c) Resulting network density. The largest friendship peaks represent point at which node distribution is either Log Normal or Normal



(d) Node degree distribution following large percolation of conflict

Figure 2: Propagation of Conflict in Networks - Effect on density and distribution of degrees

However, although our simulation value results, from figure 2 above, with a critical point hovering around a network density of roughly 0.10 is quite realistic, it is important to note here that these results come from a model with a very extreme and possibly somewhat unrealistic, in terms of most social networks, parameter probabilities (see Table 1, A).

It may not be so interesting that we were able to find realistic conflict percolation given a highly idealized friendship network. However, further runs of the model show us that this result is maintained under certain model conditions; some of which could arguably be considered empirically relevant. In figure 3.2a, we see that a critical point around 0.10 can be observed given quite different parameter settings (see Table 1, B). Figure 3.2b shows a similar result with even more moderate friendship fluctuation and higher friendship probabilities and lower decay probabilities (see Table 1, C). Therefore, these two results show that realistic conflict percolation can certainly develop within our model under conditions that would also be empirically plausible. However, through our experimentation we were able to find that not all parameter settings produce results that are quite as reasonable.

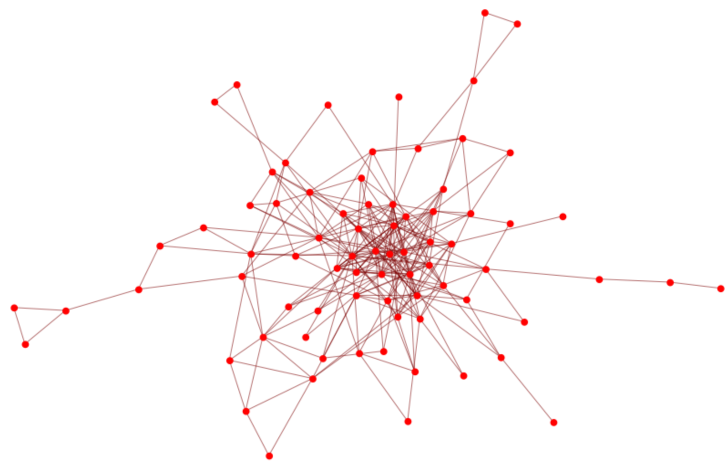
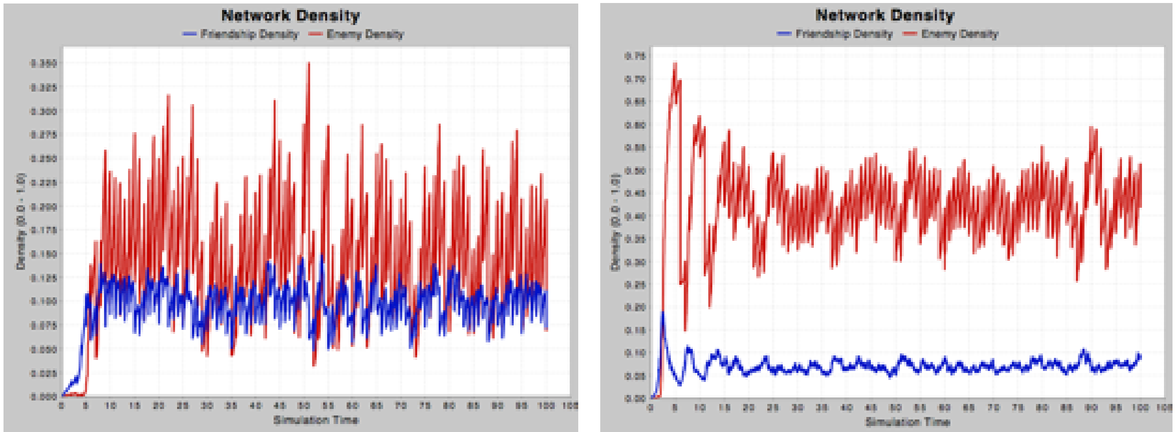


Figure 3: Typical Network Shape in low-conflict phase

3.3 Parameter Sweep

The models parameter sweep gives us a good understanding of the conditions needed to produce realistic network evolution behaviors and characteristics. For the most part, we found that a large number of network settings tended to produce network densities which either crashed all together or hovered around a critical point ≈ 0.02. The network crashes were typically found when either conflict probability was too high, conflict decay was too low, friendship decay was too high, or nodes were added to the system at a rate that was too frequent. A typical result of network crash is displayed in figure 5a. On the other hand, we found the network to survive crashes but to remain self-sustainable only at relatively low levels of network density. This result typically occurred when friendship creation probabilities were too low or when friendship assessment occurred too frequently. An example of low network criticality is displayed in figure 5b below.

In conclusion, we found that the key to producing realistic network evolution behaviours was not so much the value of the parameters themselves but the appropriate mixture of parameter value proportions in a given simulation. That is, realism could be found with either high conflict decay coupled with low conflict onset or with high tie decay in general along with low friendship assessment. Subsequently, we also found that conflict onset typically could not rise above 0.10 without disrupting the realism of the results. Furthermore, as described in the first subsection of this section, we found that the probability of making new friends from ones friends friends largely affected the overall volatility of the network density. Finally, for the range of successful or realistic simulation runs, realism was typically produced within a window of about only 10% change (5% above and below the values given above) in the aforementioned successful probability parameters.



(a) (b)

Figure 4: Sustainable network density at realistic parameter levels

4 Conclusion

In this paper, we demonstrate a simple agent-based simulation methodology that integrates destructive processes – conflicts – into the fabric of network evolution.

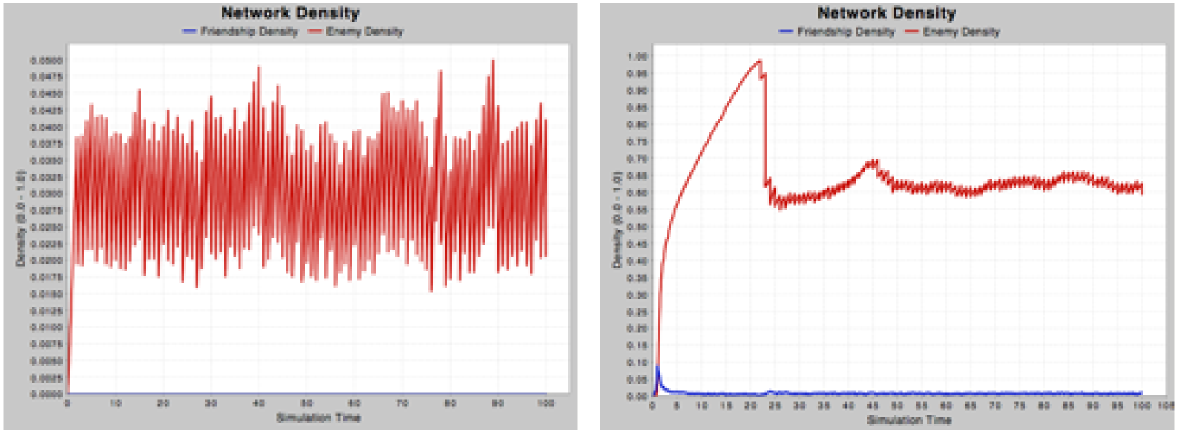
We also demonstrate two phase transitions in development of networks. The first phase transition occurs as network moves from linear growth and normally distributed degree (a-la Erdos) to exponential growth and power-law distributed degree of scale-free networks. This phase transition is percipitated by a single rule, and occurs at a critical density indepenent of network size or rate at which nodes are added.

At the same time as nodes are added, a conflict may strike a random pair of nodes with a constant probability. These conflicts propage through the network by agents seeking to be embedded in balanced triads. Thus in dense network structures a single conflict can possibly produce a large-scale avalanche of propagating conflict ties. Alternate periods of rapid growth and destruction signal a new phase transition - from a growing network to one that oscillates around a dynamic equilibrium while maintaining a relatively stable density. The densities achieved through our simulation strongly mimic these found in empirical networks.

We demonstrate that the resulting network has strong core-periphery features and a power-law distribution of degrees, yet is derived from a socially justifiable process. We also demonstrate that conflict is strongly localized and the scope of its propagation is Power Law distributed.

References

[1] Reka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Review of Modern Physics*, 74(1):47–97, 2002.



(a) Network crash - network cannot be sustained (b) Network sustained at high levels of conflict and very low density

Figure 5: Extreme conflicts in networks - is this the World at War that Hobbes predicted?

[2] P. Doreian, R. Kapuscinski, D. Krackhardt, and J. Szczypula. A brief history of balance through time. *Journal of Mathematical Sociology*, 21(1-2):113–131, 1996.

[3] Erdös and Rényi. On the evolution of random graphs. *Publication of Mathematics Institute of Hungian Academy of Sciences*, 5:1761, 1960.

[4] Terrill Frantz and Kathleen M. Carley. A formal characterization of cellular networks. Technical Report CMU-ISRI-05-109, Carnegie Mellon University School of Computer Science Institute for Software Research International, 2005.

[5] David Krackhardt. The ties that torture: Simmelian tie analysis in organizations. *Research in the Sociology of Organizations*, 16:183–210, 1999.

[6] Ravi Kumar, Jasmine Novak, and Andrew Tomkins. Structure and evolution of online social networks. In *Proceedings of KDD-2006*, 2006.

[7] J. Leskovec and J. K. C. Faloutsos. Graphs over time: Densification laws, shrinking diameters, and possible explanations. In *11th KDD*, pages 177–187, 2005.

[8] Mark Newman, Albert-László Barabási, and Duncan J. Watts. *The Structure and Dynamics of Networks*. Princeton University Press., Princeton, NJ, 2006.

[9] Camille Roth. Generalized preferential attachment: Towards realistic socio-semantic network models. In *ISWC 4th Intl Semantic Web Conference, Workshop on Semantic Network Analysis*, volume 171 of *CEUR-WS Series (ISSN 1613-0073)*, pages 29–42, Galway, Ireland, November 2005.

[10] M. Tsvetovat and K.M. Carley. Generation of realistic social network datasets for testing of analysis and simulation tools. Technical Report Technical Report CMU-ISRI-05-130,

Carnegie Mellon University, School of Computer Science, Institute for Software Research International, 2005.

[11] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, (393):440–442, 1998.

Organizational Theory and Practice

AGENT-BASED SIMULATION OF PRODUCT INNOVATION: MODULARITY, COMPLEXITY AND DIVERSITY

S.H. CHEN,* National Chengchi University, Taiwan
B.T. CHIE, National Chengchi University, Taiwan

ABSTRACT

The importance of modularity in product innovation is analyzed in this paper. Through simulations with an agent-based modular economic model, we examine the significance of the use of a modular structure in new product designs in terms of its impacts upon customer satisfaction and firms' competitiveness. To achieve the above purpose, the automatically defined terminal is proposed and is used to modify the simple genetic programming.

Keywords: agent-based computational economics, genetic programming, automatic defined terminals, modularity, hierarchy

MOTIVATION AND INTRODUCTION

This work is a continuation of [2, 3], which provide an agent-based model to simulate the evolution of product innovation by growing it from bottom up. The earlier work is not just to provide an agent-based approach, but to introduce a new representation of commodities, production, and preference, via the use of genetic programming (GP). However, [2, 3] only consider the simple genetic programming [5]. The end result is that in many of their early simulations, only primitive desires are satisfied, and the economy can rarely advance to a mature state where consumers' desires can be met to a sophisticated degree. One cause of this problem is that simple GP is not an appropriate tool to work with the idea of functional modularity (to be detailed in Section 3). This limitation has been long realized by GP researchers, e.g., [6]. In this paper, we remedy this problem by replacing the simple GP with automatically defined terminals (ADTs), which are very similar in spirit to automatically defined functions (ADFs), invented by John Koza [6]. As Koza pointed out, devices of this kind can provide some hierarchical mechanism to exploit modularities inherent in problem environments.

With this modified version of GP, two experiments are carried out. The first experiment examines the contribution of functional modularity to consumers' satisfaction. The second series of experiment then examines the importance of modularity in competition among firms. We simulate an agent-based economy to allow the firm who design new products using modular structure competing with the firm who do not. In a sense, this is equivalent to replicate the well-known story regarding the competition between Hora and Tempus, two imaginary watchmakers offered by Herbert Simon in his celebrated work on the architecture of complexity [8].

---

* Corresponding author address: Shu-Heng Chen, AI-Econ Research Center, Department of Economics, National Chengchi University, 64 Zhi-Nan Road, Sec. 2., Taipei, Taiwan; e-mail: chchen@nccu.edu.tw.

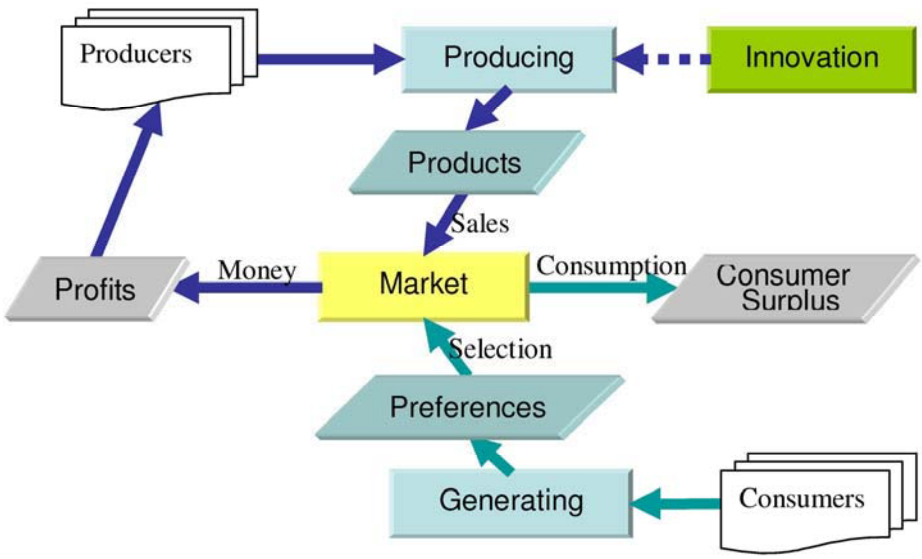


FIGURE 1 The Agent-Based Modular Economy Market

The rest of the paper is organized as follows. Section 2 provides a brief review of the agent-based modular economy introduced in [2, 3]. Section 3 proposes the automatically defined terminal and motivates this idea with its connection to hierarchical modularity. The two experiments are conducted in Sections 4 and 5. Section 6 gives the concluding remarks.

THE AGENT-BASED MODULAR ECONOMY

The economic model on which our simulation of product innovation is based is largely the same as [2, 3], which we shall briefly review here. Chen and Chie [2, 3] considered an economy composing of a number of firms and consumers with the usual objectives. To maximize profits, firms have the incentive to search for the products which can satisfy consumers to a highest degree. In the meantime, consumers allocate their limited budget to commodities which provide them largest degree of enjoyment (measured in terms of consumer surplus). The interaction between consumers and producers drives the evolution of a series of new products (innovation process), as shown in Figure 1.

The commodity of this economy is represented by a parse tree as shown in the first row of Figure 2. Each parse tree corresponds to a LISP program. The very bottom of tree, i.e., leaves, correspond to the raw inputs (materials) X1, X2,..., whereas the root and all intermediate nodes represent the processors, F1, F2,..., applied to these raw materials in a bottom-up order, as the usual behavior of a LISP program. The whole parse tree can, therefore, be interpreted as a production process associated with the commodity. The unit cost of the commodity is a positive function of the number of the processors and the number of raw inputs, i.e., a positive function of the (node) complexity of the commodity. In a simple way, we assume that the unit cost is a linear function of the node complexity.

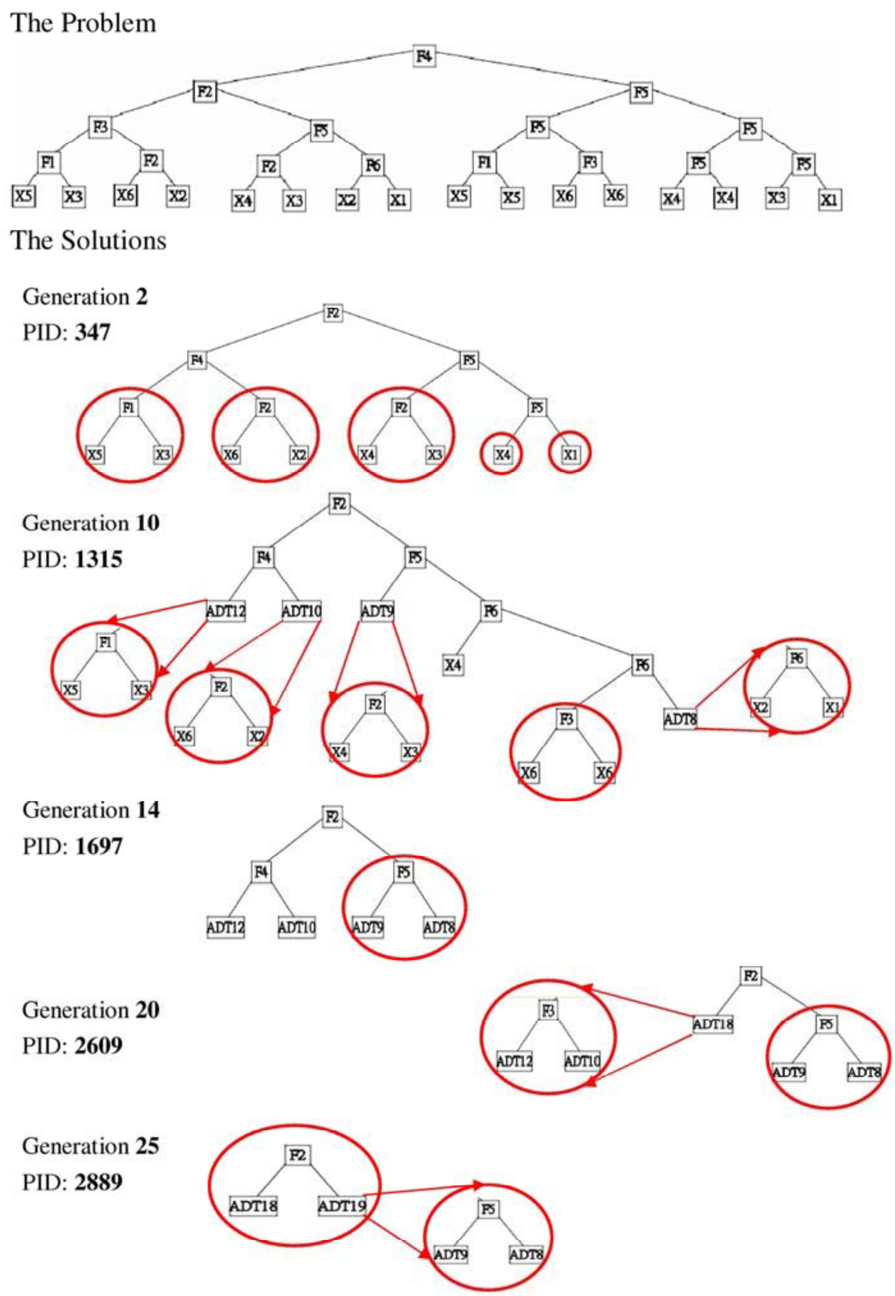


FIGURE 2 An Illustration of a Process of Product Innovation

In each market period, the firm has to decide how to allocate her limited working capital (budget) to R&D (designs of new products), production of existing commodities with different quantities, and reserves. R&D is the sole source of new products and is implemented by genetic programming with or without automatically defined terminals (to be detailed in Section 3), as shown in Figure 3.

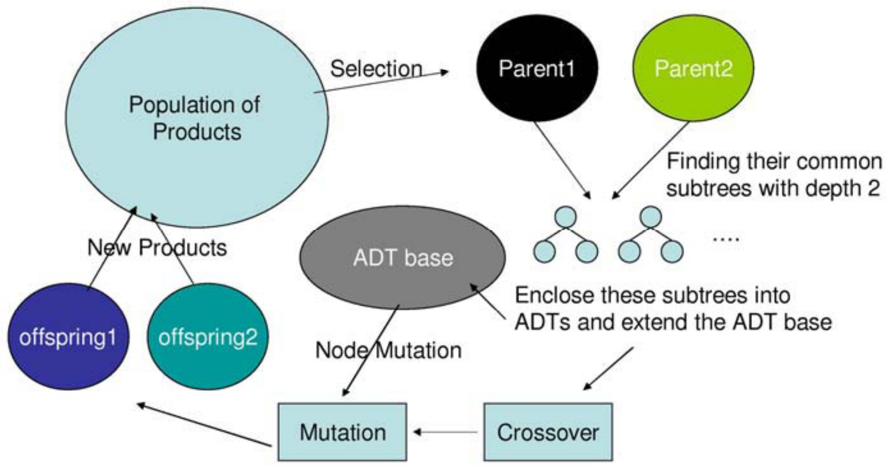


FIGURE 3 Genetic Programming as an Engine for R&D

The preference of the consumers in the economy is also represented by a parse tree. To make the preference tree be economically meaningful, three assumptions have been made [3], namely, the monotone, synergy, and consistency condition. The utility from consuming a commodity is based on the module-matching algorithm proposed in [3]. The idea is to match each possible module (subtree) of the commodity to each possible module of the preference with a descending order in the depth of the tree. So, the big modules matches first; if it success, we stop, and if it fails, we move to the next biggest one. To satisfy the synergy condition and hence the idea of added-value, [3] assumes a power utility function for the preference tree as shown in Figure 4. As a result, the utility is increasing at a faster rate when higher and higher levels of modular preferences are satisfied.

MODULARITY AND AUTOMATICALLY DEFINED TERMINALS

Simple genetic programming is designed to find the solution to a problem, which is represented by a parse tree. In our case, a solution is analogous to a product, and whose corresponding problem is the ideal product which can bring the highest enjoyment to a target consumer (Figure 2). The parse tree, from the bottom to the top, can be read as how the solution (product) can be constructed parallelly, incrementally and progressively. What is accomplished at each incremental and parallel step is a minimum or marginal effort to combine what have been established in the previous steps.

As an illustration, Figure 2 traces an artificial history of product innovation. Consider a target consumer whose preference is depicted in the first row of Figure 2, which can be regarded as the solution to the problem. Firms do not know this design, and have to figure out the best design by themselves. The five products listed below are the designs discovered in generations 2, 10, 14, 20, and 25. These products match the consumer's needs to a high and higher level. For example, the product PID 2889, i.e., the 2889th new product designed by the firm, has completely answered the target's need to the entire first half at level four. Nonetheless, this product does not come out all of a sudden; all it has done is to combine two commodities which were already known before, namely, commodities ADT 18 and ADT 19, both of which were already known to the firm before generation 25. The "marginal" effort here is to assemble them in a right way, i.e., using processor F2.[1]

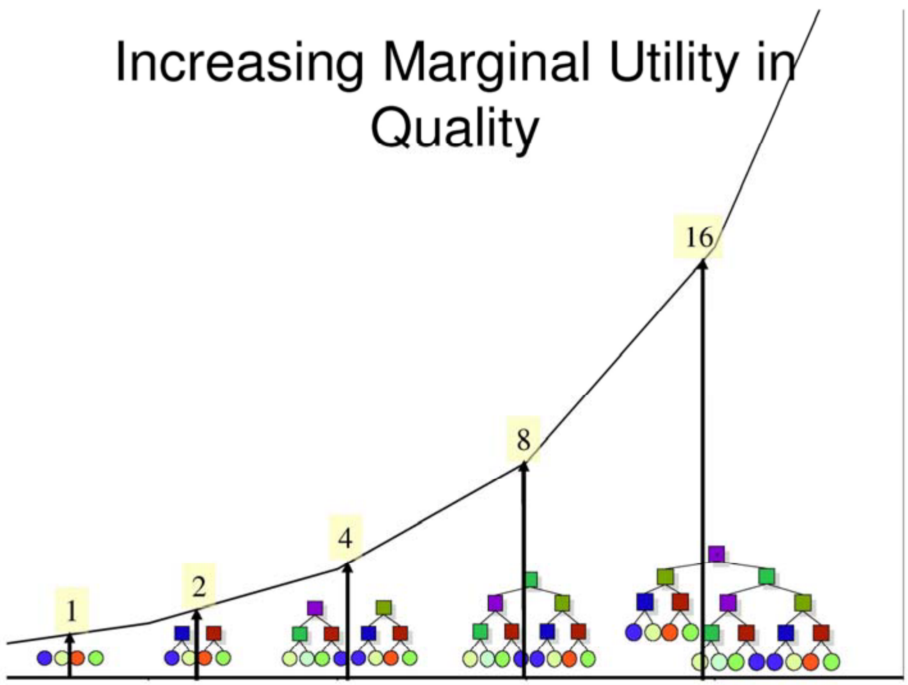


FIGURE 4 The Power Utility Function of a Preference Tree

The results obtained in each step then becomes the bases or the building blocks for the subsequent steps. For example, as shown in Figure 2, ADT 18 and ADT 19 serves as building blocks for all designs after generation 20. The entire process can then be viewed as a growing but convergent process from leaves to small trees, then to bigger and bigger trees, and finally to the target tree (the solution).

[1] Of course, from an ex ante view, knowing what to combine and in which way is not trivial. In fact, in this example, it took the firm five generations to learn this. In this sense, the contribution is not entirely marginal. However, from an ex post view, it is just a combination of what we already knew.

The description above enables us to see how genetic programming can be related to Simonian notion of complexity [8], i.e., hierarchy. Herbert Simon viewed hierarchy as a general principle of complex structures. Hierarchy, he argued, emerges almost inevitably through a wide variety of evolutionary processes, for the simple reason that hierarchical structures are stable. To demonstrate the importance of hierarchical structure or modular structure in production, Simon offered his well-known story about a competition between Hora and Tempus, two imaginary watchmakers. In this story, Hora prospered because he used the modular structure in his design of watches, whereas Tempus failed to prosper because his one is not modular. Therefore, the story is mainly about a lesson: the advantage of using modular structure in production.

While using parse tree as the representation, simple genetic programming is not good at using modular structure. The standard crossover and mutation can easily destroy the already established structure, which may cause the whole discovery or learning process non-incremental and non-progressive, and hence very inefficient. This problem is well-known in the GP literature, and has been extensively studied with various treatments [1, 6, 7, 4]. Motivated by these earlier studies, we propose automatically defined terminals (ADT) as a way to enhance GP to find structured solutions.

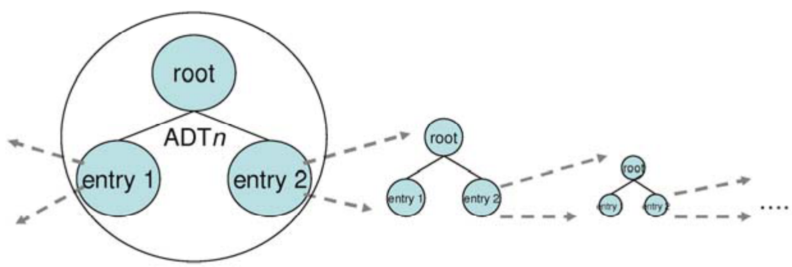


FIGURE 5 Automatically Defined Terminals

ADT, as shown in Figure 5, is very similar to the automatically defined function (ADF) [6]. Itself has a fixed structure, in this case, a tree with a depth of two. The root of ADT can be any function from the primitives (function set), while its leaf can be either a terminal from primitives (terminal set) or can be any existing ADTs. In this way, it shares the same spirit of ADF, namely, simplification, reuse, and encapsulation. The last part is particular important because it means whatever inside an ADT will not be further interrupted by crossover and mutation. In this way, ADTs can be considered as the part of learning in which we have great confidence, and leaves no room for doubt. Through ADTs we distinguish what is considered as knowledge from what is still in a trial-and-error process. Only the former can then be taken as the building blocks (modules), but not the latter. Without ADTs or equivalents, simple genetic programming essentially is not designed to develop building blocks; therefore, it is not very good at finding modular structure inherent in the problem.

MODULARITY AND CONSUMER SATISFACTION

Simple genetic programming can also detect modular structure, but it does it only by chance, and hence may be very difficult to detect complex modules.[2] To see this, in this section, we simulate how well consumers are served when the firm designs new products with modular GP (standard GP plus ADTs), and compare the result with that of standard GP.

In this simulation, there are 100 consumers in the market. Each consumer has a preference tree with a depth of six. Viewed from the most top level (the root level), the preference tree is composed of two modules. The one on the left, having a depth of five as shown in the first row of Figure 2, is identical among all consumers, whereas the one on the right, having a depth of five or less is heterogeneous, and is randomly generated by the ramped half-and-half method, an initialization method frequently used in GP. In this way, consumers' preferences have a common part as well as an idiosyncratic part. For the idiosyncratic part, the complexity is also different.

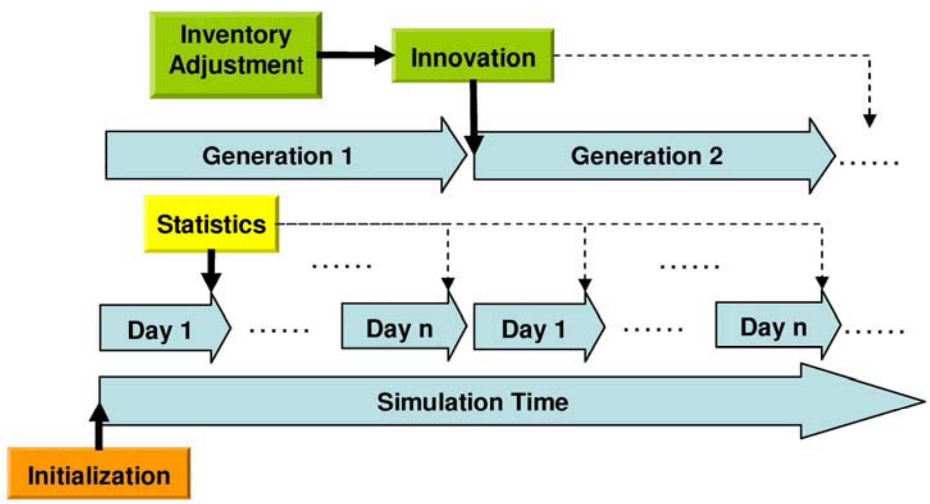


FIGURE 6 Market Days and Learning Cycles

A profit-maximizing monopoly firm will try to serve the needs of this group of consumers by producing different products with different quantities and also with different degrees of specialization or diversification (customization).[3] The firm has to learn the consumers' preferences and hence, through R&D (driven by GP), design better products. The entire market process is summarized in Figure 6. The learning cycle (GP cycle) is run with a number of generations (in our case, 5000). Each generation is composed of a number of trading days (in our case, five). After each learning cycle, the firm has to decide what to produce, including some new products developed via production innovation, how many to produce, and

---

[2] To define and measure complexity, Simon [8] advocated the use of a hierarchical measure – the number of successive levels of hierarchical structuring in a system or, in our case, the depth of the parse tree.

[3] See [3] for details.

how much to charge. The decision of production and R&D is based on the sales and profits statistics collected in the previous market days. The firm then supply what have been produced, including those new items, in the next few market days.

For further analysis, in each generation, statistics regarding consumer satisfaction are reported. Consumer satisfaction is measure by the actual utility the consumer received from consumption divided by the maximum potential utility the consumer can possible gain given his preference. The ratio is then multiplied by 1,000, and the measure lies in [0, 1000]. By averaging the consumer satisfaction over all 100 consumers, we then derive the aggregate consumer satisfaction, which also lies in the same interval. The result is shown in Figure 7.What Figure 7 shows is not the result based on a single run, but fifty runs. Accordingly, what is shown in the left panel of Figure 7 is the average of the 50 runs, whereas what is shown in the right panel is the maximum of the 50 runs. It can be seen quite easily, the firm whose product design uses modular structure can satisfy the consumers to a higher degree than the firm whose product design uses non-modular structure.

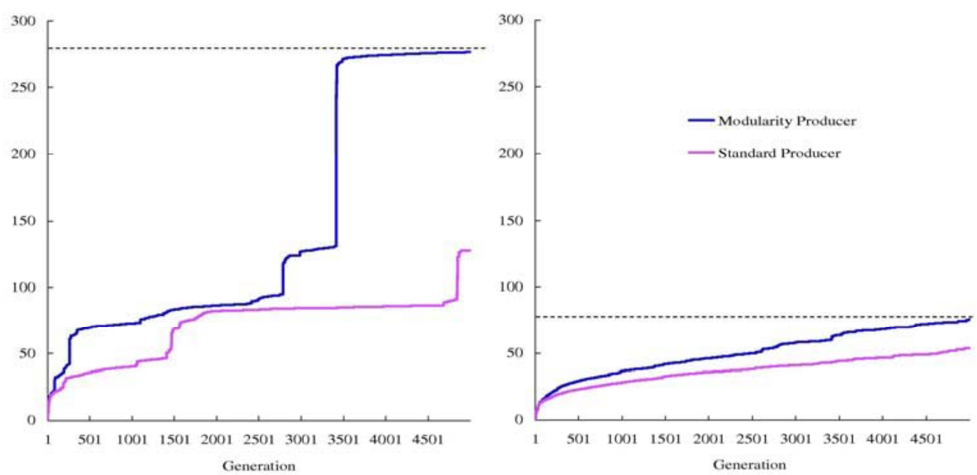


FIGURE 7 Modularity and Consumer Satisfaction

MODULAR STRUCTURE AND COMPETITIVENESS

In the previous section, under the assumption of a monopoly firm, we have seen the positive impact of using modular structure on consumer satisfaction. In this section, we shall pursue further by inquiring the implication of modular structure to the competitiveness of firms. In a sense, we attempt to re-examine the story given by Herbert Simon on the competition between two watchmakers: one using modular structure and one not. For that purpose, we consider a duopolistic competition in which one firm uses modular structure in her R&D (new product designs) and the other firm does not.

The two duopolistic firms compete with other in a market composing of 100 consumers whose preferences are partial identical and partial idiosyncratic (see Section 4). We then watch their market share, i.e., the total sales of each firm divided by the total sales of the market, and the result is displayed in Figure 8.[4] The result presented here is not based on a single run, but one hundred runs. The one shown in left panel of Figure 8 is the mean of the 100 runs, whereas the one shown in the right panel is the median of the 100 runs. Below the separation line is the market share owned by the non-modular firm, and above the line is the market share owned by the modular firm. Clearly, their sum equals 100%.

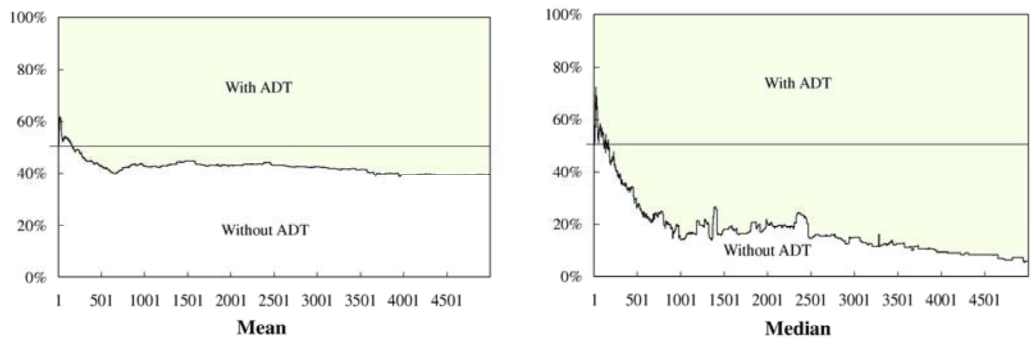


FIGURE 8 Modularity and Competitiveness

Due to the existence of outliers, the time series behavior of the mean and that of the median is not quite the same, but the eventual dominance of the modular firm is evident. In the first few hundreds of generations, non-modular firm, however, did have some competitive advantage over the modular firm. This is because establishing modules is costly. The idea of encapsulation associated with ADTs implies a fixed cost and hence a less degree of mobility, depending on the degree of encapsulation or the complexity of ADTs.[5] Hence, the modular products will generally be more expensive. Unless these products can fit the consumers' needs to a higher degree, these high-price products will exert adverse influence on marketing. Therefore, there is no guarantee (a probability of one) that the modular firm will always beat the non-modular firm. In fact, in 39 out of our 100 runs, the non-modular firm took up a higher market share than the modular firm in the last generation (the 5000th generation).

Finally, as one may expect, competition does bring a better quality to consumers. This is reflected in Figure 9.

CONCLUDING REMARKS

Consumers are not random and their behavior can be studied and patterns can be extracted. On the other hand, innovation normally is not a random jump, but follows a gradually changing process. These two together suggests that the economy can be constructed in a modular way, or the entire economy has a modular structure. In other words, Herbert Simon's notion of the architecture of complexity has the potential to be applied to the whole economy. In this paper, we study the significance of modularity in product innovation. We find that both consumers and

---

[4] Notice that firms generally produce more than one product and can be very different between each other. Therefore, it is meaningless to measure the market share based on a single product.

[5] See footnote (2) for the measure of complexity.

producers can benefit from the use of modular structure in product design. However, modularity may imply a higher cost and less mobility; therefore, its dominance is not certain. Using Simon's story, there is a chance that Tempus prospers and Hora fails.

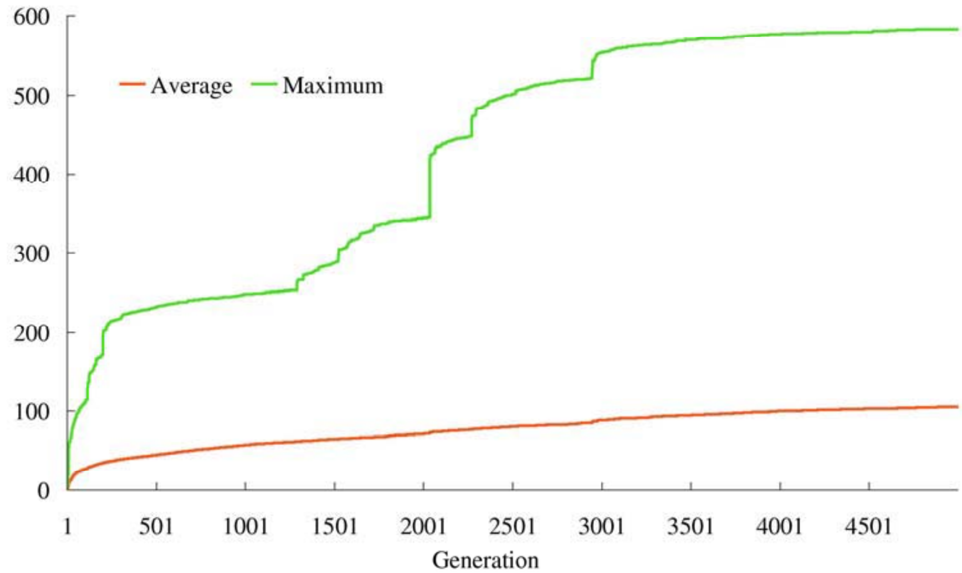


FIGURE 9 Consumer Satisfaction under Competition

ACKNOWLEDGEMENT

The authors are grateful to Herbert Dawid, Christophre Georges, and Massimo Ricottilli for their helpful discussions. NSC research grant No. 95-2415-H-004-002-MY3 is gratefully acknowledged.

REFERENCES

1. Angeline, P. and J. Pollack, 1993, "Evolutionary Module Acquisition," in *Proceedings of the 2nd Annual Conference on Evolutionary Programming*, MIT Press, pp. 154-163.

2. Chen, S.-H. and B.-T. Chie, 2004, "Agent-Based Economic Modeling of the Evolution of Technology: The Relevance of Functional Modularity and Genetic Programming," *International Journal of Modern Physics B*, Vol. 18, No. 17-19, pp. 2376-2386.

3. Chen, S.-H. and B.-T. Chie, 2005, "A Functional Modularity Approach to Agent-based Modeling of the Evolution of Technology," (with B.-T. Chie), in A. Namatame, T. Kaizouji, and Y. Aruka. (eds.), The Complex Networks of Economic Interactions: Essays

in Agent-Based Economics and Econophysics, *Lecture Notes in Economics and Mathematical Systems 567*, Springer, pp. 165-178.

4. Hoang, T.-H., E. Daryl, R. McKay, and X. H. Nguyen, 2007, “Developmental Evaluation in Genetic Programming: The TAG-Based Frame Work,” forthcoming in *Knowledge Engineering Review.*

5. Koza, J. R., 1992, *Genetic Programming: On the Programming of Computers by the Means of Natural Selection*, Cambridge, MA: The MIT Press.

6. Koza, J., 1994, *Genetic programming II–Automatic Discovery of Reusable Programs*, Cambridge, MA: The MIT Press.

7. Rosca, J. and D. Ballard, 1994, “Hierarchical Self-Organization in Genetic Programming,” In C. Rouveirol and M. Sebag (eds), Proceedings of the Eleventh International Conference on Machine Learning, Morgan Kaufmann.

8. Simon, H. A., 1965, “The Architecture of Complexity,” *General Systems*, 10, pp. 63-76.

9. Simon, H. A., 1973, “The Organization of Complex Systems,” in: H. H. Pattee (ed.), *Hierarchy Theory. The Challenge of Complex Systems*, New York: George Braziller, 1973, pp. 1-27.

THE EVOLUTION AND PERSISTENCE OF DOMINANT ROLES IN INTERORGANIZATIONAL RELATIONSHIPS

V.A. BARGER,* University of Wisconsin–Madison

ABSTRACT

Recent application of role theory to economic behavior (Montgomery 1998) has provided new insights into interorganizational relationships (Heide and Wathne 2006). In particular, role theory offers a framework for investigating the source of seemingly contradictory accounts of economic exchange, including Uzzi's (1997; 1996) finding that embeddedness enhances firm survival in the apparel industry and Wathne, et al.'s (2001) discovery that embeddedness does not insulate a firm from price competition in the commercial banking industry. The key to understanding these discrepancies lies in the divergent evolution of dominant relationship roles. This paper investigates the evolution and persistence of roles in interorganizational relationships from a role-theoretic perspective using agent-based modeling.

Keywords: interorganizational relationships, role theory, economic behavior, agent-based modeling

INTRODUCTION

According to the classical and neoclassical schools of thought, economic transactions are coordinated through the mechanism of price; that is, the totality of information necessary for exchange is communicated by the price associated with the transaction. Moreover, economic actors are rational, utility maximizing, and self-interested. Granovetter (1985) criticizes this view of economics as "undersocialized," in that it "disallow[s] by hypothesis any impact of social structure and social relations on production, distribution, or consumption." He contends that economic behavior is "embedded" in social relations and that these relations have a significant impact on how actors behave. This, then, is the "problem of embeddedness": that "behavior and institutions to be analyzed are so constrained by ongoing social relations that to construe them as independent is a grievous misunderstanding" (Granovetter 1985).

Granovetter's hypothesis is supported by empirical research in the apparel industry. Uzzi (1997; 1996) found that firms that rely on arm's length market transactions are more likely to fail than are firms that leverage social relations. He attributes this outcome to three features of embedded transactions: trust, information sharing, and joint problem solving. More recently, however, Wathne, Biong, and Heide (2001) uncovered evidence to the opposite effect: in the commercial banking industry, social relations are ineffective at protecting firms from price and product competition. Although social relations can create a barrier to switching, they are outweighed by firm-level switching costs and competitor superiority in price or product breadth.

* Corresponding author address: Victor Barger, University of Wisconsin–Madison, 975 University Ave, Madison, WI 53706; e-mail: vbarger@bus.wisc.edu.

Montgomery (1998) offers a role-theoretic conception of embeddedness that may help resolve this apparent contradiction. He attributes discrepancies in economic behavior to the roles elicited by relationships: (1) the role of "businessperson," who maximizes profit, and (2) the role of "friend," who cooperates out of principle. Which role is invoked in a particular transaction depends on the history of the relationship. Heide and Wathne (2006) extend this role-theoretic perspective to the governance of interorganizational relationships, proposing a conceptual framework that links governance strategies to roles and role activation. They identify two specific areas for future research: (1) the connection between roles and profitability, and (2) the sources of "stickiness" of roles.

The purpose of this research is to explore the consequences of embeddedness from a role-theoretic perspective. In particular, we seek to explain why certain industries exhibit a high level of embeddedness, whereas others favor arm's length market transactions. To this end, we investigate how roles can become dominant in an industry and, once established, influence the profitability of industry players. Greater understanding of how roles evolve within industries and the implications of an established, dominant role will help inform the governance decisions for firms entering or competing in such industries.

In the following section, we expand on our discussion of the role-theoretic perspective of economic behavior and introduce hypotheses on the evolution and persistence of roles in interorganizational relationships. We then propose an agent-based model for testing the hypotheses. Finally, we present our findings and discuss their implications and limitations.

CONCEPTUAL FRAMEWORK

Uzzi (1997) classifies business relationships as either "arm's length" or "embedded." Arm's length relationships are characterized by non-repeated transactions where the emphasis is on price. Performance is assured through contracts and monitoring, and problems are solved by switching suppliers. In contrast, embedded relationships are characterized by repeated transactions where price is but one consideration. Personal relations play a key role in business decisions, and relationships are governed by trust and reciprocity. As a result, embedded relationships benefit from higher levels of information sharing and joint problem solving. In other words, embedded relationships encourage cooperation, whereas arm's length relationships discourage it.

In role-theoretic terms, arm's length and embedded relationships are the result of firms assuming the roles of "businessperson" and "friend," respectively (Montgomery 1998). The businessperson follows the "logic of consequences" and seeks to maximize utility, while the friend follows the "logic of appropriateness" and behaves according to rules and norms (March 1994). Thus a friend favors cooperation even if it results in short-term loss, while the businessperson defects if doing so results in the highest gain. Different roles may be invoked at different times in different relationships.

Which role an actor assumes in a relationship depends on the disposition of the actor and the history of interactions between the actor and the other party (Heide and Wathne 2006). Montgomery (1998) conceptualizes this as a threshold model: for each party an actor interacts with, the actor maintains a mental record of "degree of friendship" (see Figure 1). When the actor interacts with another party, his degree of friendship with the other party increases or decreases depending on the characteristics of the interaction. For example, if the other party goes above and beyond the letter of a contract to help the actor, the degree of friendship will increase. If, on the other hand, the other party insists on compliance with the terms of a contract despite unforeseen difficulties, the degree of friendship will decrease. When the degree of friendship exceeds a certain level, the "friendship threshold," the actor assumes the role of friend. If the degree of friendship is beneath the friendship threshold, the actor assumes the role of businessperson.

The predisposition of an actor to assume the role of businessperson or friend is reflected in the actor's friendship threshold. An actor who is quick to befriend others will have a low friendship threshold, whereas an actor who prefers to stay at arm's length will have a high friendship threshold. The initial degree of friendship is determined by the mechanism through which the actors are first introduced. Consistent with Uzzi's (1996) findings, if the actors are introduced by a mutual friend, the initial degree of friendship will be relatively high. If instead the actors meet through search in an anonymous market, the initial degree of friendship will be low.

This conceptualization of economic behavior as a series of roles can be extended to an industry of firms. An embedded market can be viewed as one in which the dominant relationship role—i.e., the role assumed by the greatest number of firms in the industry—is that of friend. An anonymous market, in contrast, is one in which the dominant relationship role is that of businessperson. Moreover, we can speculate as to the type of industry that will arise given an initial level of embeddedness. Based on Uzzi's (1996) finding that "embedded ties can originate from anonymous market ties, but this source of embeddedness is uncommon in [the apparel] industry," we hypothesize that:

> **H_{1A}**: The dominant relationship role that will evolve in an industry that is initially devoid of embeddedness (i.e., an anonymous market) will be that of businessperson.

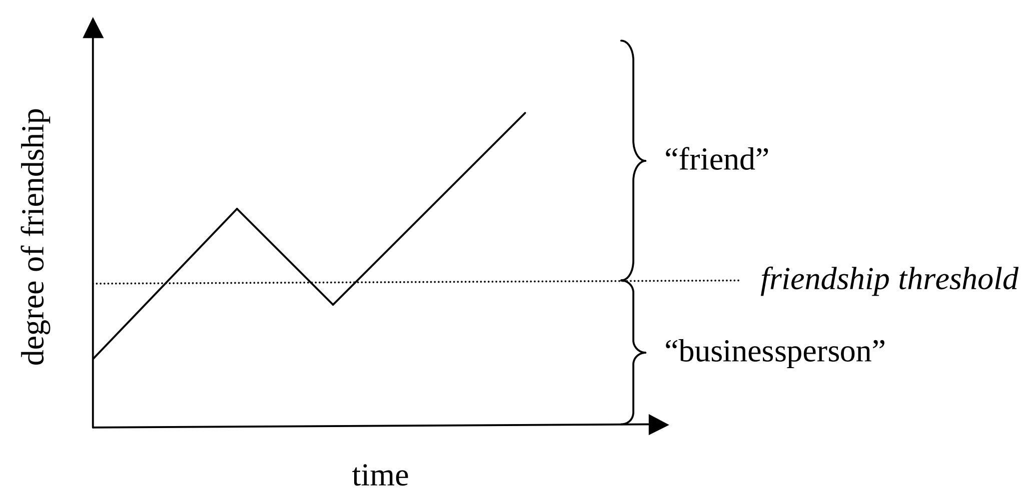


FIGURE 1 Threshold model of role determination

A market may be embedded from the outset when business relationships have been primed for embeddedness by third-party referrals or personal relationships (Uzzi 1996). As the initial level of embeddedness in an industry increases, we expect to see a greater number of firms adopting the role of friend. Moreover, we hypothesize that there is a critical initial level of embeddedness which, if exceeded, will lead to the evolution of an embedded industry. In other words:

H_{1B}: The dominant relationship role that will evolve in an industry initially characterized by a level of embeddedness above a critical point will be that of friend.

Uzzi (1996) demonstrated that in the apparel industry, firms that preferred embedded relationships over arm's length relationships were more likely to survive. Since the apparel industry most closely resembles an embedded market, we can infer that, in embedded markets, firms that are predisposed to assume the role of friend will outperform firms that operate as businesspersons. On the other hand, in an anonymous market it is predicted that firms that are predisposed to the role of businessperson will outperform firms that prefer to act as friends, since in such markets friends will not realize gains due to cooperation and, moreover, are likely to be taken advantage of by businesspersons. Therefore, we predict that:

H_{2A}: In an embedded market, firms that prefer the role of friend will outperform firms that prefer the role of businessperson.

H_{2B}: In an anonymous market, firms that prefer the role of businessperson will outperform firms that prefer the role of friend.

METHODOLOGY

Similar to Montgomery (1992), we model firm behavior as a repeated prisoner's dilemma. Rather than analyze the game mathematically, however, we employ agent-based modeling. This enables us to investigate emergent properties of the system (Axelrod and Tesfatsion 2006), which is essential in a model such as ours, where the behavior of actors is dependent on past experience and the actors continually adapt based on their interactions. Moreover, agent-based modeling permits experimentation with a heterogeneous population of actors (Axtell 2000), in this case a population of firms with different roles, predispositions, relationships, and histories.

Our model specification is presented below. This is followed by a description of how core aspects of the model are operationalized. Finally, the findings from simulations of our agent-based model are reported.

Model Specification

Two types of firms are represented in the model: manufacturers and suppliers. Manufacturers purchase materials from suppliers to use in the manufacturing process, and suppliers sell these materials to manufacturers to earn a profit. Interactions between manufacturers and suppliers take the form of a repeated prisoner's dilemma (see Figure 2). Each period, the manufacturers choose the suppliers from whom to purchase materials. A supplier can either "cooperate" by delivering the product as expected or "defect" by shirking on product quality or quantity. Similarly, a manufacturer can either "cooperate" by paying the full amount for the product or "defect" by demanding allowances from the supplier. Note that only the interactions of *principal* exchange partners are modeled. That is, the supplier selected by a manufacturer is assumed to be the one that supplies the most product to that manufacturer in that period. This does not preclude the possibility of a manufacturer obtaining materials from multiple suppliers; however, including secondary sources in the model could underestimate the effect of embeddedness (Uzzi 1996).

The interaction history between a manufacturer and a supplier is encapsulated in their "degree of friendship" (Montgomery 1992). When a manufacturer's degree of friendship with a supplier exceeds the manufacturer's "friendship threshold" (see Figure 1), it will assume the role of "friend" when interacting with the supplier. If, on the other hand, the manufacturer's degree of friendship is less than or equal to its friendship threshold, it will assume the role of "businessperson." Likewise, each supplier has a degree of friendship with each manufacturer. When the supplier's degree of friendship with a manufacturer exceeds the supplier's friendship threshold, it will assume the role of "friend" when interacting with the manufacturer. When the supplier's degree of friendship is less than or equal to the supplier's friendship threshold, it will assume the role of "businessperson." After each exchange, the firms update their respective degrees of friendship to reflect the outcome of the transaction. If the other party cooperated, the firm increases its degree of friendship with the other party. If instead the other party defected, the firm decreases its degree of friendship with the other party. Thus firms are capable of role switching (Heide and Wathne 2006; Montgomery 1992) based on their interaction histories.

		Supplier	
		C_2 ("delivers as promised")	D_2 ("shirks on quality or quantity")
Manufacturer	C_1 ("pays in full")	5, 5	3, 6
	D_1 ("demands allowances")	6, 3	4, 4

FIGURE 2 Payoff matrix for manufacturer and supplier in two-sided prisoner's dilemma

How the parties to a transaction behave (i.e., whether they cooperate or defect) determines the payoffs each receives from the transaction (see Figure 2). The firms' behavior, in turn, is a function of the role assumed: a "friend" cooperates out of principle, while a "businessperson" opts for short-term gain and defects (Montgomery 1992). If both manufacturer and supplier cooperate, the manufacturer receives the full value of its purchase ($U_1(C_1,C_2) = 5$) and the supplier receives full payment ($U_2(C_1,C_2) = 5$). If the manufacturer cooperates and the supplier defects, the manufacturer receives less than the full value of the purchase ($U_1(C_1,D_2) = 3$) due to the supplier's shirking on quality or quantity, and the supplier receives more than fair compensation ($U_2(C_1,D_2) = 6$) due to savings on materials and labor. If, instead, the manufacturer defects and the supplier cooperates, the manufacturer receives more than the full value of the purchase ($U_1(D_1,C_2) = 6$) by demanding allowances for defective merchandise when the product is in fact satisfactory, while the supplier receives less than fair compensation ($U_2(D_1,C_2) = 3$). Finally, if both manufacturer and supplier defect, the manufacturer receives less than the full value of the purchase ($U_1(D_1,D_2) = 4$) due to the opportunity cost of not having necessary supplies on hand, and the supplier loses potential revenue ($U_2(D_1,D_2) = 4$) due to allowances granted the manufacturer. Note that the payoff matrix satisfies the requirements for a two-sided prisoner's dilemma (Axelrod 1984), since $U_1(D_1,C_2) > U_1(C_1,C_2) > U_1(D_1,D_2) > U_1(C_1,D_2)$, $U_2(C_1,D_2) > U_2(C_1,C_2) > U_2(D_1,D_2) > U_2(D_1,C_2)$, and $U_1(C_1,C_2) > (U_1(D_1,C_2) + U_1(C_1,D_2))/2$.

To decide which supplier to purchase materials from, a manufacturer starts by ranking its suppliers by degree of friendship. If the manufacturer is friends with one or more suppliers (i.e., the manufacturer's degree of friendship with the supplier exceeds the manufacturer's friendship threshold), the manufacturer will choose the supplier with which it has the highest degree of friendship. If several suppliers have equally high degrees of friendship, the manufacturer will randomly select one. If the manufacturer is not friends with any suppliers (i.e., the manufacturer's degree of friendship is below its friendship threshold for all suppliers), the manufacturer will randomly select a supplier. This corresponds to a businessperson choosing the supplier with the lowest price.

Operationalization

Three aspects of the model require further elaboration with respect to how they are operationalized: (1) role activation, (2) role preference, and (3) market embeddedness.

Role activation. The degree of friendship between two firms (see Figure 1) is expressed as an integer from zero to ten, with zero corresponding to the absence of friendship and ten representing the highest possible level of friendship. Likewise, the friendship threshold is also expressed as an integer from zero to ten. When degree of friendship exceeds a firm's friendship threshold, the firm assumes the role of "friend." When degree of friendship is less than or equal to a firm's friendship threshold, the firm assumes the role of "businessperson." Note that friendship can be asymmetric, in that a manufacturer's degree of friendship with a supplier may differ from the supplier's degree of friendship with the manufacturer; moreover, the two firms may have different friendship thresholds. Thus a manufacturer may assume the role of friend with a supplier who assumes the role of businessperson or vice versa.

Role preference. The preference of a firm for the role of friend or businessperson is indicated by the firm's friendship threshold. A high friendship threshold indicates a firm that prefers the role of businessperson, since the firm requires a higher degree of friendship with another firm before it will assume the role of friend. In contrast, a firm with a low friendship threshold is characterized as preferring the role of friend, since a lower degree of friendship with another firm is needed for it to assume the role of friend. Friendship thresholds are presumed to be normally distributed in the population, and thus are randomly assigned to firms from a normal distribution with $\mu = 5$ and $\sigma = 1$.

Market embeddedness. The embeddedness of a market refers to the extent to which economic behavior is influenced by social relations (Uzzi 1996). In an anonymous, arm's length market, economic behavior is completely unaffected by social relations. In contrast, in a wholly embedded market, economic behavior is completely determined by social relations. Although neither of these may be plausible *per se*, they are valuable from a theoretic standpoint as the endpoints of a continuum of markets, from the anonymous to the wholly embedded; as the level of embeddedness increases, so does the influence of social relations on economic behavior within the market.

In this study, the level of embeddedness of a market is specified on a scale of zero to ten, where zero represents an anonymous market and ten corresponds to a wholly embedded market. To simulate an anonymous market, the initial degrees of friendship of all firms are set to zero. Similarly, to create a wholly embedded market, the initial degrees of friendship are set to ten. Between the two extremes, the initial degrees of friendship are randomly assigned from a Weibull distribution, where the parameterization of the distribution depends on the level of embeddedness (see Figure 3). Thus as the level of embeddedness increases, so does the probability of a firm being assigned a high initial degree of friendship with another firm. (For embeddedness greater than five, the mirror image of the distributions shown in Figure 3 are utilized. For instance, a market with a level of embeddedness equal to seven is generated from the mirror image of the Weibull distribution for embeddedness level three.) Note that each of the Weibull distributions has a mean equal to the level of embeddedness. For example, the expected value of a random variable distributed as Weibull(2.2, 4.52) is four, the same as the level of embeddedness represented by that distribution.

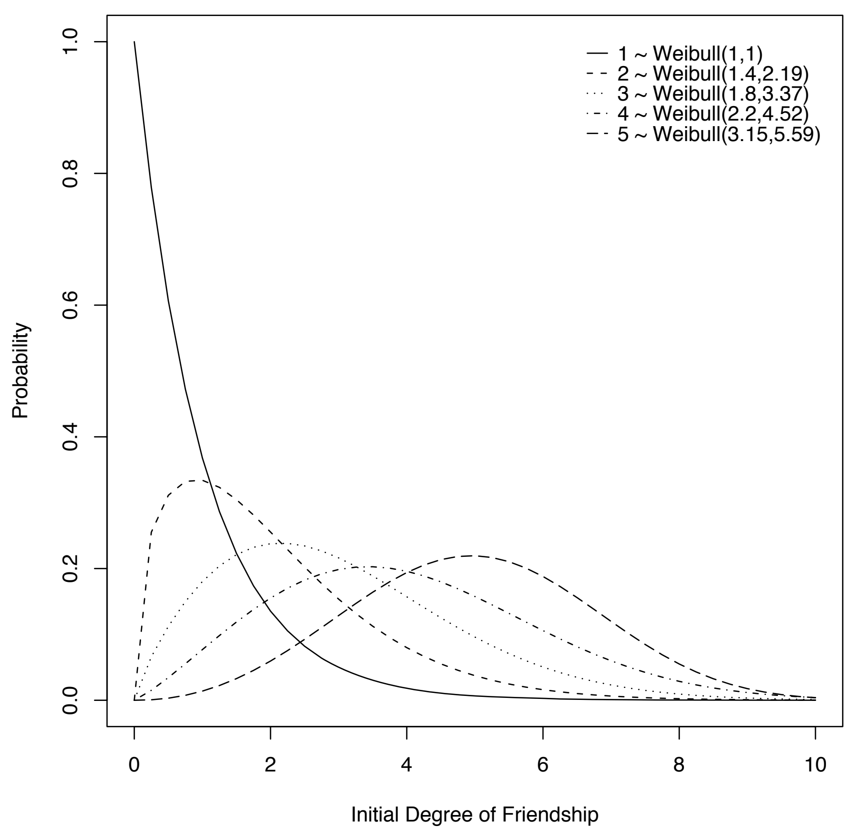


FIGURE 3 Market embeddedness probability distributions

Results

The model was programmed in Java and simulated with REPAST, an open source agent-based modeling environment (North et al. 2006). Model parameters were determined by the hypothesis being tested, as discussed below. For each set of parameters, the model was run twenty times, and results from batches of runs were aggregated for analysis. In each run, a market comprised of twenty manufacturers and twenty suppliers was simulated for 250 periods.

To test Hypotheses 1A and 1B, markets with initial levels of embeddedness from zero to ten were simulated. Each period, the number of firms that assumed the role of "friend" in a transaction was recorded; these data are graphed as percentages in Figure 4 (the numbers in italics indicate the level of embeddedness of the market at outset). In support of Hypothesis 1A, firms that start in an anonymous market assume the role of businessperson throughout the simulation. Similarly, firms that assume the role of businessperson in markets with initial levels of embeddedness equal to 1, 2, and 3 are consistently in the majority. Averaged over the last fifty periods, they account for 100%, 94.5%, and 68.5% of the firms participating in transactions, respectively. In contrast, firms assuming the role of friend in markets with an initial level of embeddedness greater than or equal to 4 quickly become the majority. Averaged over the last fifty periods, they account for 72.8%, 93.4%, and 100% of the firms participating in transactions, respectively. Thus Hypothesis 1B is supported.

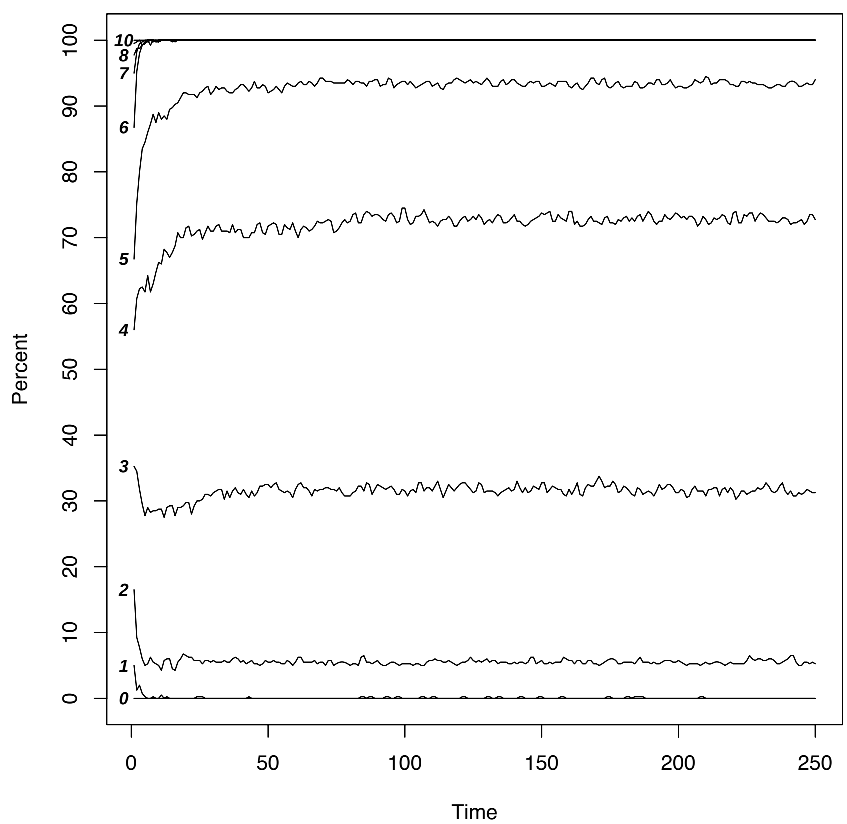


FIGURE 4 Percent active firms in friend role

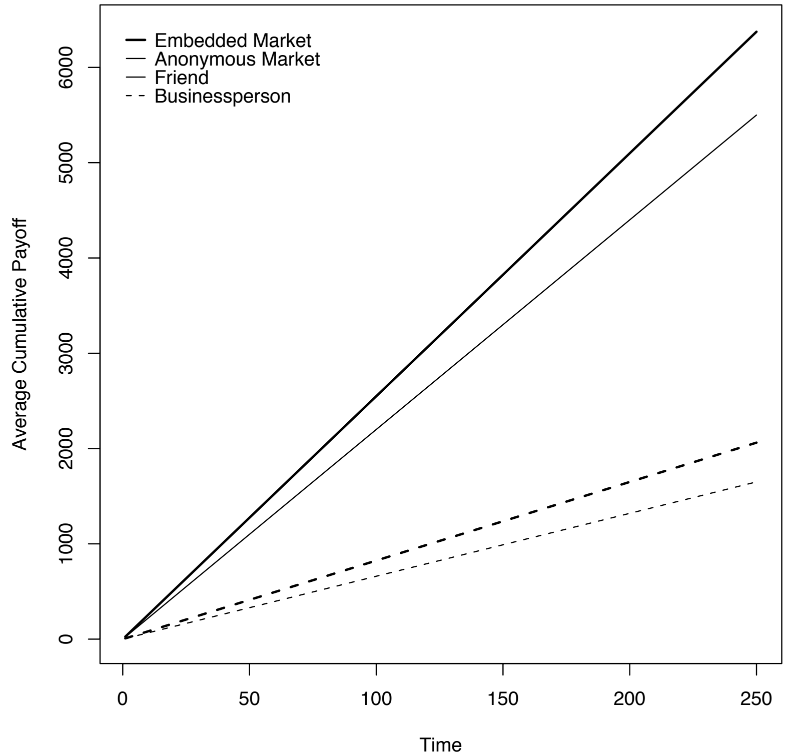


FIGURE 5 Manufacturer payoffs

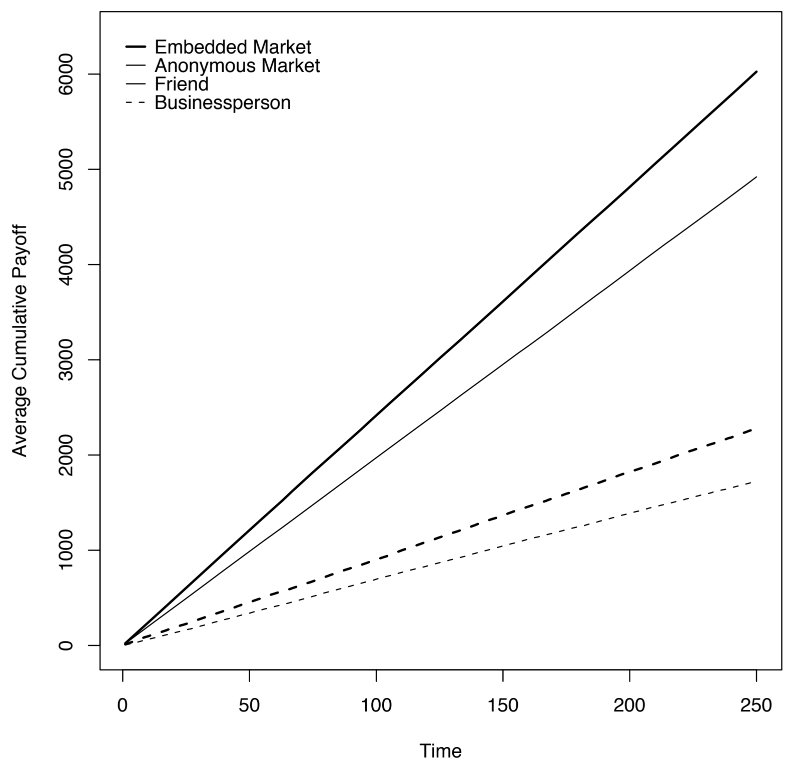


FIGURE 6 Supplier payoffs

To test Hypotheses 2A and 2B, two types of markets were simulated: an anonymous market and an embedded market (level of embeddedness = 8). Each period, the cumulative payoffs for firms with friendship thresholds greater than five (i.e., firms that prefer the role of "businessperson") and for firms with friendship thresholds less than five (i.e., firms that prefer the role of "friend") were recorded. The data for manufacturers are graphed in Figure 5 and the data for suppliers are graphed in Figure 6. Surprisingly, firms that assume the role of "friend" outperform firms that assume the role of "businessperson" in *both* anonymous and embedded markets, supporting Hypothesis 2A and disconfirming Hypothesis 2B. In the embedded market, manufacturer "friends" realize an average final payoff of $6,374, compared to $2,062 for "businesspersons." Similarly, in the anonymous market, manufacturer "friends" earn a total of $5,500, while manufacturer "businesspersons" earn only $1,650. For suppliers, "friends" in the embedded market receive $6,026, compared to $2,278 for "businesspersons". In the anonymous market, "friends" receive $4,921, compared to $1,724 for "businesspersons."

It is informative to consider the relative increases in profitability that are achieved by adopting the role of "friend" versus "businessperson" or by functioning in an embedded versus anonymous market. Manufacturers are over three times as profitable (3.1 in the embedded market and 3.3 in the anonymous market) when they assume the role of friend instead of the role of businessperson, while suppliers are between 2.6 and 2.9 times as profitable. With respect to market type, manufacturer "friends" realize a 16% gain by operating in an embedded market instead of an anonymous market, and manufacturer "businesspersons" achieve a 25% gain. Suppliers do even better in embedded markets; they realize a 22% gain when in the role of "friend" and a 32% gain when in the role of "businessperson."

DISCUSSION

Our model shows that the conditions surrounding the development of a market have a powerful influence on the ultimate characteristics of the market. Namely, a market primed for embedded relations will tend to evolve into a market dominated by "friends," whereas a market without such priming will be dominated by "businesspersons." These dominant relationship roles in turn determine the form of the market. When the majority of the firms assume the role of "friend," the market is highly embedded. When the majority assume the role of "businessperson," however, the market is more accurately portrayed as "anonymous" or "arm's length."

Beyond providing insight into why markets differ in embeddedness, our model suggests that certain relationship roles are superior from a profitability standpoint regardless of the level of embeddedness. Assuming the role of "friend" not only results in greater profit in embedded markets, it benefits firms in anonymous markets, too. In fact, firms in anonymous markets realize a larger increase in profitability by assuming the role of "friend" than do firms in embedded markets. This runs counter to the popular notion that a business in a cutthroat industry must itself behave in a cutthroat manner. On the contrary, the business may achieve superior returns by adopting the role of "friend."

Our agent-based model offers insights into the evolution and persistence of interorganizational relationship roles. Empirical verification of the predictions of the model are important for providing support for the model's external validity. This may prove difficult, however, since data on the initial conditions of markets, particularly with respect to embeddedness, is not easily obtained.

Two additional aspects of the model deserve further attention. First, the model focuses on the impact of initial market characteristics. It is conceivable, though, that shocks to the system that occur at a later time may also have a significant impact on the dominance of relationship roles. Second, the model is built on the assumption that embeddedness discourages opportunism. However, it has been suggested that firms may expose themselves to increased opportunism if they become overembedded (Granovetter 1985; Uzzi 1997; Uzzi 1996). The model could be extended to explore the impact of overembeddedness on firm performance.

REFERENCES

Axelrod, Robert (1984), *The evolution of cooperation*. New York: Basic Books.

Axelrod, Robert and Leigh Tesfatsion (2006), "A guide for newcomers to agent-based modeling in the social sciences," in *Handbook of computational economics: Agent-based computational economics*, Leigh Tesfatsion and Kenneth L. Judd, Eds. Vol. 2. Amsterdam, The Netherlands: North Holland.

Axtell, Robert (2000), "Why agents? On the varied motivations for agent computing in the social sciences," Center on Social and Economic Dynamics.

Granovetter, Mark S. (1985), "Economic action and social structure: The problem of embeddedness," *American Journal of Sociology*, 91 (3), 481-510.

Heide, Jan B. and Kenneth H. Wathne (2006), "Friends, businesspeople, and relationship roles: A conceptual framework and a research agenda," *Journal of Marketing*, 70 (3), 90-103.

March, James G. (1994), *A primer on decision making*. New York: Free Press.

Montgomery, James D. (1992), "Job search and network composition: Implications of the strength-of-weak-ties hypothesis," *American Sociological Review*, 57 (5), 586-96.

---- (1998), "Toward a role-theoretic conception of embeddedness," *American Journal of Sociology*, 104 (1), 92-125.

North, Michael J., Nicholson T. Collier, and Jerry R. Vos (2006), "Experiences creating three implementations of the REPAST agent modeling toolkit," *ACM Transactions on Modeling and Computer Simulation*, 16 (1), 1-25.

Uzzi, Brian (1997), "Social structure and competition in interfirm networks: The paradox of embeddedness," *Administrative Science Quarterly*, 42 (1), 35-67.

---- (1996), "The sources and consequences of embeddedness for the economic performance of organizations: The network effect," *American Sociological Review*, 61 (4), 674-98.

Wathne, Kenneth H., Harald Biong, and Jan B. Heide (2001), "Choice of supplier in embedded markets: Relationship and marketing program effects," *Journal of Marketing*, 65 (2), 54-66.

THE DYNAMIC ENDOGENOUS EVOLUTION OF VOTER PREFERENCES

Alberto Alexander Perez*, Trinity College Dublin. Dublin, Ireland.

ABSTRACT

Why voters vote the way they do and how their preferences are formed are themes that have inspired much interdisciplinary research. A point in particular that has stirred debate is the question of the evolution of voter preferences through time. Are voters' preferences fixed or do they evolve? And if they evolve through time, do they evolve significantly to affect elections? Is this evolution and these changes in voters' preferences generated by exogenously determined shocks or can we explain this evolution endogenously?
A model that introduces endogenous dynamic evolution of voters' preferences is presented as well as a computer simulation written in Netlogo. Together they serve as tools to explore the plausibility of a dynamic endogenous evolution of voter preferences within the context of a spatial model of party competition.

Keywords: Computational Social Science Voter Preferences Netlogo Electoral Behavior

INTRODUCTION

As with any other type of model and simulation in the social sciences, election prediction is difficult. The deterministic nature of laws and environment rules in the natural sciences does not quite exist, at least in the same easily reproducible manner that facilitates experimentation in the social sciences and so although we can build extremely sophisticated models and simulations we can only approximate our models to a degree to reality. In the end it is only the election result that provides us with a way to calibrate our models and since in many ways each election is unique, the predictive capabilities of any model and simulation are confirmed only post facto.

Should we throw up our hands at an exercise that seems so doomed to always lack in its certainty of results? If our interest is the accuracy of election prediction only, it seems indeed that our destiny is similar to that of Tantalus, with the objective ever so close to us but condemned to ever fall short of our goal, however if we wish to study theoretical possibilities an study trtends, the story is different.

ELECTORAL BEHAVIOR APPROACHES AND ASSUMPTIONS

Many models, reflecting each a particular set of assumptions and a particular understanding of the world, have been created not necessarily just to predict but also to understand voter and party behavior. Models that belong to the rational choice camp have been very prolific models producing data, predictions and establishing relationships among variables that affect election outcomes (Page, Benjamin I., and Robert Y. Shapiro. 1992). However when it

* Corresponding author address: Alberto Alexander Perez, Political Science. Trinity College Dublin, Dublin 2, Ireland. e-mail: aperez@tcd.ie

comes down to prediction, the results have been unclear, yet their explanatory power is a pillar of the elections and voter behavior field (Niemi, Richard G., and Herbert F. Weisberg. 1992). There are other models with different assumptions that deal with the more evident shortcomings presented by the rational choice camp. Some of the more widely disseminated and influential models are those of the Michigan school or voter identification theories of party competition (University of Michigan Survey Research Center and Angus Campbell.1960). Voter identification models are helpful in understanding relationships between voter choice and party behavior, election results as well as with the inner social relationship in the electoral environment.

Anthony Downs stated in his seminal work; *An economic theory of democracy* (Downs, Anthony. 1957), one of the assumptions that has raised discussion as whether being appropriate is the assumption of endogenous preferences. This is the assumption that states that voter preferences are fixed through time, only to be affected by external shocks that affect the system in such a way that the voter preference distribution changes and so parties need to adjust their stated policies in reaction to these shocks if they wish to capture enough votes to be elected. Although preference change is recognized by Downs, stating that in the long run voter preferences do change, he believes that in the short run voter preferences are more likely to be fixed, and assuming so simplifies his model.

Endogenous voter preferences

The central idea in the discussion of endogenous voters preferences is that voter preferences are not fixed over time, they do change influenced by the environment and by voters around them. The reasoning is fairly straightforward; it is impossible for voters to have a ready-made opinion about all issues at the same time (Zaller, John. 1992). They do not have the time or perhaps even the interest to inform themselves about all points possible within a policy space. And so, even if they had an opinion, often they do not know how such a policy would affect them directly (Saris, Willem E., and Paul M. Sniderman. 2004).

Faced with many of this policies, voters look around them to people they trust for indications and pointers on many policies, either establishing, re-affirming or changing preferences and opinions particular issues convinced that the position of those believed to know better, or at least have a better understanding of the information available, would provide them with a higher level of utility than the position they had or would have taken by themselves alone (Summers, David A. 1968).

Preferences, even if we believe that they are fixed, can change if we are presented good reasons to do so, as the environment and the conditions under which they were formed in the first place change. This is different from assuming that there is an external shock by stating that these environmental changes are normal and indigenous of the system (Sniderman, Paul M., Richard A. Brody, and Philip E. Tetlock. 1991). If preferences were assumed to be fixed, campaigning would make little sense, as we would assume that voters could not be convinced that the policies espoused by each candidate are the best. Opinion polls would be unnecessary as people would maintain their policy positions and would not change their minds. Grass roots organizations would have no effect in the dissemination of policy, and the concept of convincing the voter about the usefulness of certain policies would be completely irrelevant.

The dynamic endogenous evolution of voter preferences model

The model presented here and the simulation built around it addresses the endogenous evolution of voter preferences and it incorporates a dynamic analysis of the system. Both parties and voters make the system evolve continuously by adapting their positions and making the system dynamic, evolving across the policy space through time.

The model presented here has the following assumptions:

- The model consist of voters and parties only
- The parties and voters move in a two-dimension policy space.
- Parties try to gather the maximum number of votes by positioning themselves as close as possible to a maximum number of voters
- One voter one vote.
- Parties are free to move along the policy space as they see fit. There is no restriction on positional movement.
- Parties choose strategies on how to attract voters and they act accordingly to stated rules of each strategy.
- Voters vote for the party closest to them, regardless of the direction in which this party lies around them.
- Voters can modify a particular voter's preferences, i.e. voters may move among the policy space, attracting other voters.
- Voters do not move randomly across the policy space. The rules for their movement are based on explicit conditions assumed at the time of building the voting environment, and they are well specified.

Parties do not directly modify voter's movement. They only affect affiliation. However, if a rule of attraction has been assumed, and a voter can attract another after switching to a new party, by that voter's decision, a third voter may be instilled to move by this switch.

With this model a simulation of the policy space has been created for the analysis of the concepts that we have discussed. The first part of the model, the party movement side, incorporates a dynamic agent-based version of the general spatial model of party competition Created by Michael Laver (Laver, 2005). This model provides the foundation of the dynamics in the model and introduces the idea and concept of party strategies, allowing for an adaptive dynamic behavior of parties in the system. The second part if the system incorporates the concepts of voter's endogenous evolutionary preference change. It allows a set of conditions under which voter's preferences will be affected and evolves through time adapting to the conditions of the system.

The dynamic endogenous evolution model of voter preferences (or DEE from here on) is then constituted of these two parts incorporated into one simulation put together and written in Netlogo 3.1.4. For the specifics of Michael Lavers' model and simulation, please refer to Laver, Michael. 2005. For the purposes of this paper a description of the environment and the simulation incorporated is here included

THE SIMULATION

The simulation is composed of two parts, the party movement and strategy part and the voter behavior part. The party behavior side assumes that parties can choose strategies according to their gain or loss of support or how ideologically tied they are with a particular policy. Supporters, who are agents in the model, use a very simple decision rule: they affiliate to the closest party. Party leaders use one of a number of decision rules that can be chosen by the operator in the interface for each party included in the system. In the STICK rule the party leader never changes the policy position of the party, regardless of the affiliations of supporters. This models a very ideological party that cares only about its policy position.

In the AGGREGATE rule the party leader sets party policy at the mean position on each dimension of the policies of the affiliated supporters.
In the i-HUNT rule (Insatiable Hunter) the party leader hunts for support using a simple "win-stay, lose-shift" Pavlovian hunting algorithm. In the s-HUNT rule (Satiable Hunter) if the previous move neither increased nor decreased support, stand still. The PREDATOR rule specifies that the party observes the current size of all parties and move towards the position of the largest party.

Besides being able to choose these rules the model includes options to choose number of parties, number of voters and the party switching sensibility in the part of voters. A graph and monitors show us the evolution in the size of parties, and we have a button that allow us to “jitter” supporters, i.e. to introduce a random walk in voters to analyze dynamics after a steady state or to introduce a shock if we think this is appropriate. All these elements come from Laver, 2005.

The voter side is composed of a chooser switch that allows us to set up the voter environment. The setting named DOWNS makes voters behave as in a rational choice model. They are fixed. This mode is included for comparison purposes. The setting NEIGHBOR instructs voters to look at their immediate surroundings and identify the closest ideological neighbor and take a step towards them. This is to simulate an environment where voters are acutely aware of the ideological positions of everybody and have the capacity to identify who is the closest person in their immediate ideological neighborhood. The setting RANDOM ANY allows the voters to pick a supporter at random, this is to simulate an environment where there is very little information about who surround us in the preference space and we are attracted to those who have other properties that affect our decision such as physical closeness, family friendship etc. The setting RANDOM DIFFERENT is the same as the previous one, but it exclusively allows for voters to be attracted to voters affiliated to other parties. This is done to try to catch the effects of non-policy related party dissatisfaction.

The setting CLOSEST ANY simulates an environment in which the voter has a perfect knowledge of the preference space and is attracted to the closest, ideologically, voter to him. This is different to neighbor since in this case, if dispersion of voter preferences is very big, the voter still would be able to identify and be attracted by another voter, and not only by those that are immediately in their ideological “neighborhood”. The setting CLOSEST DIFFERENT is similar to the previous one, but the voter is only attracted to voters that do not share his same party affiliation. Again this is to try to include effects that are non-policy related in his attachment for a party, such a personal dislike of the leadership of a party (Figure 1).

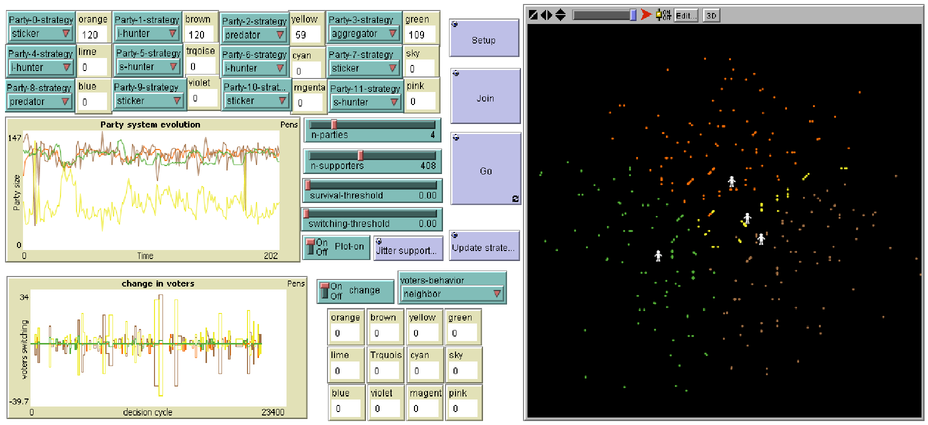


FIGURE 1 Model's interface and settings. In this case, showing dynamics of party and voters with their interactions displayed as voters lost or gained and party size.

The interaction of the two parts makes the DEE model complete. Voters react to other voter's behavior and parties to voter movement. Indirectly parties affect the movement of voters as well, in particular in the RANDOM DIFFERENT and CLOSEST DIFFERENT settings. The main interactions can be tracked in the interface where there are two plots and several reporters. The first graph keeps track of the party evolution, i.e. how parties grow or become smaller through the system. The second Graph shows the net change in voters, represented by a negative number if the party lost voters or a positive number if the party gained voters.

We will see what is the behavior of the model in the following section.

Model Evolution

The model can describe real life situations. In particular it has provided scenarios that seem to suggest that bandwagon-effects, underdog effects, sandbagging effects and the growth of extreme parties is an emergent property of a dynamic evolutionary system. We can say that experiments performed with the model indicate endogenous movement that results in these phenomena. the analysis of the dynamics seem to be scale and time dependent (Figure 2).

The key to implement the model is to accurately observe the desired party and voter environment that we want to reproduce, if that is our objective, although theoretical manipulation of the potential possibilities seems as rich if not more fertile ground for research

From preliminary results, evidently a combination of party dynamics with all using i-HUNT strategies or and CLOSEST-ANY produce the more interesting dynamics. The system is in constant movement and the results are surprising. On the other hand, the combination STICK and DOWNS give no movement to the system whatsoever.

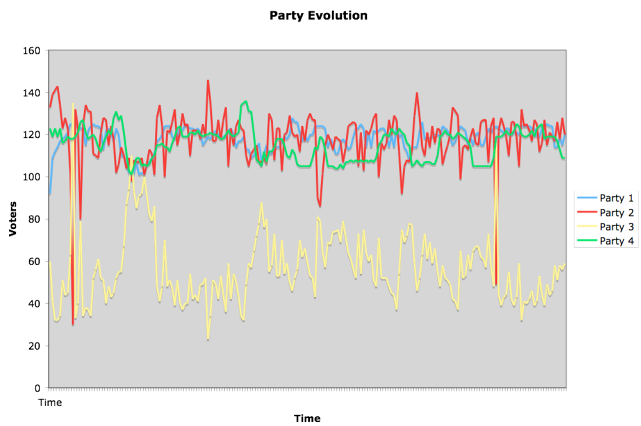


Figure 2. The stability of the systems, it seems, depends of the magnitude of the time interval used to analyze the system, which poses interesting questions on current theory.

From the combinations in between we can say that the if we set all parties at STICK we can then observe voter only behavior in the system, and vice-versa, if we set the voter environment to Downs, it reverts to the original Laver 2005 model. If we want to emulate a purely Downsian rational choice model, the settings would be PREDATOR for all parties and DOWNS for voters.

Other results that are interesting are those obtained with any party setting and a voter setting of RANDOM-ANY or RANDOM-DIFFERENT. In the first instance the system seems to collapse to the center, but since these are ideal points, even in the center there is movement and evolution. Hence the result of a party gaining all voters and collapsing into a single spot is not a trivial one as it is modified by the evolution of the system.

The settings that provided with interesting results to study the behavior of voters at the borderlines of different parties are those of NEIGHBORS and CLOSEST DIFFERENT. Most of the evolution and adaptation in these cases is strictly at the edges of all political parties regardless of their strategies, and has prompted us to consider a study of undecided voters based in an agent based model in the future.

CONCLUSIONS

The development of a model that incorporated endogenous evolutionary voter's preferences proved not only to be plausible but also theoretically and experimentally rich. The evolution of the systems described not only describe familiar systems but also propose the existence of different plausible scenarios in real life and may help to achieve a theoretical understanding of such cases. The model has proven reliable in the sense that by using the appropriate settings we can simulate standard models of electoral competition, yet it is flexible enough to allow us to modify and adapt it to systemic conditions that we may want to emulate. Just as with any other model, the transposition and usefulness to test or understand real world data, if we wished to use it for this purpose, completely relies in the understanding of the system being emulated and the appropriate conditions that need to be used to choose the settings. An immediate objective is to use this model to simulate the French, Belgian and Dutch elections to try to understand their dynamics in regards to the evolution of radical right parties in these countries, and objective that gave rise to this approach. An interesting approach would be to test this model and simulation in a two-step process, as in the primaries in the United States with many candidates, and presidential elections, as second round elections, with only two, A challenging prospect but a very rich one.

REFERENCES

Anderson, Christopher J., and Yuliya V. Tverdova. 2001. Winners, Losers, and Attitudes about Government in Contemporary Democracies. International Political Science Review / Revue internationale de science politique 22 (4):321-338.

Axelrod, Robert. 1997. The complexity of cooperation : agent-based models of competition and collaboration. Princeton, N.J. ; Chichester: Princeton University Press.

Axelrod, Robert, and Richard Dawkins. 1984. The evolution of cooperation. London: Penguin.

Beasley, Ryan K., and Mark R. Joslyn. 2001. Cognitive Dissonance and Post-Decision Attitude Change in Six Presidential Elections. Political Psychology 22 (3):521-540.

Campbell, Angus. 1964. The American voter : an abridgment: Wiley.

Carter, Richard F. 1959. Bandwagon and Sandbagging Effects: Some Measures of Dissonance Reduction. The Public Opinion Quarterly 23 (2):279-287.

Clubb, Jerome M., William H. Flanigan, and Nancy H. Zingale. 1981. Analyzing electoral history : a guide to the study of American voter behavior. Beverly Hills ; London: Sage.

Downs, Anthony. 1957. An economic theory of democracy. New York: Harper & Row.

Gallagher, Michael, Michael Laver, and Peter Mair. 2005. Representative government in modern Europe. 4th ed. London: McGraw-Hill.

Ginsberg, Benjamin, and Robert Weissberg. 1978. Elections and the Mobilization of Popular Support. American Journal of Political Science 22 (1):31-55.

Granberg, Donald, and Tim Nanneman. 1986. Attitude Change in an Electoral Context as a Function of Expectations Not Being Fulfilled. Political Psychology 7 (4):753-765.

Hill, David Lee. 2006. American voter turnout : an institutional perspective. Boulder, Colo. ; Oxford: Westview.

Kim, Chong Lim, and Donald P. Racheter. 1973. Candidates' Perception of Voter Competence: A Comparison of Winning and Losing Candidates. The American Political Science Review 67 (3):906-913.

Kirkpatrick, Samuel A. 1970. Political Attitude Structure and Component Change. The Public Opinion Quarterly 34 (3):403-407.

Laver, Michael. 1981. The politics of private desires. Harmondsworth: Penguin.

Laver, Michael. 2000. Government Formation and Public Policy. PS: Political Science and Politics 33 (1):21-23.

Laver, Michael. 2001. Estimating the policy position of political actors. London: Routledge.

Laver, Michael, and Kenneth Benoit. 2003. The Evolution of Party Systems between Elections. American Journal of Political Science 47 (2):215-233.

Laver, Michael. 2005. Policy and the dynamics of political competition. *American Political Science Review* 99:2 (May) 263-281.

Laver, Michael, and W. Ben Hunt. 1992. Policy and party competition. New York: Routledge.

Laver, Michael, and Norman Schofield. 1990. Multiparty government : the politics of coalition in Europe, Comparative European politics. Oxford: Oxford University Press.

LeDuc, Lawrence, Richard G. Niemi, and Pippa Norris. 1996. Comparing democracies. London: SAGE.

LeDuc, Lawrence, Richard G. Niemi, and Pippa Norris. 2002. Comparing democracies 2 : new challenges in the study of elections and voting. [2nd ed. London: Sage.

Miller, Warren E., and J. Merrill Shanks. 1996. The new American voter. Cambridge, Mass. ; London: Harvard University Press.

Morwitz, Vicki G., and Carol Pluzinski. 1996. Do Polls Reflect Opinions or Do Opinions Reflect Polls? The Impact of Political Polling on Voters' Expectations, Preferences, and Behavior. The Journal of Consumer Research 23 (1):53-67.

Nie, Norman H., Sidney Verba, and John R. Petrocik. 1979. The changing American voter. Enlarged ed. Cambridge, Mass. ; London: Harvard University Press.

Niemi, Richard G., and Herbert F. Weisberg. 1992. Controversies in voting behavior. 3rd ed. Washington, D.C.: CQ Press.

Niemi, Richard G., and Herbert F. Weisberg. 1993. Classics in voting behavior. Washington, D.C.: CQ Press.

Page, Benjamin I., and Robert Y. Shapiro. 1992. The rational public : fifty years of trends in Americans' policy preferences. Chicago: University of Chicago Press.

Regan, Dennis T., and Martin Kilduff. 1988. Optimism about Elections: Dissonance Reduction at the Ballot Box. Political Psychology 9 (1):101-107.

Saris, Willem E., and Paul M. Sniderman. 2004. Studies in public opinion : attitudes, nonattitudes, measurement error, and change. Princeton, N.J. ; Oxford: Princeton University Press.

Sniderman, Paul M. 1975. Personality and democratic politics. Berkeley [etc.] ; London: University of California Press.

Sniderman, Paul M., Richard A. Brody, and Philip E. Tetlock. 1991. Reasoning and choice : explorations in political psychology: Cambridge University Press.

Summers, David A. 1968. Conflict, Compromise, and Belief Change in a Decision-Making Task. Vol. 12, The Journal of Conflict Resolution: Sage Publications, Inc.

Teixeira, Ruy A. 1992. The disappearing American voter. Washington, D.C.

University of Michigan. Survey Research Center., and Angus Campbell. 1960. The American voter. New York: Wiley.

Zaller, John. 1992. The nature and origins of mass opinion. Cambridge: Cambridge University Press.

SPONTANEOUS COORDINATION

D. DIERMEIER,* Northwestern University, Evanston, IL
C. ANDONIE, Northwestern University, Evanston, IL

ABSTRACT

In this paper we study large potential games with global random matching. The model defines a discrete stationary Markov process, in discrete time. We derive the limiting distribution for the paradigmatic case of coordination 2x2 games.We show that spontaneous coordination is possible for substantial noise levels. Whether coordination occurs depends on the product of the incentives (benefit of coordinating, J) and an incentives responsiveness parameter that captures noise in the system (β). If the product is below a critical level, players fail to coordinate; if it is above, the distribution will be bi-modal with the highest peak close to the risk-dominant strategy.

Keywords: Markov process, Limiting distribution, Adaptive model, Coordination, Noisy best response

INTRODUCTION

Theoretical economics has recently begun to explore the implications of models of bounded rationality and learning.[2] The basic idea is as old as Nash equilibrium. In his unpublished Ph.D. dissertation, John Nash wrote the following:[3]

> It is unnecessary to assume that the participants have full knowledge of the total structure of the game, or the ability and inclination to go through any complex reasoning processes. But the participants are supposed to accumulate empirical information on the relative advantages of the various pure strategies at their disposal.
> To be more detailed, we assume that there is a population (in the sense of statistics) of participants for each position of the game. Let us also assume that the 'average playing' of the game involves n participants selected at random from the n populations, and that there is a stable average frequency with which each pure strategy is employed by the 'average member' of the appropriate population (...) Thus the assumption we made in this 'mass-action' interpretation leads to the conclusion that the mixed strategies representing the average behavior in each of the populations form an equilibrium point.

* *Corresponding author address:* Daniel Diermeier, MEDS Department, Kellogg School of Management, and Northwestern Institute on Complex Systems (NICO), Northwestern University, 2001 Sheridan Road, Evanston, IL 60208; e-mail: d-diermeier@kellogg.northwestern.edu

[2]For overviews see e.g. Fudenberg and Levine (1998), Young (1998), Blume (1997).

[3]Nash, pp. 21-23, quoted in Weibull (1997).

Most of the work, however, has focused on providing behavioral foundations for existing solution concepts, most importantly Nash-Equilibrium and its refinements (Foster and Young (1990), Blume (1993), Blume (1995), Kandori, Mailath, and Rob (1993), Young (1993)). The general strategy of this approach can be described as follows. For a given normal form game, the researcher defines some version of perturbed best response that induces a regular Markov process. Since regular Markov processes yield unique stationary distributions, it is then shown that for vanishing noise in the limit, the unique stationary distribution puts positive probability only on a subset of possible action configurations, the so-called "stochastically stable states" (Foster and Young (1990)). This approach thus suggests answers to two important questions in classical game theory: the problem of coordination and the problem of selection. Ordered states emerge without a central coordinating device and are frequently unique even in the case of many strict Nash equilibria.

In this approach the use of behavioral models is largely foundational. Consequently, issues of robustness, e.g. with respect to distributional assumptions and so forth, are of central concern (Foster and Young (1990), Bergin and Lipman (1996)). In this paper we take a different approach. Rather than using the behavioral models in a foundational sense we use them in an explanatory way. In other words, we will directly explore the properties of limiting distributions rather than just exploring the limits as the noise term vanishes. The goal is to explore the properties of a certain behavioral approach directly and assess whether it can provide additional insights.

The basic model can be described as follows. In each period one agent is randomly selected to change his behavior.[4] That agent then will receive (possibly partial) information about the current state of play, i.e. the vector of actions currently chosen by each agent. Based on this information the agent responds by choosing an action, according to some (noisy) best-response rule. That is, with some probability the agent chooses a best response against the current configuration of play, but with some probability he chooses some other action. The realization of that action then determines the next period's configuration of play; again an agent is chosen (with replacement) and so forth. The key idea of our model is to "decompose" the simultaneous choice of classical game-theory (where agents form conjectures about each others beliefs) into a dynamic adjustment process.

As in classical game-theory, some features of the approach model are mainly technical, while others are of substantive importance. One of the technical assumptions pertains to selecting exactly one agent in each period. This does not imply that agents cannot change their behavior "quickly". After all, periods between revisions can be arbitrarily small. The *informational* implication of this assumption, however, is substantial. That is, when revising their actions, agents base their actions on their information about the current state of the dynamic system. This information may be complete (i.e. agents observe the complete current state of play), partial (i.e. they may only receive information about some agents, i.e. their "neighbors" on some network structure), or noisy (i.e. information may come in the form of polls). How this informational structure is modeled may have important consequences on the behavior of the system.

Among the substantive assumptions, perhaps the most important pertains to bounded

[4]We focus here on a discrete time process. Equivalently, we could consider continuous time formulations where the time between revisions is exponentially distributed.

rationality. First, agents are assumed to be myopic. They optimize conditional on the current behavior in the population without anticipating the future strategic consequences of their actions. Agents are not assumed to believe that other actors reason in the same way as they do, or even that they have the same payoff function. Indeed, they do not expect that their action may influence the future decisions of other participants. Agents simply base their choice on what action maximizes their current payoff.

Second, agents are assumed to respond to incentives, but not perfectly. That is, agents behavior is characterized by noisy decision making. While the main motivation for this assumption is technical[5], it has a plausible foundation in the random utility model due to McFadden (1973).[6] This approach seems especially appropriate in models of decision situations where the perceived costs and benefits may vary over time. Finally, we are mainly interested in applications to mass behavior such as collective action problems or conventions. We will therfore focus on large-N approximations of the induced Markov process.

In this paper we derive the closed form solution for the paradigmatic case of coordination 2×2 games[7]. We also show that important phenomena are missed if attention is exclusively focused on the case of vanishing noise. First, spontaneous coordination is possible for substantial noise levels. Second, whether coordination occurs depends on the product of the incentives generated by the payoff matrix and an incentive responsiveness parameter that captures the noise in the system. If the product is below a critical level, players will not coordinate. Rather, the limiting distribution will be unimodal. If this product is above a critical level, the stationary distribution will be bimodal with both peaks close to the two Nash-equilibria and the higher peak associated with the risk-dominant equilibrium. Thus, everything else equal, higher payoff differences would result in higher expected levels of coordination.

GENERAL MODEL

We consider a large population of players $\Im$ (with $|\Im| = KN$) that is partitioned into K types. Players interact according to a K-type strategic form game. Players interact repeatedly with each other according to a model of adaptive play with persistent randomness. At each point in time each player's decision gives rise to the population's configuration of play. The set of these configurations C then corresponds to a state space of a stochastic process induced by individual adaptive behavior.

Let S_k be the (finite) number of pure strategies $s_l^k (l = 1, ..., S_k)$ for a player of type k. Further let $n(s_l^k)$ be the number of players of type k playing strategy s_l^k. Then

$$\mathbf{n} := (n(s_1^1), ..., n(s_{S_1}^1), ..., n(s_l^k), ..., n(s_1^K), ..., n(s_{S_K}^K))$$

[5]Together with random selection it ensures that the stochastic process is ergodic.

[6]Alternatively, agents may be assumed to make a "mistake" with a fixed probability. See Kandori, Mailath and Rob (1993) or Young (1993) for details.

[7]The game is widely used in political science applications, for example in the study of collective action (e.g. Chong (1991)).

is a generic configuration of the set C. We allow only for elementary adjustments. That is, at most one individual of type k can change her behavior at a given time from s_l^k to $s_{l'}^k$. Thus from a given state n only states of the form

$$\mathbf{n}' := (n(s_1^1), ..., n(s_{S_1}^1), ..., n(s_l^k) - 1, ..., n(s_{l'}^k) + 1, ...n(s_1^K), ..., n(s_{S_K}^K))$$

are accessible.

Transition probabilities $P(\mathbf{nn}')$ are defined by selection and action probabilities. Selection probabilities are defined as follows: in each period t one specific agent out of KN is randomly chosen with probability $1/(KN)$.[8] The agent then looks at the current configuration $\mathbf{n}$ of actions in the population and adjusts his action according to a given behavioral rule. During the next period, again a player (perhaps the same) is chosen at random, and so on. Given the current configuration, an actor will then probabilistically adjust his participation behavior to improve his payoff.

Let $p^\beta(s_l^k|\mathbf{n})$ denote the conditional probability that in the next period an agent will play strategy s_l^k given that the current configuration of play is $\mathbf{n}$. Specifically, we assume log-logistic response rules for all agents:

$$p^\beta(s_l^k|\mathbf{n}) = \frac{\exp[\beta u(s_l^k; \mathbf{n})]}{\sum\limits_{s_{l'}^k} \exp[\beta u(s_{l'}^k; \mathbf{n})]}.$$

This captures the assumption that the pair-wise probability ratios of choosing actions are proportional to the respective pay-off differences. The log-linear choice model is closely connected to the best-response correspondence. The parameter β formally captures the degree to which the deterministic component of utility (given by the payoff matrix) determines choice. A low β corresponds to the case where a decision is not much influenced by the incentives specified in the model. For $\beta = 0$ choice is completely random. That is, for all possible configurations, the agent will play each action with equal probability. For $\beta \to \infty$, log-linear choice converges to a distribution that puts positive probability only on best-responses to $\mathbf{n}$.

Action and selection probabilities define a (regular) Markov chain with a unique limiting distribution π that satisfies the global balance conditions $\pi = \pi P$. In general, global balance equations cannot be solved in closed form. However, for the case where a k-player game has a potential $F(\mathbf{n})$ (Monderer and Shapley (1996)), Blume (1997) has shown that the limiting distributution satisfies a stronger property, the *detailed balance condition.*

$$P(\mathbf{nn}')\pi(\mathbf{n}) = P(\mathbf{n}'\mathbf{n})\pi(\mathbf{n}')$$

We then immediately have the following result:

Theorem 1 *Let* $\{\mathbf{n}[1], ..., \mathbf{n}[j\}, ...\mathbf{n}[m]\}$ *be a sequence of configurations of length* m. *For log-logistic adjustment rule in games with a potential stationary distributions are of the form*

$$\pi(\mathbf{n}[m]) = \prod_{j=1}^{m-1} \frac{P(\mathbf{n}[j], \mathbf{n}[j+1])}{P(\mathbf{n}[j+1], \mathbf{n}[j])} \pi(\mathbf{n}[1]).$$

[8] For simplicity, we assume that revisions are made each period. All results, however, continue to hold in continuous time when the time between revisions is exponentially distributed.

2x2 COORDINATION GAMES WITH RANDOM GLOBAL MATCHING

We now apply this approach to the paradigmatic case of coordination problems: 2×2 coordination game with random global matching. So, consider a symmetric 2×2 games with payoff matrix

$$M = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{pmatrix}.$$

We denote the top/left action by s_1, the bottom/right action by s_2. Then, for random global matching, a player i's total payoff from a configuration $\mathbf{n}$ is given as

$$u_i(\mathbf{n}) := \sum_{i' \in \Im \setminus \{i\}} \frac{1}{2N-1} u_{ii'}$$

It can be easily seen that the following matrix F constitutes a potential function for the game M.

$$F = \begin{pmatrix} \frac{3u_{11}+u_{12}-3u_{21}-u_{22}}{4} & \frac{-u_{11}+u_{12}+u_{21}-u_{22}}{4} \\ \frac{-u_{11}+u_{12}+u_{21}-u_{22}}{4} & \frac{-u_{11}-3u_{12}+u_{21}+3u_{22}}{4} \end{pmatrix}$$

By defining

$$J = \frac{u_{11} - u_{12} - u_{21} + u_{22}}{2} \text{ and } h = \frac{u_{11} + u_{12} - u_{21} - u_{22}}{4}$$

we can rewrite F as[9]

$$F = \begin{pmatrix} \frac{J}{2} + 2h & -\frac{J}{2} \\ -\frac{J}{2} & \frac{J}{2} - 2h \end{pmatrix}$$

Further, if $F(\cdot)$ exists and is symmetric for a 2×2 game we can easily define a potential function $F(\cdot)$ for any finite network game (Blume (1997)). Thus, we can apply Theorem 1 to derive the limiting distribution.

Since, $\mathbf{n} = (n(s_1), n(s_2))$, each configuration is uniquely determined by $\frac{n(s_1)-n(s_2)}{2}$ which we abbreviate by n. By a slight abuse of notation, we also refer to each state by n, and write $n+1$ for $\mathbf{n}' = (n(s_1)+1, n(s_2)-1)$ and $n-1$ if $\mathbf{n}' = (n(s_1)-1, n(s_2)+1)$. Let $p_s(n)$ denote the (selection) probability of chosing a player using action $s = s_1, s_2$ in state n. Then

$$p_{s_1}(n) = \frac{n(s_1)}{2N} = \frac{N+n}{2N} \qquad \text{and} \qquad p_{s_2}(n) = \frac{n(s_2)}{2N} = \frac{2N - n(s_1)}{2N} = \frac{N-n}{2N}$$

[9] In coordination games, i.e. $u_{11} > u_{21}, u_{22} > u_{12}, u_{11} \geq u_{22}$, these parameters have a natural interpretation. The parameter $+J$ measures the benefit from coordinating. The parameter h indicates which of the two equilibria is risk-dominant. Since in coordination games we have $J > 0$, the argmax set of $F(\cdot)$ depends on h. If $h > 0$, then (s_1, s_1) is risk-dominant. If $h < 0$, (s_2, s_2) is risk-dominant

We will now apply Theorem 1 to explicitly calculate $\pi(n)$.[10]. First, using $F(.)$ we have

$$
\begin{aligned}
\frac{\pi_n}{\pi_{n-1}} &= \frac{P(n-1,n)}{P(n,n-1)} = \\
&= \frac{p_{s_2}(n-1)}{p_{s_1}(n)} \frac{\exp[\beta(\frac{n(s_1)-1}{2N-1}u_{11} + \frac{2N-1-(n(s_1)-1)}{2N-1}u_{12})]}{\exp[\beta(\frac{n(s_1)-1}{2N-1}u_{21} + \frac{2N-1-(n(s_1)-1)}{2N-1}u_{22})]} \\
&= \frac{p_{s_2}(n-1)}{p_{s_1}(n)} \frac{\exp[\beta(\frac{n(s_1)-1}{2N-1}(\frac{J}{2}+2h) + \frac{n(s_2)}{2N-1}(-\frac{J}{2}))]}{\exp[\beta(\frac{n(s_1)-1}{2N-1}(-\frac{J}{2}) + \frac{n(s_2)}{2N-1}(\frac{J}{2}-2h))]} \\
&= \frac{n(s_2)+1}{n(s_1)} \frac{\exp[\beta(\frac{n(s_1)-1}{2N-1}2h + \frac{n(s_1)-n(s_2)-1}{2N-1}\frac{J}{2})]}{\exp[\beta(\frac{n(s_1)-n(s_2)-1}{2N-1}(-\frac{J}{2}) - \frac{n(s_2)}{2N-1}(2h))]} \\
&= \frac{N-n+1}{N+n} \exp[\beta(\frac{n(s_1)-n(s_2)-1}{2N-1}J + 2h)] \\
&= \frac{N-n+1}{N+n} \exp[\beta(\frac{2n-1}{2N-1}J + 2h)] \\
&= \frac{N-n+1}{N+n} \exp[\beta(\frac{n}{2N-1}2J - \frac{1}{2N-1}J + 2h)] \\
&= \frac{N-n+1}{N+n} \exp[\beta(2n\tilde{J} - \tilde{J} + 2h)]
\end{aligned}
$$

where $\tilde{J} = \frac{1}{2N-1}J$. Now, using Theorem 1, we can derive π_n for each n with $1 \leq n \leq N$.[11]

$$
\begin{aligned}
\pi_n &= \pi_0 \prod_{i=1}^{n} \frac{N-i+1}{N+i} \exp[\beta(2i\tilde{J} - \tilde{J} + 2h)] = \\
&= \pi_0 \frac{N\cdot(N-1)\cdots(N-(n-1))}{(N+1)\cdot(N+2)\cdots(N+n)} \exp[\beta(\sum_{i=1}^{n} 2i\tilde{J} - n\tilde{J} + 2hn)] \\
&= \pi_0 \frac{N!/(N-n)!}{(N+n)!/N!} \exp[\beta((\frac{1}{2}n^2 + \frac{1}{2}n)2\tilde{J} - n\tilde{J} + 2hn)] \\
&= \pi_0 \exp[\beta(n^2\tilde{J} + 2hn) + \ln(\frac{(N!)^2}{(N+n)!(N-n)!})]
\end{aligned}
$$

We now have the following result.

Proposition 1 *For N sufficiently large, the most likely state is at n^*, with:*
- $n^* > 0$ if $h > 0$
- $n^* < 0$ if $h < 0$

Proof.

Consider the function $f : [-N+1, N] \to R$ given by

$$f(x) = \frac{N-x+1}{N+x} \exp\left[\beta\left(\frac{2x-1}{2N-1}J + 2h\right)\right]$$

---

[10] The case of uniform, global matching corresponds to a two-state model with a homogenous population which is well-known in the statistical mechanics literature (e.g. Weidlich (1991)).

[11] The case of $-N \leq n \leq -1$ is completely analogous and thus omittted.

Note that

$$\frac{df(x)}{dx} = \exp\left[\beta\left(\frac{2x-1}{2N-1}J+2h\right)\right] \frac{1}{(N+x)^2(2N-1)}\frac{1}{2\beta J}\left(-x^2+x+\left(1-\frac{2}{\beta J}\right)N^2+N+\frac{1}{2\beta J}\right)$$

To determine the sign of $\frac{df(x)}{dx}$, it is sufficient to determine the sign of

$$-x^2+x+\left(1-\frac{2}{\beta J}\right)N^2+N+\frac{1}{2\beta J}$$

which we denote by $h(x)$. Also, denote by $b := \frac{1}{2}\beta J$.

We consider 3 cases.

Case 1) $b<1$. Observe that h has a maximum at $x^* = \frac{1}{2}$, which corresponds to

$$h(\frac{1}{2}) = \frac{1}{4}+\left(1-\frac{2}{\beta J}\right)N^2+N+\frac{1}{2\beta J}$$

Since $1-\frac{2}{\beta J}<0$, it follows that, for large N:

$$h(x) \leq h\left(\frac{1}{2}\right) < 0$$

for all $x \in [-N+1, N]$. This implies that f decreases from

$$f(-N+1) = 2N\exp\left[\beta\left(-J+2h\right)\right] > 1$$

to

$$f(N) = \frac{1}{2N}\exp\left[\beta\left(J+2h\right)\right] < 1$$

We conclude that π_n has a unique maximum at n^*, with $f(n^*)=1$. Moreover, we have:

a) if $h>0$, then $n^*>0$, since $f(0) = \frac{N+1}{N}\exp\left[\beta\left(-\frac{1}{2N-1}J+2h\right)\right] > 1$ for large N.

b) if $h<0$, then $n^*<0$, since $f(0) = \frac{N+1}{N}\exp\left[\beta\left(-\frac{1}{2N-1}J+2h\right)\right] < 1$ for large N.

Case 2) $b=1$. In this case

$$h(x) = -x^2+x+N+\frac{1}{4}$$

The equation $h(x)=0$ has two real and distinct solutions, which we denote by x_1 and x_2, given by:

$$x_1 = \frac{1-\sqrt{\Delta}}{2} \text{ and } x_2 = \frac{1+\sqrt{\Delta}}{2}$$

where $\Delta = 1+4\left(N+\frac{1}{4}\right)$. Note that $-N+1<x_1<0<x_2<N$. We summarize the behavior of f in Table 1.

As $N\to\infty$, $f(x_1)\to\exp(\beta 2h)$ and $f(x_2)\to\exp(\beta 2h)$.

a) If $h>0$, then, for large N, both $f(x_1)>1$ and $f(x_2)>1$, which together with $f(-N+1)>1$ and $f(N)<1$ imply that π_n has a unique max at n^*, where $n^*>0$.

Table 1: Behavior of f in Case 2

x	$-N+1............x_1............x_2........N$
$f(x)$	$f(-N+1)\searrow f(x_1)\nearrow f(x_2)\searrow f(N)$

Table 2: Behavior of f in Case 3

x	$-N+1............x_1............x_2........N$
$f(x)$	$f(-N+1)\searrow f(x_1)\nearrow f(x_2)\searrow f(N)$

b) If $h<0$, then, for large N, both $f(x_1)<1$ and $f(x_2)<1$, which together with $f(-N+1)>1$ and $f(N)<1$ imply that π_n has a unique max at n^*, where $n^*<0$.

Case 3) $b>1$. The equation $h(x)=0$ has two real and distinct solutions, which we denote by x_1 and x_2 given by:

$$x_1=\frac{1-\sqrt{\Delta}}{2} \text{ and } x_2=\frac{1+\sqrt{\Delta}}{2}$$

where

$$\Delta=1+4\left[\left(1-\frac{2}{\beta J}\right)N^2+N+\frac{1}{2\beta J}\right]$$

We summarize the behavior of f in Table 2.

As $N\to\infty$,

$$f\left(x_1\right)\to\frac{1+\sqrt{1-\frac{2}{\beta J}}}{1-\sqrt{1-\frac{2}{\beta J}}}\exp\left[\beta\left(-\sqrt{1-\frac{2}{\beta J}}J+2h\right)\right]$$

and

$$f\left(x_2\right)\to\frac{1-\sqrt{1-\frac{2}{\beta J}}}{1+\sqrt{1-\frac{2}{\beta J}}}\exp\left[\beta\left(\sqrt{1-\frac{2}{\beta J}}J+2h\right)\right]$$

Let's denote by

$$a_0:=\frac{1}{2}\left[\beta J\sqrt{1-\frac{2}{\beta J}}-\log\frac{1+\sqrt{1-\frac{2}{\beta J}}}{1-\sqrt{1-\frac{2}{\beta J}}}\right]$$

and by

$$a:=\beta h$$

Observe that $f\left(x_1\right)<1$ is equivalent to $a<a_0$, and $f\left(x_2\right)<1$ is equivalent to $a<-a_0$. Depending on the values of a, we split our analysis into 3 subcases:

Subcase 3A). If $|a|<a_0$ then, for large N, $f\left(x_1\right)<1$ and $f\left(x_2\right)>1$, which together with $f(-N+1)>1$ and $f(N)<1$ imply that π_n has two maxima at n_1^* and n_2^*, with $n_1^*<x_1<0<x_2<n_2^*$, and one minimum at n_3^*, with $x_1<n_3^*<x_2$. To determine the global maximum, we consider 2 subcases:

1) $h>0$. Note that

$$f\left(0\right)=\frac{N+1}{N}\exp\left[\beta\left(\frac{-1}{2N-1}J+2h\right)\right]\to\exp\left(2\beta h\right)>1$$

Table 3: Extrema of π_n for different parameter values

	$b<1$	$b=1$	$b>1$
$\|a\|<a_o$	1 maximum	1 maximum	2 maxima, 1 minimum
$\|a\|=a_o$	1 maximum	1 maximum	2 maxima, 1 minimum
$\|a\|>a_o$	1 maximum	1 maximum	1 maximum

so the minimum is at $n_3^* < 0$. Using this fact, we can write:

$$\begin{aligned} \pi_{n_2^*} &\geq \pi_{-n_1^*} = \pi_0 \frac{(N!)^2}{(N+n_1^*)!\,(N-n_1^*)!} \exp\left[\beta\left((n_1^*)^2 \tilde{J} - 2hn_1^*\right)\right] \\ &= \pi_{n_1^*} \exp\left(-\beta 4hn_1^*\right) > \pi_{n_1^*} \end{aligned}$$

so the global max is at $n_2^* > 0$.

2) $h<0$. Note that

$$f(0) = \frac{N+1}{N} \exp\left[\beta\left(\frac{-1}{2N-1}J + 2h\right)\right] \to \exp(2\beta h) < 1$$

so the min is at $n_3^* > 0$. Using this fact, we can write:

$$\begin{aligned} \pi_{n_1^*} &\geq \pi_{-n_2^*} = \pi_0 \frac{(N!)^2}{(N+n_2^*)!\,(N-n_2^*)!} \exp\left[\beta\left((n_2^*)^2 \tilde{J} - 2hn_2^*\right)\right] \\ &= \pi_{n_2^*} \exp\left(-\beta 4hn_2^*\right) > \pi_{n_2^*} \end{aligned}$$

so the global max is at $n_1^* < 0$.

Subcase 3B) If $|a| > a_0$ then, for large N, $f(x_1) > 1$, $f(x_2) > 1$ or $f(x_1) < 1$, $f(x_2) < 1$. In both cases, π_n has a unique max at n^*, with $f(n^*) = 1$. Note that:

- if $a > a_0$, that is $h > 0$, the global maximum is at $n^* > 0$
- if $a < -a_0$, that is $h < 0$, the global maximum is at $n^* < 0$.

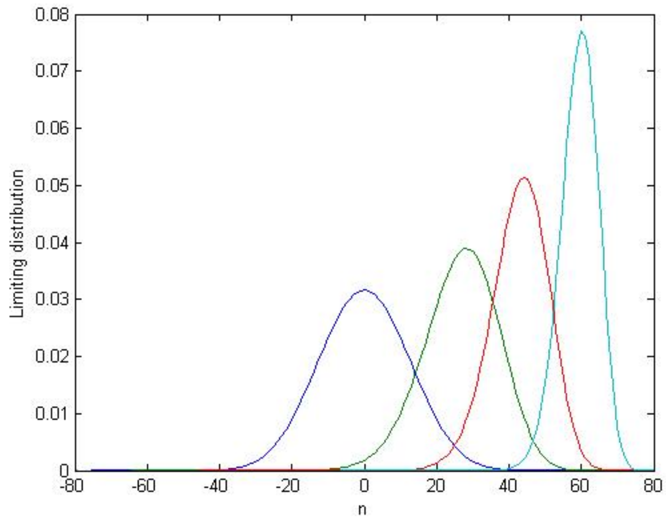
Subcase 3C) Finally $|a| = a_0$. First consider $a = a_0$. Then $f(x_1) \to 1$ and $f(_2) > 1$ (for large N). We can show that $f(x_1)$ is increasing in N, and since $f(x_1) \to 1$, we necessarily have $f(x_1) < 1$, for sufficiently large N. These observations, together with $f(-N+1) > 1$ and $f(N) < 1$, imply that π_n has two maxima and one minimum.
The second case $a = -a_0$ is similar i.e. π_n has two maxima and one minimum.
Moreover, we can prove the following (by analogy with subcase 3A):

- if $a = a_0$, that is $h > 0$, the global maximum is at $n^* > 0$
- if $a = -a_0$, that is $h < 0$, the global maximum is at $n^* < 0$. **QED**

We summarize the results in Table 3.

Of course, we can also derive the stochastically stable states for coordination games. Assume first that $h > 0$. As $\beta \to \infty$, we have $b \to \infty$ and $a < a_0$ (actually we can show that $(a - a_0) \to -\infty$) so we are in case 3A from above. But in this case we know that the global maximum is at n_2^*, with $x_2 < n_2^* < N$. Since, as $\beta \to \infty$, $f(x_2) \to \infty$ and $f(N) \to 1$ from below, it must be that $n_2^* \to N$. The case $h < 0$ is similar, i.e. the global maximum $n_1^* \to -N$. That is, in the limit, all players are expected to coordinate on the risk-dominant equilibrium.

Figure 1: As a increases, the mode shifts towards $N = 80$



However, by using Table 3 we can also identify the qualitative features of the limiting distribution for substantial noise. In the case of $b = \frac{1}{2}\beta J < 1$, the limiting distribution is unimodal and coordination on one of the pure Nash-equilibria does not occur. For $a = 0$ the mode is at state $n = 0$, but shifts towards $n = N$ for increasing a i.e. eveybody coordinating on s_1. This is illustrated in Figure 1. We note that for $a = 0$, the distribution has a peak at $n = 0$, but as we increase a (by increasing h) the peak shifts towards $n = 80$. Also we note that for high values of a, most of the distribution mass is concentrated around the mode. That is, most of the players are expected to play the risk-dominant strategy.

At $b = \frac{1}{2}\beta J > 1$, on the other hand, the limiting distribution is bimodal with both modes close to the extremes. The most likely states are those where almost all players play the same action. We refer to this phenomenon as *spontaneous coordination*. Note that the global maximum depends on a (more specifically depends on h) and is close to the risk-dominant equilibrium. This is illustrated in Figure 2. The plotted distribution is bi-modal, with the highest peak close to $N = 80$. We note that for high values of J (benefit from coordinating), most of the distribution mass is concentrated around the highest peak, so the players will coordinate on the risk-dominant strategy, which in this example is s_1.

Coordination is thus not a limit phenomenon, but may emerge even if individual choice is characterized by substantial noise. For high noise levels (β close to 0) spontaneous coordination will emerge for sufficiently high payoff differences between the coordinated and miscoordinated case captured by parameter J. This can be clearly seen in Figure 3: even though β is close to 0 (in our example we set $\beta = 1.7$), increasing J determines a shift of the highest peak towards the risk-dominant strategy. The issue of risk dominance captured by the parameter h is of secondary importance. It only determines which state is more likely on average. This insight is hidden in the double-limit analyses, since an increase in β simultaneously changes both parameters. One could imagine to test this model using experimental data. Then our model would predict that for low levels of J coordination will not be present, but will suddenly emerge as we increase the benefits from coordinating.

Figure 2: Highest peak is close to $N = 80$

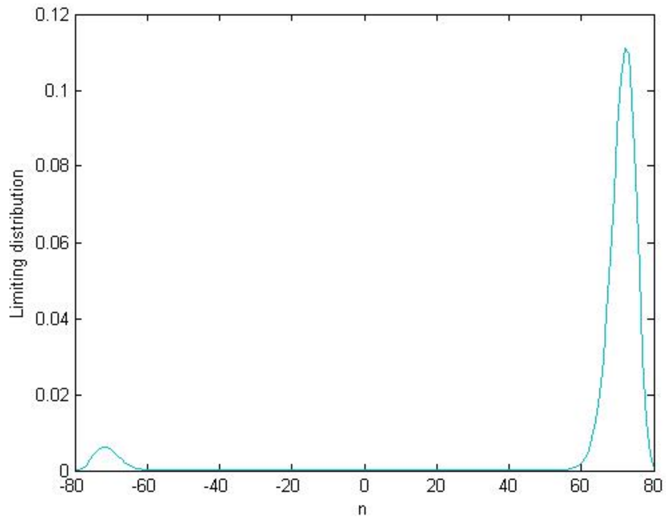
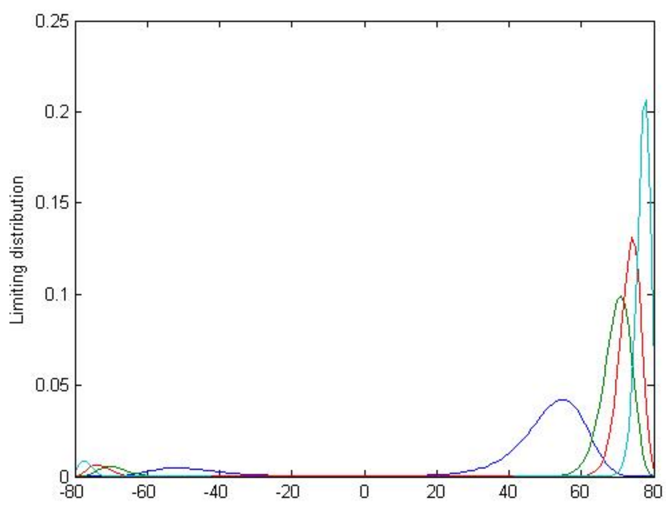


Figure 3: Increasing J shifts the mode towards the risk-dominant strategy



CONCLUSION

In this paper we study large potential games with global random matching. This approach allows us to analyze coordination even if individual choice behavior exhibits substantial levels of noise. The usual selection results (e.g. Blume (1993)) are derived as a corollary. For the case of many actors we derive a simple closed form representation of the unique stationary distribution. We show that coordination is not a limited phenomenon, but may occur even for substantial noise levels, depending on the relative benefits from coordinating as specified by the payoff matrix.

REFERENCES

Bergin, James, and Barton l. Lipman. 1996. "Evolution with State-Dependent Mutations." *Econometrica* 64(4):943-956.

Blume, Lawrence E. 1993. "The Statistical Mechanics of Strategic Interaction". *Games and Economic Behavior* 4:387-424.

Blume, Larry. 1995. "The Statistical Mechanics of Best-Response Strategy Revision." *Games and Economic Behavior* 11: 111-45.

Blume, Lawrence E. 1997. "Population Games". In W. Brian Arthur, Steven N. Durlauf, and David A. Lane, eds. *The Economy as an Evolving Complex System II.* Reading: Addison-Wesley.

Chong, Dennis. 1991. *Collective Action and the Civil Rights Movement.* Chicago: University of Chicago Press.

Foster, Dean, and H. Peyton Young. 1990. "Stochastic Evolutionary Game Dynamics". *Theoretical Population Biology* 38: 219-232.

Fudenberg, Drew, and David K. Levine. 1998. *The Theory of Learning in Games.* Cambridge, Mass.: MIT Press.

Kandori, Michiro, George Mailath, and Raffael Rob. 1993. "Learning, Mutation, and Long-Run Equilibria in Games." *Econometrica* 61:29-56.

McFadden, David. 1973. "Conditional Logit Analysis of Qualitative Choice Behavior." in P. Zarembka, ed., *Frontiers in Econometrics.* New York: Academic Press.

Monderer, Dov, and Lloyd Shapley. 1996. "Potential Games." *Games and Economic Behavior* 7:62-91.

Weibull, Jorgen W. 1997. *Evolutionary Game Theory.* Cambridge: The MIT Press.

Weidlich Wolfgang. 1991. "Physics and Social Science - The Approach of Synergetics". *Physics Reports* 204(1):1-163.

Young, H. Peyton. 1993. "The Evolution of Conventions". *Econometrica* 61:57-84.

Young, H. Peyton. 1998. *Individual Strategy and Social Structure.* Princeton: Princeton University Press.

Saturday, November 17, 2007

Social Simulation Applications

Parallel Track I

Social Interaction and Cognition

LEGAL AGENTS: AGENT-BASED MODELING OF DISPUTE RESOLUTION

M. T. K. KOEHLER,[*] George Mason University-Center for Social Complexity

ABSTRACT

In 1983 Cover introduced the terms jurisgenerative and jurispathic to the legal community. These terms describe particular styles of judicial decision-making. Jurisgenerative judicial decisions closely resemble mediation. When a judge attempts to create a jurisgenerative result they craft a decision that takes into account both sides and thus create a novel outcome or legal theory. When a judge follows a jurispathic approach they will find the plaintiff or defendant to be the entire winner and, in effect, kill a legal line of reasoning. The author is not aware of any attempt to systematically understand the utility of one judicial decision-making method over another. Presented in this paper is an agent-based model that explores the effects of judicial decision-making on a population of agents trying to extract resources from the environment. When agents have a dispute about which strategy to use they "sue" each other. The decision method (jurispathic or jurisgenerative) and the amount of information the judge has about both the strategies used by the agents as well as the global optimal strategy is varied. Finally, agents may circumvent the trial altogether and settle the dispute with a wealth transfer. The likelihood of settlement is a function of local precedent from other suits and the settling agent's wealth. In this way very successful agents can maintain their strategy by paying a "fee" rather than risk a trial. The paper will conclude with a description of next steps and a brief discussion of the use of agent-based modeling for jurisprudential research.

INTRODUCTION

Laws pervade all aspects of our lives. They are an integral part of our social system—they shape our actions as individuals and as groups. Yet the legal system is not above us. The laws are made and interpreted by us and, therefore, are subject to many of the same dynamics. The pressures of a static versus a dynamic legal system are perhaps some of the most interesting. A legal system that does not change cannot continue to be relevant and function well. However, a legal system that changes too frequently will be impossible to understand and will increase uncertainty within society to untenable levels. This paper begins to explore the effect of judicial behavior on these dynamics.

In 1983 Cover introduced the terms jurisgenerative and jurispathic judicial decision-making and opened an extremely interesting dialogue in the legal community (Cover 1983). Jurispathic judicial decision-making is the notion that courts "kill" a line of legal reasoning when a winner and looser are chosen in a case. This can effectively end the line of legal reasoning presented by the losing side. This is especially true in instances that involve alternative meanings to a single law (Henderson 1991). At times, this is seen as necessary because "noise" will tend to build up within a legal system. As laws move through time they have a tendency to evolve new and different meanings, especially through spatial diffusion into various

[*] Corresponding author address: Matthew Koehler, 7515 Colshire Dr., MailStop H305, McLean, VA 22102; e-mail: mkoehler@gmu.edu.

communities. These time-generated meanings will create noise, uncertainty, and confusion within the legal system, it is here the courts feel a need to perform jurispathic pruning (Juarrero-Roque 1991). This is also important from an economic perspective as increasing the uncertainty of a system may also increase the transaction costs associated with doing business within it.

Conversely, Jurisgenerative operations generate legal meanings. This can occur with or without control or direction, potentially creating anarchy (Kahn 1989). A jurisgenerative outcome synthesizes the legal solutions from the alternative legal perspectives raised before the court by the interested parties. This solution takes into account the needs, positions, perspectives, and so on of all parties before the court. Though one party may leave with more of the "legal pie" than the rest, no legal perspective is completely "killed" in the procedure. In this way, jurisgenerative decision-making has a very mediation-like feel to it. The problem, of course, is that the parties cannot necessarily count on an outcome driven by precedence (aspects of this topic were explored in earlier work by Isaac Dilanni (Dilanni 2006), which explored the evolution of common (judge made) law). This may tend to increase transaction costs within the system. However, there are times where both parties do have interests that should be taken into account during judicial review—custody disputes are perhaps a good example.

To my knowledge these ideas of judicial decision-making have not be examined systematically to determine their potential effects on the "fitness" of society. The intuition is as follows: A population of evolving agents should converge on an optimal resources extraction strategy (in the model this is essentially pattern matching, optimal means an agent has the list of numbers as the environment) for their local environment at some particular rate. If they are "told" to use some other resource extraction strategy it may change the evolutionary rate of convergence. If the new resource extraction strategy is imposed with no knowledge of or regard to the agent's initial extraction strategy or the resource availability pattern, then it should have an effect similar to massive semi-random mutation and increase the time required to converge on the optimal resource extraction strategy. If the new resource extraction pattern is imposed with some knowledge of or regard to the fitness of the agent's initial extraction strategy then it should have the effect of large scale, pointed selection and reproduction creating more optimal resource extraction strategies, thus lowering the time required to converge on the optimal resource extraction strategy.

THE MODEL

The landscape

The model that was used in this examination was created using NetLogo 3.1 (Wilensky 1999). This model utilizes agents living on a torus landscape and extracting resources. Agents will change their extraction strategy only by judicial edict. In the homogeneous environment, resources are available to agents in the following pattern: 1, 2, 3, 4, 5. Finally, patches are instantiated with two variables penalties-here and mean-penalties. These two variables are used to keep track of local penalties imposed on defendants that lose at trial. As will be discussed *infra* this value is used to determine settlement offers before the trial commences.

Initialization of the agents

Agents are created as a heterogeneous population with 1000 members. Initially they are randomly placed on the torus. Each agent is given a random metabolic rate which is a random number drawn from a normal-distribution that is unique to each agent. The metabolism level is drawn every time step, in this way agents have a heterogeneous metabolic rate. Agents are also given a "de minimis" level for damages that they suffer during the model run. Damages that are less than their de minimis level are not pursued via trial. Each agent also starts with a random amount of resources drawn from a random uniform distribution from 0 to 99. All agents also start with an initial feeling that they have a 50% chance of winning at trial as either the plaintiff or defendant. Finally, each agent is given a resource extraction strategy. This is simply a list of 5 numbers, each number being between 1 and 5 (uniformly distributed).

Model Runtime Procedures

Model runtime procedures are really quite simple. Patches are asked to calculate their mean penalties if any penalties have been doled out on them. Then the agents that are not actively engaged in a suit are asked to update their "wealth" based upon their metabolism and resource extraction strategy, move, and then interact with other agents that are co-located with them. Finally, data is collected, the clock is iterated, and output is written to a file.

Updating agent wealth

Agent wealth is a function of their resource extraction strategy and their metabolism (settlement and trials will also affect agent resources, see *infra*). Equation 1 is the general income function:

$$wealth_t = wealth_{t-1} + income - metabolism. \qquad \text{Eq. 1}$$

Income is a function of the agent's extraction strategy and is determined in the following way: Each position in the agent's extraction strategy is compared to the same position in the "environmental availability strategy." If the agent's strategy number matches that of the environment they are given that amount of wealth. If the agent's strategy number is less than the environment the agent is penalized by 2 * the difference; if, however, the agent's strategy number is more than that of the environment the agent is given .5 * the difference. Equation 2 describes this system:

$$income = \sum_{n=1}^{5} x_n \begin{cases} = s_{an} iff\,(s_{en} - s_{an} = 0) \\ = -2(s_{en} - s_{an}) iff\,(s_{en} - s_{an} > 0) \\ = 0.5(s_{an} - s_{en}) iff\,(s_{en} - s_{an} < 0) \end{cases}, \quad \text{Eq. 2}$$

where s_{an} is the strategy of the agent at position n, s_{en} is the strategy of the environment at position n. In this way agents have an income that will range from positive 15 to -20. As mentioned *supra*, agent metabolism is simply a random variable drawn from a normal distribution with a mean and standard deviation that is unique to the agent.

After an agent has updated its wealth it moves to a new location. The movement algorithm that the agents use is very simple. Specifically, agents set their heading equal to their current heading plus a random jitter drawn form a normal distribution with a mean of 0 and a standard deviation of 10. Finally, the agents move forward one unit. Upon completion of their movement the agents interact with another agent if there is another agent in their immediate vicinity. Interaction begins by updating an agent's feelings of their probability of winning if they go to trial. This is a function of their current feelings on the matter and how those feelings relate to the probability associated with the area in which they find themselves. First, the agent sets their internal win probability equal to the number of times they have won divided by the number of times they went to trial. Then if that value is less than the probability of win in their area they increase their probability by half the difference between the two numbers.

Once this update is complete the agent looks for a neighbor agent in the immediate vicinity that is not involved in a suit and establishes them as a partner. Once the agent chooses a partner there is a 1/3 chance that the partnership will lead to some level of damages. Damages are a function of the difference in strategy between the two agents—simply the sum of the absolute value of the difference between the two strategies at each position. If damages occur the damaged agent compares the size of the damages to their de minimis level. If the damages are greater than their de minimis level then the agent "files a law suit" against the offending agent. Once the agent creates a suit the two agents try to settle the suit before it goes to court. To calculate the settlement and determine if it will end the suit the potential plaintiff and defendant go through the process outlined in Table 1.

Table 1. The pre-trial process within the model.

Pre-Trial Settlement Steps
1) Calculate the expected value of the damages they will receive if they go to court, this is determined by multiplying the damages value by the plaintiff's feeling about their chances of winning.
2) The defendant agent prepares a settlement value by taking the mean damages value from the region they are in and multiplying it by 1 - their notions of their chances of winning at trial. That value then becomes the settlement value.
3) If the settlement value is greater than 25% of the expected value of the damages at trial and less than the full value of the expected value of the damages at trial that the plaintiff expects then,
4) The plaintiff checks to determine if their expected damages value minus the settlement is less than their de minimis value. If it is then the agents settle the lawsuit; if it is not,
5) Then the defendant has the option to double the settlement amount if it has a positive wealth. The same calculus applies to the plaintiff when trying to determine if they will accept the second settlement offer.
6) If it does not fall within the 25%-100% range the defendant picks a new value for the settlement. The new value is drawn from a random uniform ranging from 0 to actual damages – 1. The same steps as described above hold for the plaintiff's calculus when determining whether or not to accept the new offer.
7) Again, if the damages minus the settlement offer is not less than the plaintiff's de minimis level then the defendant has the ability to double the offer if they have positive wealth.
8) If a settlement occurs then the defendant's wealth is decremented by the settlement amount and the plaintiff's wealth is incremented by the settlement amount. No changes are made to the strategies used by either agent.

If the agents do not settle then they move to trial. What happens at trial depends upon the judicial decision-making methodology in use and the amount of information that the judge has available to them. As mentioned above, there are two potential methods for judicial decision-making: jurispathic (where one side wins everything and one side looses everything), and jurisgenerative (where both sides are seen as having legitimate claims to the outcome of the suit). There are three levels of information available to the judge: none—meaning the judge makes a random decision, some—meaning the judge has aggregated information about the strategies being used by the agents, and total—meaning the judge knows exactly how well the strategies work at resource extraction in every position. The type of judicial decision-making and the level of information available are both user defined and not modifiable by the agents.

At trial first a comparison is made of both strategies relative to the ideal strategy (one that perfectly matches the environmental strategy). This comparison is simply the sum of the absolute value of the deviation at each position of the strategies from the environmental ideal. Then a winner is chosen. If the amount of information is none, the winner is chosen at random. If the decision methodology is jurispathic then the strategy of the winner is given to the loser. If the methodology is jurisgenerative then the two strategies are blended randomly and given to the two agents. If some information is known to a jurispathic decision-maker then a winner is chosen by picking the strategy with the smallest deviation of the environmental optimal. Once again, the winning strategy is given to the loser. If some information is known to a jurisgenerative decision-maker then, as before, a winner is chosen based upon the smallest deviation from environmental optimal; however, this time the strategies are blended. The winning strategy is used for the 1, 3, 5 positions and the losing strategy is used for the 2 and 4 position. Finally, if a jurispathic or jurisgenerative decision-maker has total information then a winner is chosen based upon the number of times the strategy has a smaller deviation than the other with respect to the environmental ideal. If the decision-making process is jurispathic the winning strategy is then given to the loosing agent. If however, the decision was made jurisgeneratively, then the strategies are blended based upon each strategy's deviation from the environmental ideal by position.

RESULTS

For this initial examination a full factorial design of experiments was used. Testing both judicial decision-making regimes, all levels of information, and whether or not agents were allowed to double their initial settlement offer. Each run consisted of 1000 time-steps. Unfortunately, the same random-seeds were not used for all runs making direct comparisons of the results among the runs problematic. As discussed *supra* the environment is relatively harsh from the perspective of agent wealth accumulation. Figure 1 shows the mean wealth accumulated by agents over the course of the runs. As can be seen, most agents are unable to accumulate positive wealth and remain "in debt" over the entire run.

FIGURE 1: Mean Agent Accumulated Wealth.

As one would expect, runs that used no judicial decision-making, or judicial decision-making with no information, produced results with lowest agent wealth accumulation. On the other hand, runs that used judicial decision-making that included some or complete information produced runs in which agents were better able to accumulate wealth. Of note in Figure 1, is that it takes approximately 75 time steps to begin to see the differences. This leads one to hypothesize that it takes a certain amount of time for the judicial decisions to have an effect, meaning there needs to be a certain number of decisions so that a certain proportion of agents are affected before macroscopic effects are seen.

A better view of the relative effect of the judicial decision-making regimes is contained in Figure 2. This figure depicts the change in the mean agent's ability to extract resources in a single time step. Put another way, it is a depiction of how close to the environmental ideal the average agent is. As can be seen, there appears to be three basic groups: a) Jurisgenerative decision-making with total or some information, and jurispathic decision-making with some information; b) Jurispathic decision-making with total information; and c) all other types of judicial decision-making.

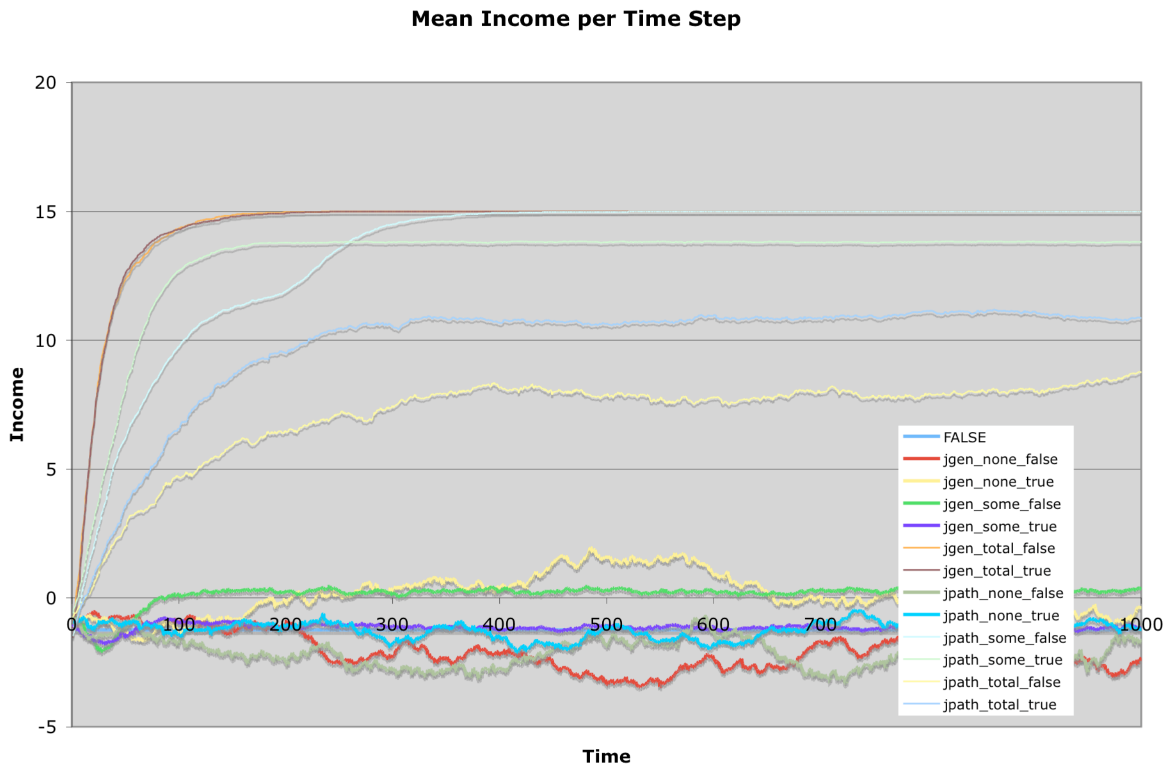


FIGURE 2: Mean Accumulation of Wealth per Time Step

As shown in Figure 2, the only regimes that make a difference are those that use some or total information. When there is a difference it is quite pronounced moving the mean from approximately zero to more than seven resource units extracted per time step. It is interesting to note that the best judicial decision-making regime appears to be jurisgenerative with complete information. However, this is not the case with jurispathic decision-making. Oddly, jurispathic decision-making with partial information seems to be better than that of jurispathic decision-making with complete information. Other metrics of interest from the model include total number of trials and total number of settlements. Figures 3 and 4 show these metrics respectively.

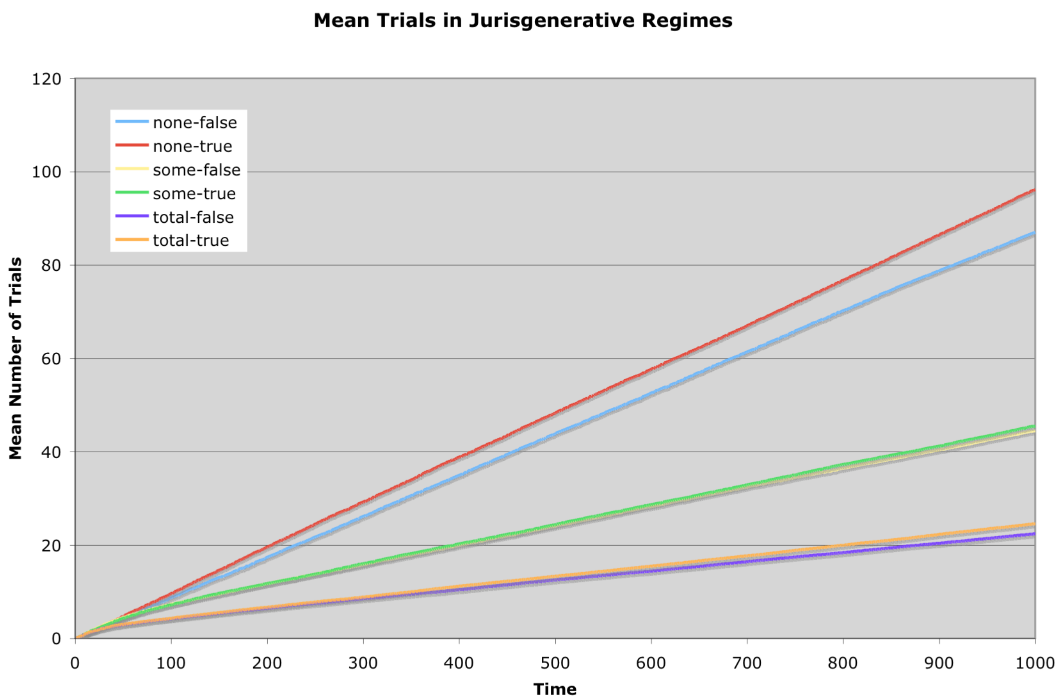


FIGURE 3(a): Mean Trial in Jurisgenerative Regimes

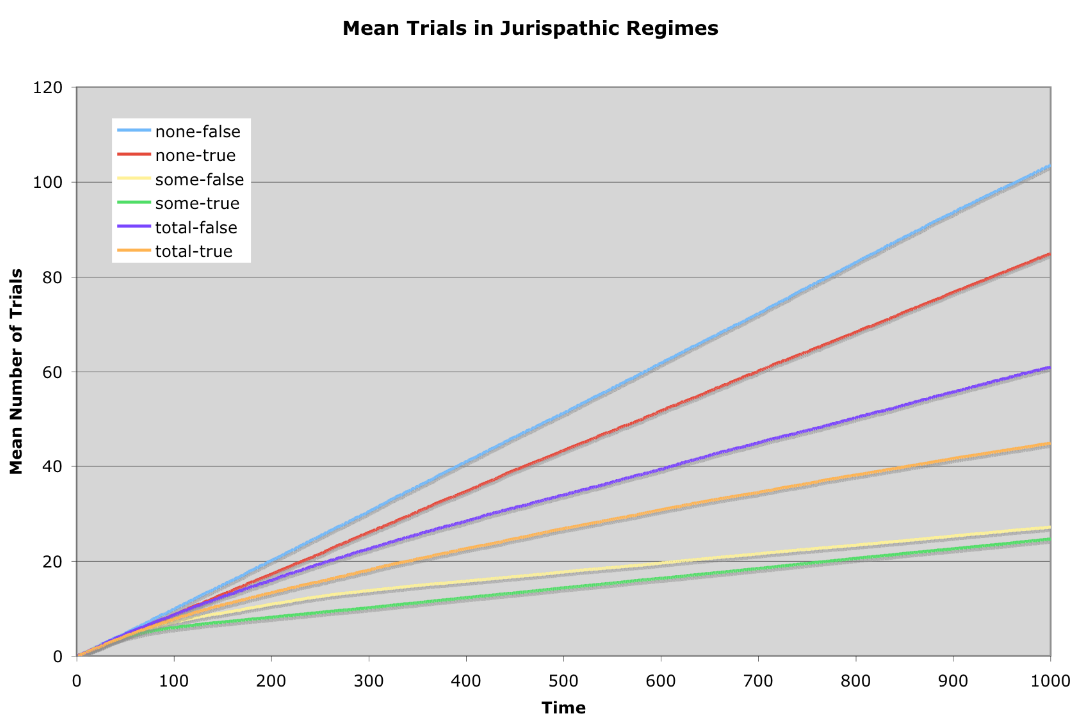


FIGURE 3(b): Mean Trials in Jurispathic Regimes

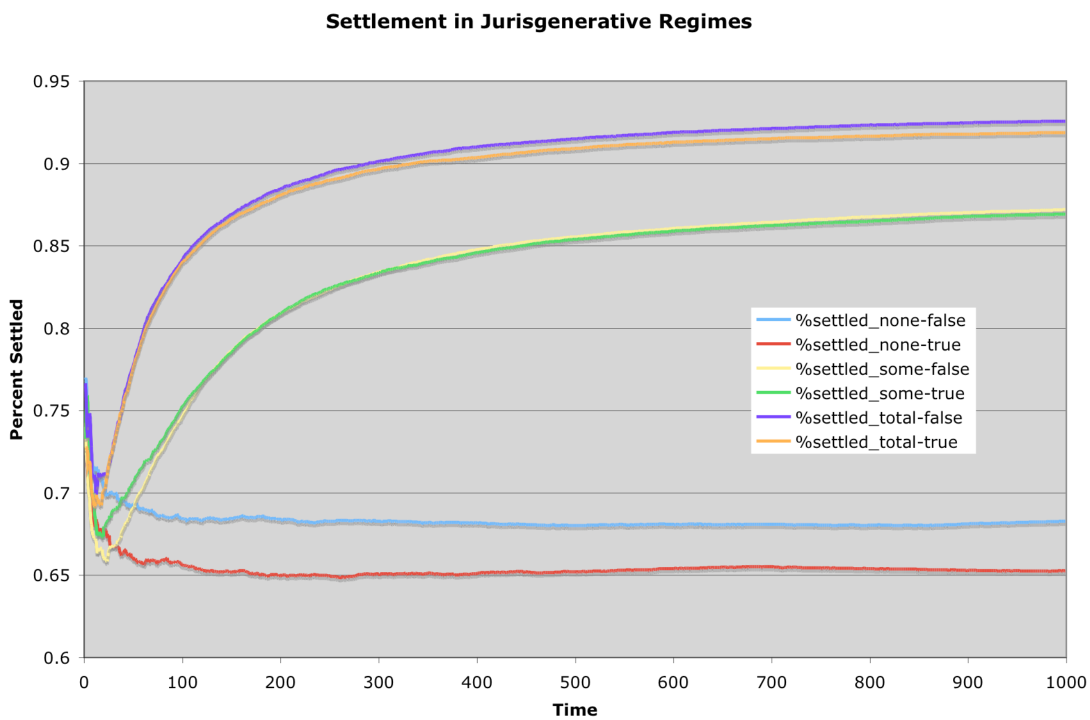


FIGURE 4(a): Jurisgenerative Regime Settlement Rates

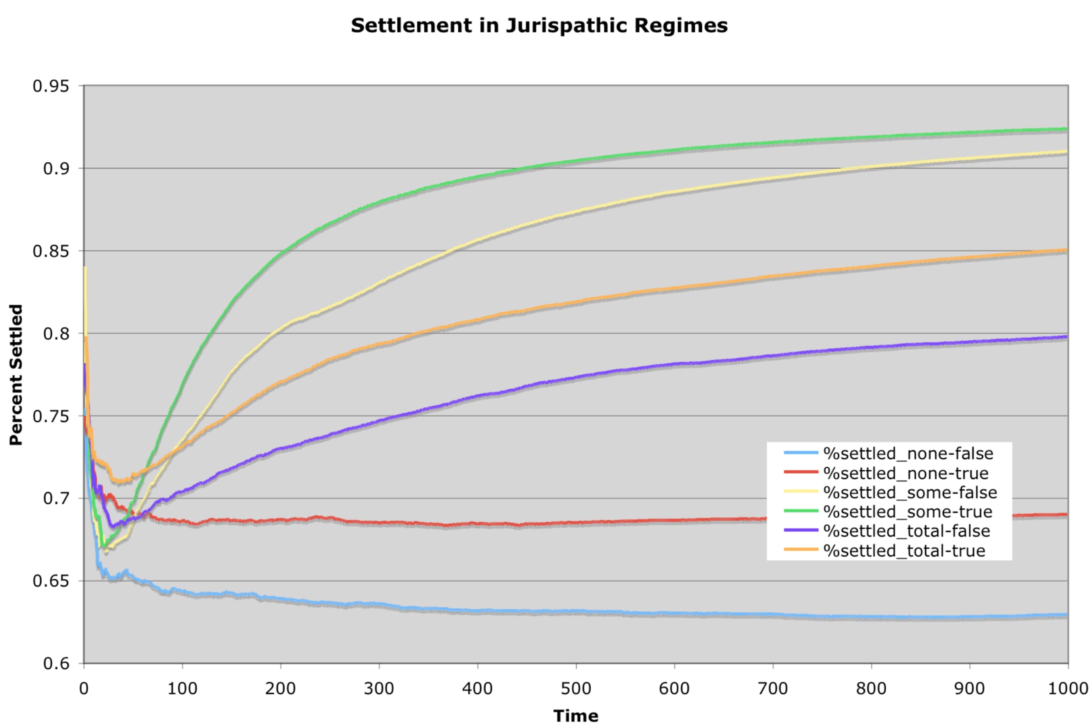


FIGURE 4(b): Jurispathic Regime Settlement Rates

Figure 3 shows the mean trials for agents under various regimes. As seen in other results there are three basic groupings. In Figure 3(a) these groups are most obvious, with total information producing fewer trials than some information. Finally, judicial decision-making that is random (no information) produces the most trials. Once again it is worthy of note that in the case of jurispathic decision-making the relative order of total information and some information is switched, with some information being generally better than total information (see Figure 3(b)).

Figure 4 shows the percentage of settled cases over the course of the runs. Once again we see three groups divided along the lines of information with the order of partial vs. complete switched from jurispathic to jurisgenerative decision-making. What is particularly remarkable is that the settlement rate of the highest performance judicial decision-making (jurisgenerative with complete information and jurispathic with partial information) begins to converge on realistic settlement rates. It is estimated that only between 2% and 7% of cases that are filed actually go to trial (Posner 1998). Both jurisgenerative decision-making with complete information and jurispathic with partial information begin to reach those values.

CONCLUSION

The current model shows promise as a method to test the utility and effect of judicial decision-making regimes. Using the system introduced by Axtell and Epstein (Axtell 1994 and Axtell 2005) to evaluate the empirical substance of the models described in this study, one could say that it is Level 0, i.e. the code functions properly, and may be beginning to approach Level 1 as it shows some similarity to actual macro-level dynamics, namely settlement dynamics. In principle it should be possible to move the model firmly to Level 1. Furthermore, there are large amounts of data available on our legal system, this should make it possible to move, at least parts of it, to Level 2. It would be a very serious challenge to move into Level 3, however, it may be possible.

This model represents a first step towards a careful examination of the effects of judicial decision-making in our legal system. To fully appreciate the effects of jurisgenerative and jurispathic decisions, and the effects that differing amounts of information will have on these decisions, the next generation of models must perform much more sophisticated analyses on the resource extraction strategies of the agents. Transaction costs must be included, as well as the effects and costs of information and precedence. Only after inclusion of these important features could an informed study be performed regarding reforms that may be useful to the judicial decision-making process. Obviously it is impossible to create a decision-making regime with complete information; therefore, it is particularly intriguing to see results indicating that partial information in a jurispathic regime is better than complete information in that regime. Hopefully, this study is a first step in that direction and will help to inform discussions of jurispathic and jurisgenerative decision-making and to begin to articulate the utility agent-based modeling could have in jurisprudential study. Given the inroads made by Economics (of particular interest is Behavioral Law and Economics) and Game Theory, incorporation of agent-based modeling seems only natural.

REFERENCES

Cover, R., 1983, "The Supreme Court, 1982 Term Foreword: Nomos and Narrative," 97 Harv. L. Rev. 1, 4.

Henderson, L., 1991, "Authoritarianism and the Rule of Law," 66 Ind. L. J. 379, 405.

Juarrero-Roque, A., 1991, "Fail-Safe versus Safe-Fail: Suggestions Toward an Evolutionary Model of Justice," 69 Tex. L. Rev. 1745, 1771.

Kahn, P., 1989, "Community in Conventional Constitutional Theory," 99 Yale L. J. 1, 59.

Dilanni, I., 2006, "Evolution and Inefficiency," Unpublished manuscript obtained from Professor Axtell, George Mason University—Center for Social Complexity.

Wilensky, U., 1999, NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL.

Richard Posner, R., 1998, *Economic Analysis of Law*, 5th Ed. Aspen Law & Business, New York, NY.

Axtell, R. and Epstein, J., 1994, "Agent-Based Models: Understanding Our Creations." Bulletin of the Santa Fe Institute.

Axtell, R., 2005, "Three Distinct Kinds of Empirically-Relevant Agent-Based Models." The Brookings Institution Working Paper.

AGENT-BASED MODELING OF USABILITY FROM A DISTRIBUTED COGNITIVE PERSPECTIVE

J. EDEN*, The iSchool, Drexel University

ABSTRACT

Many designers are moving beyond traditional Human Computer Interaction (HCI) views of design, now viewing design as more than the design of individual products to interact with in a snapshot of time, and instead designing interactions among people, artifacts, and information (Saffer, 2006) over dimensions such as time, space, and social structures. Interaction designs must use some instance of the context of use as the substrate of evaluation; in many cases, these instances of the context of use are necessarily artificial (e.g. lab-based usability testing), or without real users (e.g. discount usability evaluation methods such as Heuristic Evaluation, or the Cognitive Walkthrough). Although much research and practice has been carried out under the name user-centered design, research paths that would help close this gulf of consequence for designers may be best called use-centered design, moving context of use to the front as a more appropriate focus of analysis (i.e. users are but one subset of actors in the context of use). We propose a path towards closing this gulf of design consequence by focusing on ways to better foster reflection-in-action during interaction and user experience design through the use of computational simulation of both users and contexts of use. This proposed path leverages and integrates existing research and practice in agent-based modeling and theories of distributed cognition. An important goal of this paper is to start a dialogue with the modeling community, and to get feedback on the idea of modeling distributed cognitive perspectives of contexts of use, as part of the usability engineering process.

Keywords: Distributed Cognition, Usability evaluation, agent-based modeling

INTRODUCTION - GULF OF CONSEQUENCE IN DESIGN

Many designers are moving beyond traditional Human Computer Interaction (HCI) views of design, now viewing design as more than the design of individual products to interact with in a snapshot of time, and instead designing interactions among people, artifacts, and information (Saffer, 2006) over dimensions such as time, space, and social structures. This view of design goes by many differing names such as Interaction Design, Service Design, and User Experience Design, but all share the idea that design thinking needs to transcend static (in time, space, and social dimensions) views of how people, artifacts, and information interact.

A science of design of interactive systems can be seen as differing from many other (natural) sciences such as physics, chemistry, and mathematics with respect to the substrate of evaluation; many fields have well understand rules and relations between these rules that can be used to come to conclusions prior to implementation, whereas interaction designs must use some instance of the context of use as the substrate of evaluation; in many cases, these instances of the

*Corresponding author email address: Joel Eden, joel.eden@gmail.com

context of use are necessarily artificial (e.g. lab-based usability testing), or without real users (e.g. discount usability evaluation methods such as Heuristic Evaluation, or the Cognitive Walkthrough).

Making design choices without the ability to evaluate the consequences widens the gulf between the designer's model of and the reality of the consequences of these design moves in the context of use. Although much research and practice has been carried out under the name user-centered design, research paths that would help close this gulf of consequence for designers may be best called use-centered design, moving context of use to the front as a more appropriate focus of analysis (i.e. users are but one subset of actors in the context of use).

We propose a path towards closing this gulf of design consequence by focusing on ways to better foster reflection-in-action during interaction and user experience design through the use of computational simulation of both users and contexts of use. This proposed path leverages and integrates existing research and practice in agent-based modeling and theories of distributed cognition.

EXTENDING COGNITION BEYOND THE BRAIN

Most design and evaluation methods in the interaction design and usability fields are grounded in specific theories of cognition. For example, methods may be based on cognitivist theories, such as the popular Information Processing and Symbol Systems models of cognition, which place the concept of cognition physically in the human brain. Alternatively, methods may be grounded in a fundamentally different theoretical perspective, such as post-cognitivist theories of cognition (e.g. Distributed Cognition, Activity Theory, Situated Cognition), which extend the view of cognition and the mind beyond the physical boundaries of the human skull.

Although intelligence, cognition, and knowledge are words that are usually associated with an individual person, and moreover usually that individual's brain, recent views in the field of cognitive science (Clark, 1997) are questioning the physical boundary of the human brain as the boundary for the concept of intelligence, cognition, and knowledge. Knowledge can be viewed as residing "in the world," as well as the more commonly held view of knowledge "in the head." (Norman, 1993).

Distributed Cognition

Distributed Cognition (Hutchins 1995a; Hollan el al., 2002, Rogers, 1997) is a theory of cognition that views cognitive functionality as "computation as propagation of representational state across representation media." Three main principles of Distributed Cognition as a theory of cognition which distance it from other cognitive theories are listed below:

- Cognitive processes may be distributed across the members of a social group.
- Cognitive processes may involve coordination between internal and external (material or environmental) structure.
- Processes may be distributed through time in such a way that the products of earlier events can transform the nature of later events.

(Clark & Chalmers, 1998) makes clear (at least relative to the cognitivist view) why it is proper to refer to cognition outside of the brain as 'cognition,' "If as we confront some task, a part of the world functions as a process which, *were it done in the head*, we would have no hesitation in recognizing as part of the cognitive process, then that part of the world is (so we claim) part of the cognitive process. Cognitive processes ain't (all) in the head!" (emphasis present in original text)

Distributed Cognitive Tasks

Research in Distributed Cognitive Tasks focuses on how differing isomorphic distributions of information and operations required by an abstract task structure over internal and external representations effects the resultant usability of a system (Zhang & Norman, 1994; Zhang, 1997a; Zhang, 1997b). Agent-based modeling can be used to try out many different 'what if' scenarios represented by different isomorphs that represent different design alternatives related to whether or not a human user or a non-human artifact is responsible for accomplishing a portion of a task.

Distributed Intelligence

Pea discusses 'Distributed Intelligence' as the idea that through the use of external artifacts that have been imbued with intelligence, intelligence is accomplished rather than possessed (Pea, 1993). Pea 'problemitizes' the concept of scaffolding as it has been recently used to represent too much of a percentage of external learning and activity aids. Pea argues that 'distributed intelligence' is a more appropriate term for non-fading scaffolding (humans will always make use of the external aid) versus fading-scaffolding (learning and activity aids that will cease to be used once the desired skill or knowledge is learned) (Pea, 2004).

Pea (1993) states, "objects…have become so deeply a part of our consciousness that we do not notice them. Turned from history into nature, they are invisible, un-'remarkable' aspects of our experiential world." "These tools literally carry intelligence *in* them, in that they represent some individual's or some community's decision that the means thus offered should be reified, made stable, as a quasi-permanent from, for use by others."

This means that we need to keep questioning our assumptions about what role artifacts and the environment play in the process of cognition. Pea stresses that educational activities should focus more on making learners aware of the usefulness of imbuing intelligence into artifacts; i.e. educating learners in how to notice, construct, and evaluate such intelligent tools, rather than only focusing on the use of already existing tools. This in a sense is saying that methods such as the Distributed Cognitive Walkthrough (that aid us in noticing the intelligence, or lack of it, in the external world) should be used in the educational process of even young learners.

Distributed Cognition as an Emergentist view of Cognition

Andy Clarks refers to distributed and embodied cognition (Clark, 1997) as an emergentist view of cognition, in that cognition can be viewed as an emergent phenomena that arises from the interaction of multiple cognitive resources (i.e. relevant here is the embodied interaction between external and internal cognitive resources). This emergentist view can be thought of as contrasting the more Cartesian, or reductionist flavor of cognition typically seen in characterizations of the cognitivist views of cognitive science; moreover, these reductionist views are closely related to the foundational views of cognition that underlie existing usability evaluation techniques such as the Cognitive Walkthrough method. Clark proposes (Clark, 1997) the use of tools and methods from complexity science (i.e. fields related to the study of emergence) for the study of cognition (when using a distributed and embodied view of cognition).

PREVIOUS RESEARCH APPLYING RELATED THEORIES TO USABILITY EVALUATION AND DESIGN

The application of theories of distributed cognition (and related theories) for design and evaluation of usable systems has primarily been carried out by researchers and practitioners with much time and education invested in these theories (Hutchins, 1995; Zhang & Norman, 1994; Zhang, 1997a; Zhang, 1997b). The majority of uses of distributed cognitive principles in system design and evaluation more closely resemble ethnographic studies (Hutchins, 1995b) than the type of evaluation typically carried out by interaction design and usability practitioners (e.g. Cognitive Walkthrough or Heuristic Evaluation) (Rogers, 1997). Moreover, these applications of distributed cognitive theories therefore require time and education requirements that are cost, time, and resource prohibitive with typical practitioner resources (Rogers, 1997). The Information Resources Model (Wright, et al., 1996), a model of HCI related interaction based on Distributed Cognition "identifies a number of interaction strategies" and discusses how different "information structures can be used as resources for action." Although the literature (Wright, et al., 1996) for this model provides examples of application of Distributed Cognition to design and evaluation related to HCI, there is no representation of this model as a 'method' to be used by theory-novice or evaluation-novice practitioners; it should be taken as a proof of concept that Distributed Cognition concepts can be used for the design and evaluation related to HCI (by researchers educated in Distributed Cognition).

The Distributed Cognitive Walkthrough (DCW) method

The Distributed Cognitive Walkthrough (DCW) method (Eden, 2007) uses concepts from distributed cognitive theory to view interaction between people and information as transcending interactions with graphical user interfaces, allowing the DCW method to be useful for evaluation of design ideas in many areas of interaction design; for example, evaluation of ubiquitous computing, service design (e.g. Starbucks customer/worker experience), and mathematical notations (e.g. Newton versus Leibniz Calculus notation).

The DCW is a walkthrough-style usability evaluation method based on theories of distributed cognition, Distributed Cognition (Hutchins, 1995a; Hollan et al., 2002; Rogers, 1997), Distributed Cognitive Tasks (Zhang & Norman, 1994), Distributed Intelligence (Pea, 1993; Pea 2004), and Embodied Cognition (Clark, 1997). The DCW method is useful for the identification of potential usability issues related to interaction between people, artifacts, and information, across dimensions such as time, space, and social structures. The DCW method will serve two objectives; 1) embodiment of principles of distributed cognitive theory, while also 2) being practically useful by novices in generating actionable information regarding potential usability issues.

While methods such as the DCW method guide designers in considering implications of their design choices from a distributed cognitive perspective, in many cases what it still needed are tools that allow for "what if" exploration to see non-intuitive implications; this is where agent-based modeling can be leveraged.

ABM FOR THE STUDY OF A DISTRIBUTED COGNITIVE VIEW OF USABILITY

Agent-based modeling to support the study of usability in multi-person and multi-artifact situations fulfils the need for new types of evaluation that provides timely “talk back” (Schon, 1990) to interaction designers who now realize they are designing for usability in “the wild” (Hutchins, 1995). There is a large opportunity to leverage tools such as agent based modeling to try ‘what if’ scenarios to see the results of design changes, without the prohibitive cost of involving real users (i.e. usability testing).

What we are modeling - How a Coffee Shop Remembers its Orders

(Spin off of Hutchins’ How a Cockpit Remembers its Speeds)

Scenarios to be modeled are taken from everyday situations where multiple people interact with multiple artifacts; for example, the domain of coffee shops, where many different complicated drink orders must be taken by workers from customers, and these drink orders must be tracked until the drinks can be made and given to the customer. Figure 1 below shows a Starbucks coffee cup that has six boxes on the cup where Starbucks workers can write directly on the cup to represent a specific drink that a customer ordered. The Starbucks workers do in fact memorize (long term internal memory) the defaults of all of the drinks that can be ordered, but the interesting point is that the variations on these defaults are marked up in the six boxes on the cup. Therefore, much of the specific drink information is represented externally, relieving the Starbucks worker from having to use more error-prone internal cognitive resources; moreover, the size of the physical cup itself represents the size of the drink ordered, serving as a good example of how information can be represented in inherent physical constraints of physical artifacts (i.e. the Starbucks cup does not need a box for drink size, because this information is inherently represented in the physical size of the cup). Many other popular (i.e. busy) coffee shops (e.g. Dunkin Donuts and Seattle’s Best Coffee) do not use such externalization of drink order information, resulting in a higher rate of drink errors, and in many cases requiring customers to take part in the tracking of whether or not the correct drink is being made as ordered.

Figure 1 Starbucks cup showing boxes that workers mark up to represent different drinks. This allows the workers to handle more concurrent drink orders than would be possible to keep track of using only internal cognitive resources.

Figure 2 below shows a ‘cheat sheet’ used by Waffle House restaurant chain grill cooks that is used in a manner similar to the example of the Starbucks coffee cup. The cheat sheet shows how grill cooks use different configurations of food condiments to represent large numbers of complicated, yet similar food orders. These configurations allow the grill cook to be working on more concurrent orders than would be possible to track using only internal cognitive resources. The plate itself represents information such as the size or type (e.g. eggs, steak) of the order, making use of real world inherent physical constraints to represent information, in the same way that the Starbucks cup itself represents the size of the drink ordered.

Figure 2 Waffle House grill cook cheat sheet. Different configurations of condiments placed on plates represent specific food orders. These configurations allow the grill cook to be working on more concurrent orders than would be possible to keep track of using only internal cognitive resources.

REFERENCES

Clark, A. (1997) Being There: Putting Brain, Body, and World Together Again, The MIT Press

Clark, A. & Chalmers, D. (1998) The Extended Mind, Analysis, 58.1, 7-19

Eden, J. (2007) The Distributed Cognitive Walkthrough method: A walkthrough-style usability evaluation method based on theories of distributed cognition, Proceedings of the 2007 Conference on Creativity and Cognition

Hollan, J., Hutchins, E., & Kirsh, D. (2002). Distributed Cognition: Toward a New Foundation for Human-Computer Interaction Research. In J. M. Carroll (Ed.) Human-Computer Interaction in the New Millennium. New York: ACM Press Addison Wesley), 2002, pp. 75-94. (Reprinted from ACM Transactions on Computer-Human Interaction 7(2), June 2000.)

Hutchins, E. (1995a): Cognition in the Wild, The MIT Press

Hutchins, E. (1995b). How a cockpit remembers its speed. Cognitive Science, 19, 265288.

Nielsen, J., Mack, R. (editors) (1994) Usability Inspection Methods, John Wiley and Sons

Norman, D. (1990) The Design of Everyday Things, Doubleday

Norman, D. (1993) Things That Make Us Smart, Perseus

Pea, R. D. (1993). Practices of distributed intelligence and designs for education. In G. Salomon (Ed.). Distributed cognitions. New York: Cambridge University Press, pp. 47-87

Pea, R. D. (2004). The social and technological dimensions of "scaffolding" and related theoretical concepts for learning, education and human activity, The Journal of the Learning Sciences, 13 (3)

Pinelle, D., Gutwin, C. (2002) Groupware Walkthrough: Adding context to groupware usability evaluation, In Proceedings of the SIGCHI conference on Human factors in computing systems

Polson, P, Lewis, CH, Rieman, J., and Wharton, C. (1992) Cognitive walkthroughs: A method for theory-based evaluation of user interfaces. In International Journal of Man-Machines Studies, 36, pp. 741-773.

Rogers, Y. (1997) A brief introduction to Distributed Cognition
Retrieved from http://www.slis.indiana.edu/faculty/yrogers/papers/dcog/dcog-brief-intro.pdf

Saffer, D. (2006) Designing for Interaction, New Riders

Schon, D. (1990). Educating the Reflective Practitioner, Jossey-Bass

Wright, P.C., Fields, B. & Harrison, M.D. (1996) Distributed information resources: A new approach to interaction modeling. In Proceedings of ECCE8: European Conference on Cognitive Ergonomics, 1996

Zhang, J., Norman, D. A. (1994). Representations in distributed cognitive tasks. Cognitive Science, 18, 87-122.

Zhang, J., Norman, D. A. (1995). A representational analysis of numeration systems. Cognition, 57, 271-295.

Zhang, J. (1997a). Distributed representation as a principle for the analysis of cockpit information displays. International Journal of Aviation Psychology, 7(2), 105-121.

Zhang, J. (1997b). The nature of external representations in problem solving. Cognitive Science, 21(2), 179-217.

SPY V. SPY: A UTILITY-BASED APPROACH TO AGENT-BASED ADVERSARIAL REASONING

P. BARRY,* The MITRE Corporation
G. JACYNA, The MITRE Corporation
M. KOEHLER, The MITRE Corporation

ABSTRACT

This paper introduces an agent-based model that utilizes a utility based framework for agent behavior that represents action propensity based upon the agent's volatility and the perception of the current environmental context compared to the agent's ideal context. Propensity for an agent to conduct an action is modeled as a logit function. As the environment changes and events occur around the agent, where they fall on the logit function will change. If the logit function exceeds a given value the agent will take an action. Moreover, the threshold that must be exceeded before the agent will take the aforementioned action is malleable. As events occur the agent may lower the threshold in an attempt to satisfice (making the best of a bad situation) based upon their perceived likelihood for success. For example, although a suicide bomber would like to attack a large crowd, they will attack a small crowd rather than fail completely by being stopped while looking for a large crowd. We experimented with these dynamics as part of an agent-based model of a large crowd moving through an entry point. Security personnel are modeled as are civilians and bombers. As sensors are a critical aspect of the scenario high resolution sensor simulations were created to determine the appropriate parameter values to use to create "realistic" sensor performance.

INTRODUCTION

This paper introduces a framework for agent-based behavior that models the propensity for action based upon the agent's volatility and the perception of the current environmental context as compared to the ideal context. The framework is instantiated as an agent-based model that examines the optimal sensor placement and security-force behaviors to thwart a terrorist operation. An extensive exploration through a large variety of simulation runs was conducted. This exploration resulted in avenues for additional simulation as well as the need to explore more carefully the role of tactics in conjunction with sensor technology in defeating terrorists. As part of the data analysis, the paper compares the model results to those from a formal equation based model as described by Kaplan and Kress (Kaplan 2005). Conclusions are drawn as to the additional value of representing the behavior in the agent-based model.

The construction of the paper is as follows. The concept of activation energy is presented as a basis for modeling the internal propensity of red agents to attack. Activation energy is then used as a lens to focus the discussion on satisficing behavior as well as how blue agents might defeat the red agents. The discussion is actualized in the following section that describes the simulation framework and presents the results. Implications for additional work and references close the paper out.

* Corresponding author address: Philip Barry, The MITRE Corporation, 7515 Colshire Drive, Mclean, VA 22102; e-mail: pbarry@mitre.org

Activation Energy

Extending the work done in modeling stability and support operations (Koehler 2004), we refer to the propensity for an individual to carry out an action as a function of a term called activation energy. Activation energy is modeled as a logit function, whose general form is described by Equation 1. If an individual's activation energy for a given situation exceeds an endogenous threshold then an action is taken.

$$f(x) = \frac{Lb + (Ub - Lb)}{1 + e^{-slope(k-m)}} \quad \text{(Eq. 1),}$$

where **Lb** and **Ub** are the lower and upper bounds of the function respectively. **Slope** is a parameter that adjusts the general steepness of the activation energy curve. In this framework, we suggest that one can view the **slope** parameter as a proxy for volatility; an agent with a large slope is considerably more unpredictable than an agent with a smaller slope. The mean of the function **m** is used as a proxy to model how close the context must be to the ideal value to increase activation energy. The values on the x-axis represent **k**, the average distance of the actual context parameters from the ideal values as measured in percent. (Negative values of **k** indicate that the average of the parameters is "better" than the mean of the activation energy curve). In this framework distance refers to the geometric distance within this vector space rather than physical distance in the simulation. It should be noted that we are defining one action per agent rather than creating a detailed goal hierarchy as other approaches (e.g., Johns 2001). We assume that a bad actor only has one goal, that of executing some terrorist action.

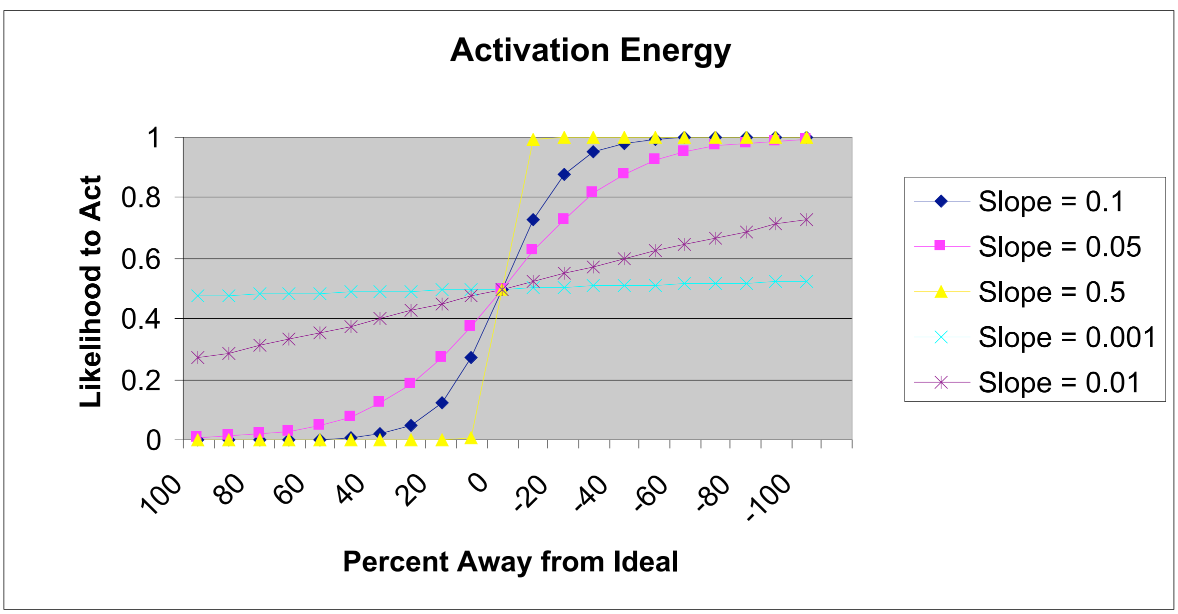


Figure 1 Illustration of Activation Energy

To illustrate this concept, consider Figure 1. In Figure 1, for all curves, **m** is set to zero and the behavior of various values of **k** are compared at specific **slope** settings. As the **slope** approaches zero, the tendency is for the behavior to linearize (e.g., **slope** = 0.001 and 0.01). As the **slope** increases to 0.05, the volatility of behavior increases and the distinctive "S" shape shows a consistent increase in activation energy exponentially increasing around 0 and leveling off as **k** approaches -100. At **slope** = 0.5 there is virtually no increase in activation energy until **k** = 0 at which time the activation energy approaches 1.

We can also use activation energy as an external indicator of the agent's tendency for action. Agents that have sub-threshold but measurable activation energy can be considered to be "telegraphing" their intention to act. Consider the case of an agent that shows no external indicators of untoward behavior until the conditions are ripe for action. This agent would be modeled with a large **slope** value (e.g., 0.5 in Figure 1). On the other hand, agents with smaller **slope** values (e.g., 0.01 in Figure 1) show their intentions and build up to activation much more slowly.

Representing Satisficing Behavior

Satisficing is a behavior which attempts to achieve at least some minimum level of a particular variable, but which does not necessarily maximize its value (see generally, Simon 1957). Often satisficing is used within a learning context as an alternative to strict optimization (Izquierdo 2004). It is reasonable and expected that the agents being modeled would exhibit satisficing behavior. Consider the example above, where the agent is taking the decision to carry out the negative action. Only when the agent is directly at the center of a crowded area will the activation energy be sufficient to trigger an action.

Satisficing is observed where the agent has determined that the optimal combination of parameters is too restrictive, in other words the allowable distance from the actual context to the ideal context is too large for activation. This possibility can be shown using figure 1, where the activation energy is shown to be below the mean until the agent is 100 meters from its target. When the agent reaches 100 meters from its target the activation energy reaches the mean value of 0.5. In the case where the activation energy threshold is 1.0, the agent will essentially need to collocate itself with the target prior to action. However, if the agent satisfices and reduces the activation energy threshold to 0.8, any distance less than 60 meters would suffice for the agent (given that the slope of the curve is 0.05).

Satisficing behavior is observed in the model when the agent perceives no other options. For example, if an agent perceives impending arrest the agent may trigger an explosive device even if the context is significantly far from optimal. This aspect of behavior is a significant factor that contributes to the adversarial reasoning. To thwart the undesirable actions of a given agent, the opposition will attempt to increase the requisite activation energy threshold or increase the perceived distance between the actual and ideal context variables.

EXPERIMENTAL FRAMEWORK

A simulation was developed to test these concepts using NetLogo 4.0 (Wilensky 1999). The simulation was designed to model an entrance to a public venue. The entrance is a non-torus plane 100 pixels high by 500 pixels wide. In the model there are five major regions to the entrance. Agents are instantiated at the extreme left edge. From here they move to the right into the second region, which is a wide hallway. At the end of the wide hallway the environment narrows to a region with two "turnstiles." All agents move through one of the turnstiles and proceed to the forth area, a narrow hallway. This narrow hallway ends in the final region, the extreme right side of the environment where all surviving agents are removed from the system. Figure 2 provides a screen shot that illustrates the model.

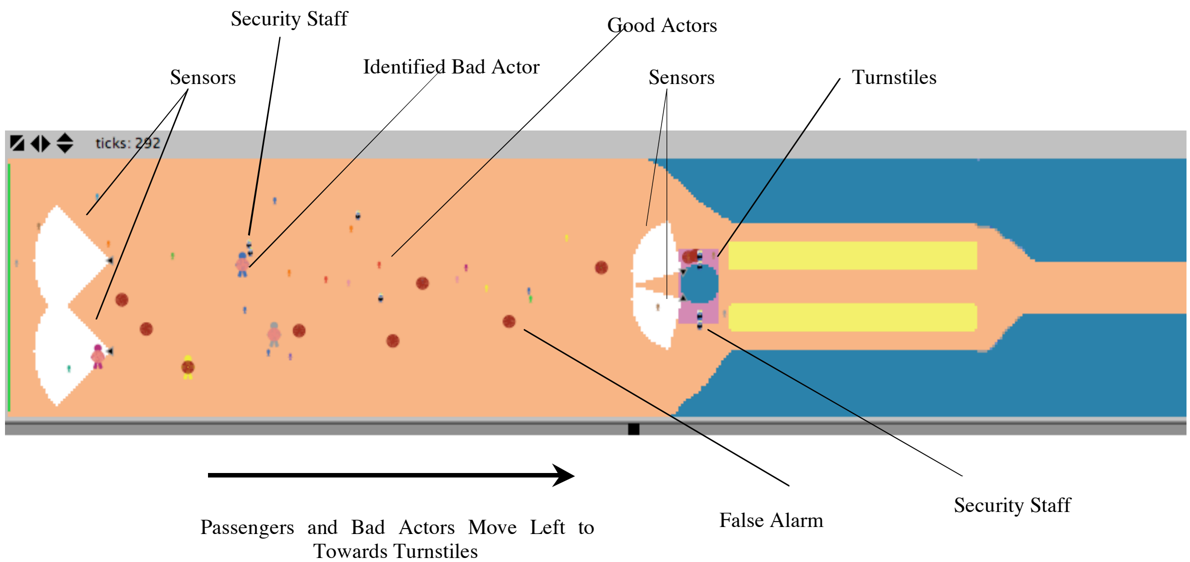


Figure 2 General Simulation Setup

The Agents

There are three types of agents in the model: Security, Civilians, and Sensors. Security agents are generally homogeneous, except that they have different "stations." Some security agents are stationed forward, in the middle of the wide hallway; the other security agents are stationed around the turnstiles. The security agent behaviors are relatively homogenous. If civilians are within range of the security agent and they have a suspicion level that is greater than the security threshold then, if there are more than one other security agents around, they will detain the civilian. All security agents can pursue civilians that fall into that category, save a small number permanently manning the turnstiles.

Civilian agents are instantiated during runtime. The number created each time-step is drawn from a random-exponential distribution. Upon instantiation civilians have a 0.005% chance of becoming either an individual with a gun or a bomb (but not both). If they have a bomb then they also set a trigger for themselves. This trigger is a value based upon the activation energy curve (described *supra*) and if, during the course of the run, their perceptions of the situation yield a value greater than their trigger they will explode their bomb killing themselves and some number of people around them. Data are gathered from civilians when events of interest happen to them. These events include: creation, detainment, first detection and first detection that increased their suspicion level above the security threshold, suicide bombing, and anytime they are removed from the simulation.

The Sensors

Sensors are categorized by one of three types; two types of passive millimeter wave (MMW), and one type of infrared (IR)[1]. The sensors are placed in fixed locations about the environment. The primary purpose of the sensors is the detection and classification of concealed weapons under a person's clothing at this public venue. It is often desirable to screen people from a standoff distance to reduce the chances of long lines or unnecessary crowding. However, the composition, placement, and tasking of semi-autonomous sensor assets, the extraction and utilization of actionable information from these systems, and the generation of reasonable concepts of operations are important problems that have not yet been resolved.

The first stage of this effort involved the development of passive MMW sensor models that can be used to accurately predict the detection and classification probabilities of concealed weapons at range. In the MMW band, objects are described by their apparent temperature–a combination of the actual temperature and the temperature of the reflected background. For a metallic object, the apparent temperature is basically the background temperature. A passive MMW imager uses a radiometer (energy detector) to estimate the apparent temperature in a beam pointed at the object. After accounting for system noise and possible antenna losses, the effective contrast temperature between an object and its background across each beam can be determined as a function of the imager aperture size, focal length, detector time-bandwidth product, weapon size, sky and ground temperatures, and the corresponding weapon emissivities (the amount an object radiates). The detection probability is determined using the effective contrast temperature and the number of beams covering the object at a fixed false alarm probability. A similar procedure is used to determine the classification probabilities for various weapon sizes and shapes with possibly different material compositions.

The simulation uses these detailed high fidelity models to develop performance curves that simulate probability of detection (pd) as well as the probability of false alarms (pfa). When an agent that has a gun or bomb moves through a sensor beam the pd and pfa scaled for distance from the sensor are provided as evidence. The pd and pfa are considered evidence that accumulates to increase the suspicion that a given agent is carrying a bomb or a gun. A random variable modeled possible occlusion from clothes and other travelers; this resulted in situations where a sensor would not return a "hit" even if a passenger had a bomb or a gun.

EXPERIMENTAL DESIGN

A full factorial design of experiments was used for the initial, exploratory analysis reported in this paper. As initial testing confirmed that the performance of this system is highly dependent upon sensor performance and the behavior of the security agents, the experimental design varied parameters associated with those features. Table 1 presents the specific parameters and their associated values. Due to the high degree of randomness in the design, each design point was run 25 times for 5000 time-steps. This gave us a sample size of approximately 70,000 agents per design point, or 3.5×10^8 total agents.

---

[1] Results using IR sensors are not discussed herein.

Table 1: Parameter values varied in the initial experiment.

Parameter Name	Parameter Values
Sensor Degradation	1, 0.75, 0.5
Security Agent Threshold	0.6, 0.7, 0.8
Civilian-pd	0.012, 0.0125, 0.013

Sensor degradation is a parameter that affects the sensor's likelihood of detecting an agent. If the agent is detected then the sensor reports back the appropriate pd to the Bayesian Updater (BU). As described *supra*, the BU creates a "suspicion" score for each agent detected by the sensors. The BU assumes independence of observations to manage the combinatorial complexity that would result from determining a myriad of conditional probabilities, similar to an approach taken in a recent technical report from General Electric (Skatter 2005). The security agent set a suspicion level threshold that sets the trigger when the security agent will attempt to detain a civilian agent. Finally, the Civilian-pd models the probability of detection that is passed to the BU as evidence for determining the suspicion level of a civilian (an agent without a bomb or a gun).

The simulation introduced several modeling constructs. Specifically, sensor degradation was defined as an abstract concept to scale how well the sensors deal with very noisy environments and to specifically model phenomena such as sensor occlusion. Varying the security agent threshold allowed an exploration of the effects of the faith the security agents have in their sensor information. The higher the threshold the higher the necessary information required before a suspicious individual will be detained. Finally, varying the civilian-pd provide a path explore the effects of changes to the sensor fusion algorithm. When civilian-pd values are greater than 0.01 the suspicion level of a civilian will increase. At values of about 0.02 this growth is very slight but non-linear. Therefore, if the civilian chooses a path that leads them through many sensors or causes them to linger in a sensor field they eventually may have a suspicion score that is greater than the security agents' threshold for action. The results of this experiment are presented, *infra*.

SIMULATION RESULTS

Overall, the sensor suite was able to identify roughly 96% of the bombs and guns that were being carried into the venue. There was a larger variance for detecting bombs versus guns, but both performed reasonably well considering that the scalar was dropped to 50%, meaning that in a given time-step there was only a 50% probability that a sensor would register a hit on an agent who had a gun or bomb. Table 2 provides a summary of the simulation results.

As part of the experiment we investigated how increasing the likelihood that "innocent" agents would be incorrectly determined to have a bomb or gun by virtue of being in the sensor's range when an actual threat was detected. For a very small increase (<0.5%), the likelihood of the "innocent" agent being falsely detained rose significantly. This is seen as the probability of false detainment that has a mean of 0.648, but a large standard deviation of .466 and a range of values of 0.991.

Table 2: Result Summary

Variable	Mean	StDev	Minimum	Maximum	Range
P(Detect_Bomb)	0.955	0.023	0.906	0.993	0.086
P(Detect_Gun)	0.968	0.013	0.944	0.993	0.049
P(Miss_Gun_Bomb)	0.040	0.013	0.010	0.068	0.058
P(Correct Detainment)	0.960	0.013	0.932	0.990	0.058
P(False Detainment)	0.648	0.466	0.000	0.991	0.991

Although the simulation was highly stylized, some clear areas of optimality were observed. Consider the probability of correct detainment shown in Figure 3. At low suspicion levels (meaning more likely to detain) and low scalar (meaning less sensor hits) the probability of correct detainment falls significantly. This trend is even more pronounced in the diagram illustrating the probability for correctly detecting a bomb. Interestingly enough, while this area has non-optimal performance for the probability of detecting a bomb, there are other apparent areas of even poorer performance. Similarly, the performance of the probability for missing a gun or a bomb is non-optimal in this area.

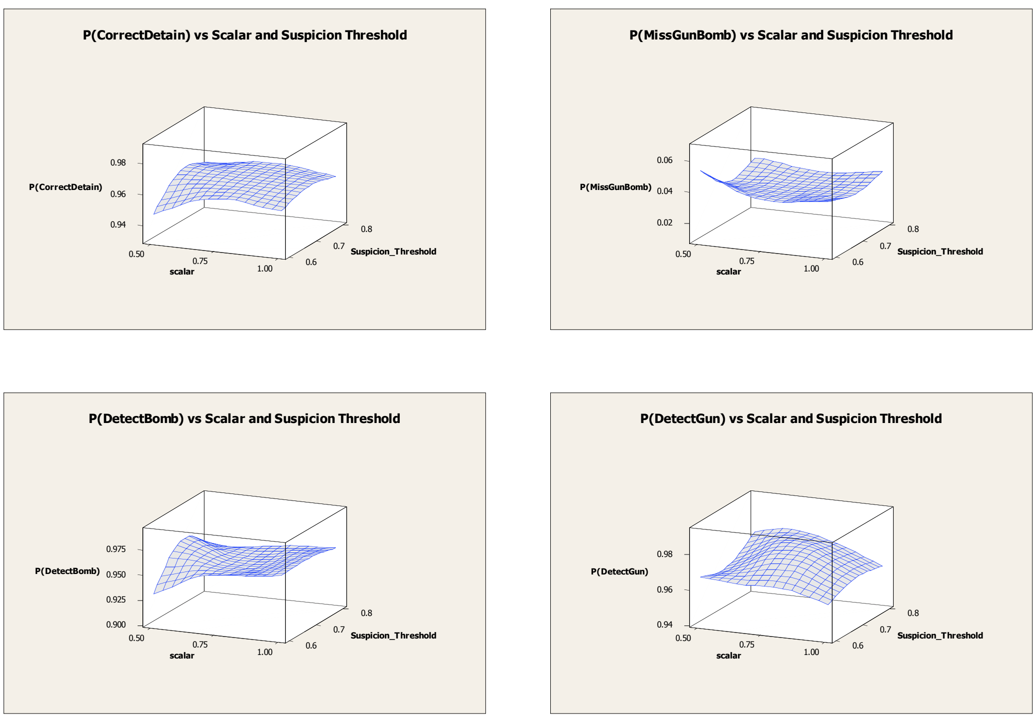


Figure 3 Surface Plots: Prob(event) vs. Suspicion Threshold and Scalar

By way of comparison, Kaplan (2005) offers an idealized assessment of the operational effectiveness of sensors. Specifically, he describes the probability of timely detection given detection of the device on arrival shown as Equation 2 (Idealized Probability of Timely Detection Given Detection on Arrival):

$$P(D_{timely} \mid D_{arrival}) = \Pr\left(L > l\right) = e^{-\left(\frac{\pi}{4}\right)\left(\frac{l}{E(L)}\right)^2} \qquad \text{(Eq. 2)},$$

where L is the distance to the target, l is the distance to detection and E(L) is the mean distance to the target.

In the simulation setup, the distance to detection was 40 meters and the distance to the target was 268 meters. Entering these parameters into Equation 2 yields an idealized timely detection of .983, given that the bomber was detected. This corresponds reasonably well to the summarized results from the simulation, which yielded a probability of .960, indicating that 4 percent of the bombers/gunners were able to either attack or get through the security.

INSIGHTS

The importance of effectively identifying hazardous materials on potential passengers cannot be overestimated. Los Angeles International Airport is anticipating in excess of 78 million passengers per year by 2015, up from 68 million in 2000. Historically, 65 percent of the attacks on airports resulted from portable explosives as well as the majority of the 4280 fatalities resulting from attacks on aircraft (Schell 2003). Cleary, aggressive use of technology and innovative tactics are critical to attempt to mitigate this very real threat.

Our exploratory research results indicate that by using optimistic sensor models and stylized agent behaviors, roughly 96% of the guns or bombs would be detected or interdicted. These estimates are extremely preliminary but compare favorably to the theoretical calculations. However, the simulation does give cause for concern. While individual sensor performance was modeled accurately and to published specifications for MMW and IR radars, perfect tracking was assumed which facilitated the accumulation of evidence. In essence, once an agent was identified as possessing a gun or a bomb a probability was ascribed to the agent and subsequently modified as additional evidence was gathered. In practice in a crowded venue such as a stadium or airport, such perfect tracking is impossible and disambiguating radar hits is non-trivial. The net result is that it is likely that the numbers presented here are optimistic. Consequently it is reasonable to infer that a purely technical solution will likely not provide adequate security.

To address this gap there are a number of tactics that the blue agents can employ to enhance their technical advantage. This is consistent with the current state of the practice that seeks to increase the security capability of airport personnel with techniques such as behavioral pattern recognition (Elliott 2006) which train personnel to spot and understand erratic behavior. Towards this end, additional experimentation is warranted to develop creative blue tactics that will increase the perceived contextual distance of red agents and reduce the likelihood of undesirable action.

REFERENCES

Elliott, R. (2006) , "Assessing Threats from Passengers", in Security Management, Sep. 2006, http://www.securitymanagement.com/article/assessing-threats-passengers.

Johns, M. and Silverman, B. (2001). "How Emotion and Personality Effect the Utility of Alternative Decisions: A Terrorist Target Selection Case Study". In the Proceedings of the 10th Conf. On Computer Generated Forces and Behavioral Representation, SISO, May. 2001:

Kaplan, E.H. and Kress, M. (2005), “Operational Effectiveness of Suicide-Bomber-Detector Schemes: A Best Case Analysis”, Proceedings of the National Academy of Science, Vol.102, No. 29, pgs. 10399-10404.

Koehler, M.T.K., Barry, P.S., Widdowson, B.L., and Forsyth, A.J. (2004), “Case Study: Using Agents to Model Stability and Support Operations”, in the Proceedings of Agents 2004, pp. 667-672.

Izquierdo, L.R., Gotts, N. and Polhill, J.G. (2004) Case-Based Reasoning, Social Dilemmas, and a New Equilibrium Concept Journal of Artificial Societies and Social Simulation vol. 7, no. 3. http://jasss.soc.surrey.ac.uk/7/3/1.html

Schell, T., Chow, B. and Grammich, C., "Designing Airports for Security: An Analysis of Proposed Changes at LAX" (2003). RAND Issue Paper

Skatter, S. , "Detection Systems Fusion Protocol (DFSP). Guidelines for DSFP Compliance" (2005). GE Infrastructure Security technical report document number 961712-1, Feb 2005.

Simon, H. (1957). *Models of Man: Social and Rational; Mathematical Essays on Rational Human Behavior in a Social Setting*, John Wiley and Sons, New York.

Wilensky, U. (1999). NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

Banking, Finance, Business, and Economics

AN AGENT-BASED MODEL FOR CRISIS SIMULATION IN PAYMENT SYSTEMS

L. ARCIERO, Bank of Italy
C. BIANCOTTI*, Bank of Italy
L. D'AURIZIO, Bank of Italy
C. IMPENNA, Bank of Italy

ABSTRACT

This paper presents an agent-based model of a Real Time Gross Settlement (RTGS) payment system. Banks are represented as agents who exchange payment requests, and decide on the timing and the mode of settlement based on a set of simple rules. While highly stylized, the model features the main elements of a real-life system, including a central bank acting as lender of last resort, and a simplified money market. Simulations are run to predict the impact of a disruptive event on the flow of interbank payments. When one of the banks participating in the system is hit by such an event, resulting in the impossibility to perform transactions, three distinct phases emerge. The first one is characterized by inflated liquidity expectations, the second one features a thickening of the money market and of payment queues, and the third one is marked by an increase in defaulted obligations. In order to staunch the flow of losses and restore the orderly functioning of the payment system, central bank intervention must not only be timely, but also relatively intense in terms of the amount of liquidity funneled to the system.

Keywords: Agent-based modeling, Payment systems, RTGS, Liquidity, Crisis simulation

INTRODUCTION

In modern exchange economies, the reliability and the efficiency of payment systems represent fundamental pre-conditions for smooth and safe financial transactions of banks, firms, and households.

The value of payments has increased dramatically in the last decade, as a result of financial liberalization, innovation processes, and increasing globalization of the real economy. In the European Union, interbank payments amounted to 57 times annual GDP in 2005, up from 40 times at the end of the Nineties. Given these developments, in the last decade central banks and market participants have been devoting specific attention to the payments settlement phase, where financial risks are more likely to produce potential systemic impacts. The need to manage and mitigate such risks, at the same time facilitating the handling of an increased volume of transactions, has led to the widespread adoption of Real Time Gross Settlement (RTGS) systems, where individual transactions are settled in real time and with immediate finality.

* Corresponding author address: Claudia Biancotti, Bank of Italy, Department of Economic and Financial Statistics, Via Nazionale 91, 00184 Rome, Italy; e-mail: claudia.biancotti@bancaditalia.it.

Such systems require each participant to hold adequate liquidity levels on an intraday basis; should this not be the case, streams of payment operations might go unfulfilled, triggering undesirable domino effects. A disruptive event, be it physical, technical or financial, may induce prolonged illiquidity conditions, with potentially severe consequences for economic activity. It is therefore very important, especially for central banks as promoters of financial stability, to gain an understanding of how these illiquidity conditions arise, which parts of the system they affect most, and which strategies are most effective when attempting to counteract them.

This paper presents an agent-based model aimed at discerning how a traumatic event affecting a single bank at a given time impacts on liquidity levels and expectations of all other banks participating in the same payment system for the rest of the operational day. While by no means exhaustive in representing the complexity of actual RTGS systems, the model incorporates the core behavioral rules of banks under ordinary conditions; emerging post-crisis behavior appears to be consistent with observations of real-life episodes, if still very simplified.

The paper is structured as follows. Section 2 presents a brief overview of the literature on the simulation of payment systems. Section 3 describes the operating principles of a stylized RTGS system. Section 4 provides the detail of our agent-based model. Section 5 discusses the results. Section 6 concludes and puts forth proposals for future work.

SIMULATION METHODS FOR PAYMENT SYSTEMS

The simulation approach is very suitable for the representation of payment systems: it enables researchers to build models closely replicating the real operational environment, unconstrained by the existence of numerous complex interrelations which are typically hard to represent through traditional econometric tools. Simulations provide information both on the normal functioning of the system and on extreme, not frequently observed scenarios. Input data can be of different kinds: time series of payments submitted by banks can be used for "what if" analyses under different settlement mechanisms, whereas stochastic inputs can be either used for theoretical studies or for models aimed at extrapolating the consequences of particular behavioral assumptions on small-scale settings.

These techniques are now a reliable support in designing payment systems that can control their typical risks, as described by the Bank of England (2000): credit, liquidity, operational, and legal. The Bank of Finland pioneered the construction of simulation models by building an *ad hoc* algorithm (Leinonen, 2005): a deterministic stream of payments is accepted as input and dealt with according to different sets of rules. Bank behavior is taken as given, or is made able to evolve in a predetermined manner.[1]

When considering the largish menu of possible simulation methods currently available, the agent-based framework (Gilbert and Terna, 2000; Fioretti 2004) appears to the best option for our task. Payment systems are coherent in a recognizable way, but their elements, interactions, and dynamics generate structures admitting surprise and novelty which cannot be defined *a priori*. They are more of the sum of their parts; also, the time dimension is explicitly relevant in their functioning, in that different aggregate scenarios may emerge according to the

[1] Payment systems can also be represented as a complex network, with banks as nodes and mutual liabilities/claims defining the arches. This kind of modeling is, however, static: in the sense that the time dimension is not directly taken into account. Network theory has been exploited to study the main features of real interrelations among banks (Boss et al., 2004), making it possible to understand the concentration level of the system, i.e. whether few banks are responsible for the bulk of the links. The consequences of catastrophic events are simply modeled by removing a specific node and measuring the performances of the rest of the network.

payment sequence and rules governing interactions. In other words, they fit the definition of complex systems as provided by Wolfram (1994).

In a way that lends itself well to agent-based modeling, the behavior of commercial banks, at least in the short run, can also be represented in terms of a simple and consistent set of rules, governing a core set of decisions. Given a flow of payments, banks mainly choose in which order these payments have to be submitted in the system, and how to obtain the liquidity necessary to meet obligations, subject to known constraints (Markose et al., 2006). The set of available strategies can be described in the language of game theory (Bech and Garrat, 2003), and translated into algorithms with ease. Learning mechanisms can also be implemented, allowing banks to move from a set of rules to another according to the values of an objective function dynamically updated by simulation results. Predictive learning modules can double as tests of whether banks' adaptive behaviors converge to a steady state in terms, for example, of the liquidity committed in the system (Galbiati and Soramäki, 2007).

SOME RELEVANT FEATURES OF RTGS SYSTEMS

Figure 1 describes the basic functioning of an RTGS environment. In the following, we give a sketch only of those features that are especially relevant for our simulation, and do not appear to have been fully considered in any other similar exercise. For a comprehensive illustration of the underlying system, see Arnold et al. (2006).

Independent of the underlying instrument (bills, checks, electronic transfers, etc.), each payment operation in an RGTS context generates an integrated process, going from the initial decision to transfer funds to a counterparty and until the final settlement in central bank money. In such processes, the main role is played by commercial banks: at some stage, payments between customers of different banks are likely to be treated as interbank flows of liquidity

Several intraday liquidity sources are available to banks. Within the system itself, each bank does normally rely on a continuous flow of incoming payments from its counterparties. Moreover, it can obtain central bank intraday credit, which entails a cost either explicit, if such credit is subject to a fee, or implicit, whenever the provision of funds is not priced but is conditional on the availability of collateral. Alternatively, funds can be borrowed from other banks, in the interbank market.

Since liquidity is costly, banks are however involved in a strategic game, possibly affecting the time pattern chosen to send payments. More specifically, they face, on a continuous time basis, a trade-off between liquidity and delay costs. By releasing payments as soon as funds are available they satisfy customer and counterparty needs and benefit from a sound reputation, but can incur high liquidity costs, to the extent that they borrow from the money market or the central bank. On the other hand, banks can more effectively play on the intraday dynamics of the money market by choosing to delay payments, at the expense of increased sytemic uncertainty and worsening of reputation.

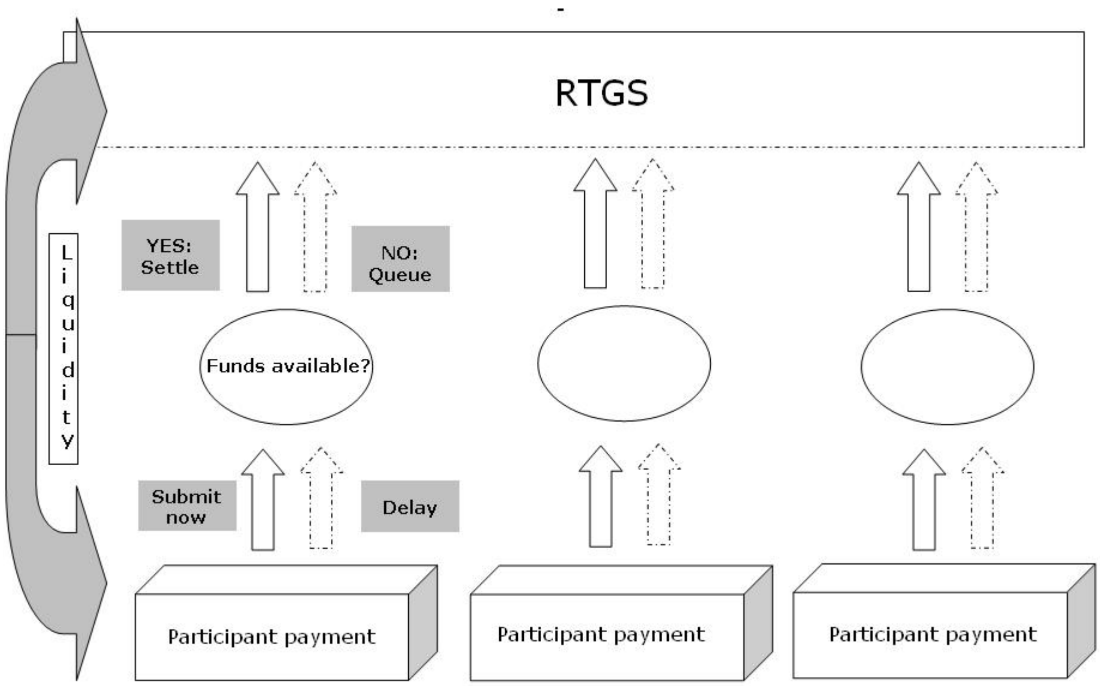


FIGURE 1 Basic functioning of an RTGS environment

DESCRIPTION OF THE MODEL

We model a stylized version of a "plain vanilla" RTGS system, excluding advanced liquidity management tools such as optimization and centralized queues. The model is implemented in the StarLogo[2] environment, and it includes seven breeds of agents: banks, the central bank, payment requests, defaulted operations, interbank loans, crisis events and craters, representing banks hit by such events and accordingly unable to perform any operation. One StarLogo second corresponds to one real-life minute.

During the setup phase, representing the start of the operational day, banks are endowed with a starting level of cash and collateral. During the day, for every tick of the clock, each bank hatches a certain number of new agents, representing aggregations of all payment requests to be delivered to a single counterparty in that moment. For computational reasons, the RTGS system is treated as direct-debit based, with payments always requested by the payee. Payment requests are assigned individual deadlines, ranging from "upon reception" (time-critical payments) to a certain number of minutes after reception. Amounts, deadlines and counterparties are determined through a random draw, whose features can vary depending on the desired scenario.

Each payment request proceeds to cross the StarLogo terrain at fixed speed towards its destination. Upon arrival, it is queued until its deadline expires, and triggers the settlement routine. It is noteworthy to stress that this submission rule allows for the implicit modeling of settlement delay costs banks incur: for each operation, the delay cost is assumed to be a

---

2 Starlogo TNG Preview 4.2, released in April 2007, freely available at http://education.mit.edu/starlogo-tng/.

discontinuous function of time with a jump at the payment deadline, being lower than liquidity costs until the deadline and greater afterward.[3]

From the moment a payment request is generated and until it is settled, the amount thereof is incorporated in the expectations of a liquidity change for both the originating and receiving bank. We therefore assume that in each moment banks are perfectly informed on all payment requests concerning them either as payer or payee; at time t banks are able to calculate their future liquidity up to the moment T, where T-t is the maximum lifespan of a payment request generated at t. The expectation is constantly updated as new payment requests enter the world.

When the settlement process starts, the intended payer tries to meet its obligation with cash. If the cash balance held at the moment is not sufficient, it tries to pledge collateral at the central bank, who provides liquidity based on 100% percentage. If collateral is also insufficient the bank tries to borrow on the money market ("short" bank), thus looking for a lending counterparty ("long" bank). The short bank i randomly draws a potential counterparty k among all the other banks with the exception of j, the bank who originated the payment request p^{ij}_t. Bank k agrees to the loan if the condition $p^{ij}_t \leq E\,[L^k_T]$ is met; $E\,[L^k_T]$ is the expected cash balance for k at time T. In other words, the loan to cover p^{ij}_t is extended on the basis of both present cash balances and future liquidity expectations, as determined by other payments currently in existence initiated by or sent to the potential lender.

Whenever a loan request is refused the payment request bounces, and the short bank looks for another lender. A counter keeps track of the number of bounces per request: when they exceed a certain threshold - a function of the number of possible available counterparties - the bank who has to settle the payment is unable to obtain sufficient funds from anyone. The request is cancelled and flagged as defaulted obligation, whose amount is recorded as an instance of insolvency by the short bank, and as a loss by the intended payee bank. Liquidity expectations for both ends of the transaction are adjusted accordingly.

Disaster is simulated through the introduction of an agent of the "disruptive event" breed. This agent can be called from the StarLogo interface and its job is to pick a bank at random and destroy it, turning it into an agent of the "crater" breed. A crater symbolizes a completely inactive bank, who neither makes or receive payment requests, nor operates in the interbank market (as lender or borrower). In the thirty StarLogo seconds after the disruption has taken place, no agent is aware of it, and all surviving banks continue their routine activity. A random process then makes the banks aware of disaster, with the probability of awareness increasing over time. Once a bank is aware, it stops requesting payments from the bank-turned-crater, no longer considers it as a counterparty for loan requests, transforms all payment and loan requests that are pending toward the crater into defaulted obligations, and updates losses and expected liquidity.

Depending on the simulated scenario, the central bank monitoring module may also be activated. Should this be the case, after a given amount of time after the disruption the central bank starts checking whether banks are delaying their payments beyond some pre-determined physiological threshold. If a bank features an excessive number of delays, the central bank supplies it with an amount of cash proportional to them. The intervention routine stops when the number of delays is back below the threshold.

[3] Though not game-theoretically founded, this rule reflects satisfactorily the fact that banks schedule a large share of outgoing payments according to institutional cut-offs. These cut-offs are agreed with customers, who initiate payments, with the receiving bank, when the payment arise from interbank trades (see e.g. the guidelines of the Euro Banking Federation for money market related payments), or established by the system rules (e.g. for payments related to monetary policy operations).

PARAMETERIZATION AND RESULTS

We parameterize our simulation based on real Summer 2007 data for the Italian RGTS system (BI-REL). More than 100 banks participate in BI-REL directly; for the sake of simplicity, we collapse them into five agents. Each agent is a "superbank", incorporating banks that appear homogeneous in terms of payment traffic, opening balances, and collateral.[4]

Table 1 describes the five agents as they are in the real world. Superbanks 1 to 4 are aggregation of Italian banks, while superbank 5 is the aggregation of Italian branches of foreign banks. We draw simulated endowments and payment traffic from random distributions constructed so as to consider:

1) end-of-day liquidity and collateral, used as a proxy of starting values;

2) payment flows per minute among the five entities, obtained by excising certain categories of payments from the original dataset: intra-agent, cross-border, and payments to and from the central bank, whose role in our model is limited to the one of lender of last resort. The quota of neglected payments is subtracted from the endowments of cash and collateral at the beginning of the simulation. The stream of simulated payment flows sticks to the real distribution pattern, highly skewed, with frequent small payments and rare big ones.

Figure 2 depicts the model predictions. The four panels represent the evolution of liquidity levels, liquidity expectations (expressed as differences between current liquidity and liquidity as estimated after the settlement of all payment flows currently existing in the system), money market thickness, and delays incurred by banks in settlement activity.

Under normal operational conditions, the evolution of both liquidity and expected liquidity predictably resembles a random walk; the specific pattern observed in the sample is almost entirely driven by the few large payment orders circulating in the system, and the starting conditions do not appear to generate any path-dependent evolution. Banks rely on the money market infrequently, and mostly when they need considerable amounts of liquidity; the result is consistent with the real share of interbank loans, estimated at 5-7 per cent of the total intraday payment traffic.

TABLE 1 Main scenario *(€ millions)*

Agent number	Average payments settled daily	Large[a] payments settled (% of the total)	Average end-of-day liquidity	Average end-of-day collateral
1	18,384	3.0	5,302	2,648
2	22,669	6.6	3,583	3,947
3	13,718	3.1	4,780	2,766
4	18,451	8.8	2,211	850
5	43,339	7.5	439	10,119
Total	116,562	5.0	16,315	20,329

[a] Payments are defined "large" if their amount exceeds the 95th percentile of the global distribution of flow values.

[4] The reduction in the number of agents, forced by computational limitations, impacts on the evolution of the complex system. The BI-REL system is, however, quite concentrated; two superbanks, for example, correspond to actual single banking groups, with a high level of internal coordination in the payment system.

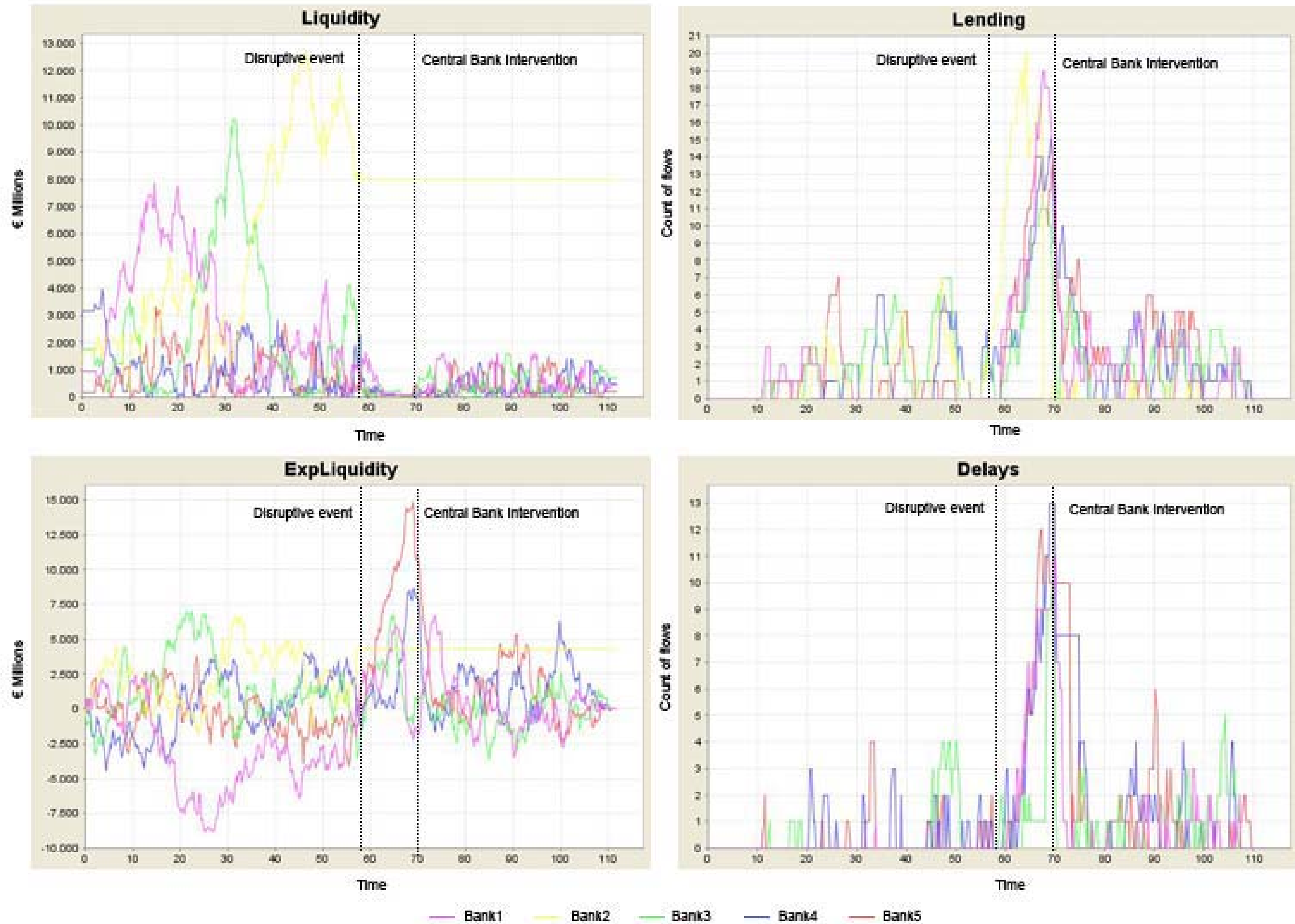


FIGURE 2 Predictions of the model in the minimal intervention scenario

When the disruptive event occurs, and one of the banks is rendered unable to perform any operation, the system evolves through three distinct phases. At first, when no agent is aware of what happened, a spike in expected liquidity emerges. Banks keep on sending payment requests to the bank-turned-crater, incorporating the future settlement of such requests in their expectations; the crater, however, is not able to send out requests of its own, resulting in failure of the regular counterbalancing mechanism for expectations, and illusions of short-run liquidity increases for all its counterparties.

This heralds the second phase, marked by a sizable boom of the money market. Banks experience a lack of liquidity, because they do settle all requests from the crater that were pending at the time of the disaster, but their own requests toward the crater are not settled. After consuming their whole endowment of collateral in exchange for central bank money, they try to counteract the lack of liquidity by turning to the money market. Since loans are granted based on current and expected liquidity both, and expected liquidity is artificially inflated for the reasons stated above, all banks are willing to lend to other banks money they do not yet have: the thickness of the interbank market rises sharply, and as the actual liquidity fails to come in, delays accumulate.

The third phase sets in as banks start to become aware of the disruptive event. One by one, they realize that a bank is not operational anymore, and adjust their liquidity expectations accordingly. Money market activity slows down, and losses are accumulated.[1]

The impact of central bank intervention depends crucially on both timing and intensity. In the scenario depicted in Figure 2 the central bank is relatively slow-moving, and it provides banks with small amounts of extra liquidity: this results in a reduction of delays to physiological levels, but it is not enough to staunch the flow of defaults completely. Runs of the simulation with different behavioral assumptions for the lender of last resort show that the amount of liquidity to funnel so as to neutralize the domino effect entirely can be estimated at somewhere between 1.5 and 2 times the aggregate starting liquidity in the system, depending on circumstances.

CONCLUSIONS AND FURTHER RESEARCH

The model predictions approximate the macro-features of reality adequately, but the framework can be improved along several directions, with the aim to better reflect real RTGS and money market environments. The payment submission process can be refined moving from a direct-debit to a credit-transfer based system, where payments are submitted by the payer. The number of banks should be enlarged and some source of uncertainty could be introduced, by relaxing the independence assumption underlying the payment generation and the common knowledge hypothesis. Moreover, banks should be assigned an end-of-day target in terms of cash balances, to mimic the interday liquidity management optimization they pursue during the maintenance period of required reserves.

As for the central bank, it has to be considered that it autonomously makes and receives payments in real RTGS world, beyond the well-known liquidity supplier function. Modeling of the money market can be refined to take into account the role of overnight interest rates in influencing banks borrowing and lending decisions. Finally, the rest of the world could be

[1] Whether the system goes back to normal functioning, aside from the accumulated stock of defaulted obligations, depends on the relative impact of the disruptive event and its consequences compared with the global amount of liquidity in the system. According to our model, the system would not be able to react autonomously, i.e. without bailouts from the central bank, to any crisis neutralizing one of its major players.

introduced at least as an external shocking agent, also to account for the relevant (and increasing) real-life share of cross-border payment traffic.

REFERENCES

A. Alentorn, S. Markose, S. Millard and J. Yang, 2006, "Designing large value payment systems: An agent-based approach", *mimeo*.

J. Arnold, M. L. Bech, W. E. Beyeler, R. J. Glass, K. Soramäki, 2006, "The Topology of Interbank Payment Flows, Federal Reserve Bank of New York, *staff report* n. 243.

M. Bech, R. Garratt, "The Intraday Liquidity Management Game", 2003, *Journal of Economic Theory*, **109**: 198-219.

M. Boss, H. Elsinger, M. Summer, S. Thurner, 2004, "Network topology of the interbank market", *Quantitative Finance*, **4**: 677-684.

G. Fioretti, 2005, "Financial Fragility in a Basic Agent Based Model", *mimeo*.

M. Galbiati, K. Soramäki, 2007, "A competitive multi-agent model of interbank payment systems", *mimeo*.

N. Gilbert and P. Terna, 2000, "How to Build and Use Agent-Based Models in Social Science", *Mind & Society*, **1**: 57-72.

H. Leinonen (ed.) 2005, *Liquidity, risks and speed in payment and settlement systems – a simulation approach*, Bank of Finland Studies, E:31.

S. Wolfram, 1994, *Cellular Automata and Complexity*, Addison Wesley Publishing Company, Reading, MA.

ADVERSARIAL RISK AND FINANCIAL INSTABILITY: A HYBRID MODEL

M. BRAGEN, Argonne National Laboratory
D.L. SALLACH, Argonne National Laboratory
P. THIMMPURAM, Argonne National Laboratory
H. RICH, Argonne National Laboratory
J.F. BURKE, Argonne National Laboratory

ABSTRACT

Because of its inherent complexity, terrorist attacks that attempt to disrupt the international financial system are difficult to model. The present project assembles three interaction layers — global, national, and regional, each with its distinct dynamics — in order to explore the types of risks the financial sector faces. The model that connects the three layers is a hybrid, with the first two mechanisms being in the form of systems dynamics mechanisms, and the third being a fine-grain agent simulation. Currently, a variety of scenarios are used to exercise the model, but it is planned that these will be replaced with a scenario generator that can perform sensitivity analysis as well.

Keywords: financial system, systemic risk, adversarial risk, capital flight, liquidity crisis

INTRODUCTION

Financial instability can be caused by endogenous or exogenous factors, or a combination of the two (Johansen and Sornette 2002). Exogenous factors can be highly disruptive and further exacerbated by endogenous weaknesses (Horwich 2000). A particular kind of exogenous threat to financial stability is posed by substate actors (van Creveld 1991). Unlike conventional exogenous dangers, the terrorist threat is intentional and, thus, designed to exacerbate and amplify natural weaknesses of the market.

Days before the most recent U.S. presidential election, for example, Osama bin Laden (2004) released a speech to *Al Jazeera* in which he took satisfaction in putatively causing a million dollars of economic disruption for each dollar that Al Qaeda spent. He indicated that he and his co-conspirators will continue a policy of “bleeding America to the point of bankruptcy.”

Economic and financial attacks can be used to complement and/or intensify more comprehensive terrorist attack goals. An attack on financial institutions might have multiple effects within a coordinated attack, including:

- Intensification of a complementary physical attack
- Undermining of confidence in financial and/or government institutions
- Prevention of the provision of liquidity or other financial resources
- Penetration of banking institutions to conduct proscribed transactions
- Generation of revenue for terrorist networks
- Disruption of global financial stability

However, attempting to model the consequences of an attack or a series of attacks is a vast, unwieldy process.

The financial system is global, dynamic, and immersed in the much larger economic system. There is extensive data on financial transactions, but most pertains to endogenous economic and financial concerns that might mask the effects of an adversarial perturbation. Complexities notwithstanding, a critical part of mitigation requires recognizing vulnerabilities and assessing the potential mitigation effects of various possible policy alternatives.

A model designed for these purposes will necessarily be partial. It must be notional in a way that seeks to decenter endogenous economic and financial interactions, while focusing upon perturbative effects and the conditions which heighten them. Such notionality will need to be multilevel, recognizing that financial processes interleave global, national, and local aspects. Finally, the threat itself must be modeled in order to trace its consequences.

The present paper reports on a project that undertakes to address these objectives. An integrated model is presented that accepts a particular attack scenario and explores how the effects of the disruption spread through national, international, and regional financial and economic institutions.

THE FINANCIAL DOMAIN

The global financial system can be conceived as having three interacting levels. First is the *national* system, which includes exchanges and their regulation, payment and clearance system infrastructures, and a central bank and its policy capabilities.

There are multiple national systems, so their markets, payment systems, central banks, etc., influence each other internationally; thus, the second level is global. One of its major processes — the one addressed in the present model — is the flow of international capital. Investments of various types move from one currency to another, from one market to another, etc. Of course, the flow of investments influences the financial well-being of the impacted economies. The secure communications infrastructure has both national and international components, and some online markets (e.g., NASDAQ) can best be regarded as global as well.

The third level concerns firms and, especially, their operations that initiate and respond to orders and payments. These actors utilize financial infrastructures and are impacted by international capital flows. Their decisions, in turn, greatly impact national and global liquidity. Taken together, these three levels provide a complex target for terrorists, a platform for subtle policy considerations to the Federal Reserve Board, and a complex working environment for all financial participants.

Exchanges and the Economy

Stock markets impact and, in some ways, represent the larger economy with its diverse and intertwined industries. From the standpoint of disruption, there are two major categories of events: (1) major market shifts, most of which are entirely endogenous, and (2) material and operational disruptions.

The infrastructure component represents flows of transactions through the payment, execution, clearance, and settlement phases. Trading activities are processed only during normal (user-specified) operating hours on normal operating days subject to the exchange being available. The availability of the exchange can be limited by infrastructure and workforce availability as well as automatic shutdown due to market conditions.

Global Capital Flows

The capital flow mechanism is drawn from Tirole's model (2002) of instability in emerging economies and is global in nature. Its focus is the tendency for capital to flee during disruptive crises. This pattern can be observed as arising in endogenous financial dynamics and has the potential to be exacerbated during adversarial attacks. More particularly, a massive and sustained withdraw of capital is potentially a source of deep economic disruption and, accordingly, one of the fervent goals of terrorist movements. Figure 1 summarizes the structure of the model.

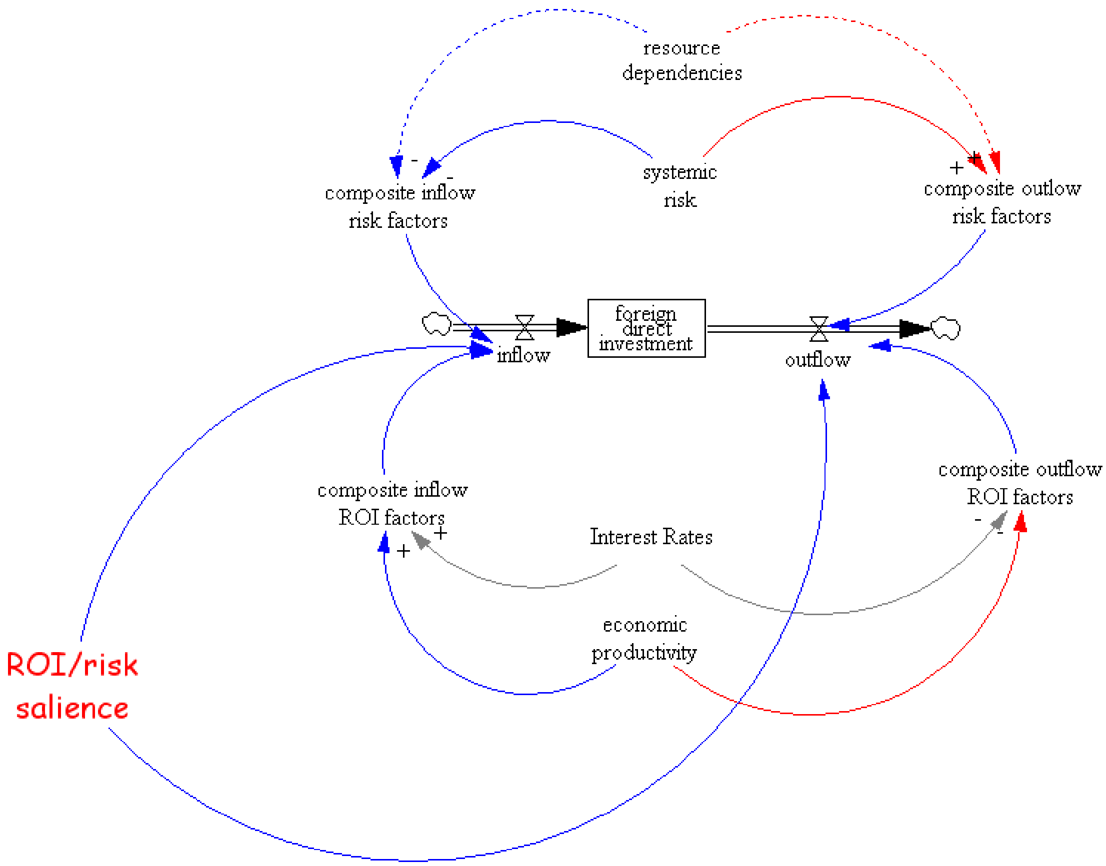


FIGURE 1 The structure of capital flows

During normal economic periods, the importance of return on investment (i.e., interest rates and economic productivity) causes the lower loops to dominate. However, in a crisis period, risk becomes more salient, and the upper loop dominates the flow of capital.

Transaction Practices

In addition to robustness issues related to physical infrastructure and operations, there are also robustness issues related to firm responses to disruption. As an example of the latter, after an adversarial attack on economically sensitive targets and/or international financial infrastructures, systemic risk can be exacerbated, albeit inadvertently, by a reluctance of firms to resume payments until the flow of payments owed them has resumed. Because there is a densely connected network of financial obligations, each delayed response, measured in hours, has the potential to create, and then intensify, a liquidity crisis, correlatively deepening the danger of national and global systemic risk.

The robustness issues inherent in payment practices can best be captured by using a fine grain, agent-based model with the potential to clarify the effects of the range of responses of diverse firms to multiple interacting risks. The model differentiates representation of firms as distributed by industry, region, and size on the basis of empirical data from the U.S. Census Bureau (see Figure 2 for an illustration).

	A	B	C	D	F	G	H	I	J
5	Geographic	2002 NAIC	Meaning of 2002 NAICS code	Meaning o	Year	Number of esta	Sls, shps,	Annual pa	Number of e
6	Alabama	211	Oil and gas extraction	Total	2002	45	1,393,849	47,556	792
7	Alabama	212	Mining, except oil and gas	Total	2002	154	1,068,472	257,667	5,549
8	Alabama	213	Support activities for mining	Total	2002	83	152,739	41,568	1,167
9	Alabama	221	Utilities	Total	2002	503	Q	948,747	16,014
10	Alabama	236	Construction of buildings	Total	2002	2,862	7,409,912	866,960	25,486
11	Alabama	237	Heavy and civil engineering const	Total	2002	835	2,963,658	655,496	17,128
12	Alabama	238	Specialty trade contractors	Total	2002	5,648	5,208,721	1,470,274	55,941
13	Alabama	311	Food mfg	Total	2002	299	7,150,635	875,050	36,481
14	Alabama	312	Beverage & tobacco product mfg	Total	2002	25	D	D	g
15	Alabama	313	Textile mills	Total	2002	90	2,765,167	407,110	15,628
16	Alabama	314	Textile product mills	Total	2002	118	1,389,825	152,636	6,292
115	Alaska	211	Oil and gas extraction	Total	2002	18	6,997,919	254,766	2,746
116	Alaska	212	Mining, except oil and gas	Total	2002	50	524,586	92,176	1,392
117	Alaska	213	Support activities for mining	Total	2002	63	731,621	307,635	6,162
118	Alaska	221	Utilities	Total	2002	89	Q	113,593	1,721
119	Alaska	236	Construction of buildings	Total	2002	785	1,682,501	240,198	5,399
120	Alaska	237	Heavy and civil engineering const	Total	2002	295	1,375,925	331,317	6,195
121	Alaska	238	Specialty trade contractors	Total	2002	1,279	1,358,943	366,942	9,763
122	Alaska	311	Food mfg	Total	2002	148	1,338,970	198,941	7,365
123	Alaska	321	Wood product mfg	Total	2002	33	27,756	5,150	176
124	Alaska	323	Printing & related support activiti	Total	2002	57	D	D	e
125	Alaska	324	Petroleum & coal products mfg	Total	2002	17	1,906,102	41,947	632
126	Alaska	325	Chemical mfg	Total	2002	6	D	D	e
127	Alaska	326	Plastics & rubber products mfg	Total	2002	13	28,472	5,032	121
128	Alaska	327	Nonmetallic mineral product mfg	Total	2002	30	D	D	e
129	Alaska	332	Fabricated metal product mfg	Total	2002	51	69,318	17,637	417
130	Alaska	336	Transportation equipment mfg	Total	2002	16	D	D	c

IBQDownload070306B

Ready

FIGURE 2 Illustrative firm data from the U.S. Census Bureau (Source: U.S. Census Bureau undated)

Mechanism Integration

Taken together, the three financial layers represent different scales of interaction and variegated types of risk. The capital flow mechanism is global in scope and places national issues in the context of international investment decisions. Notwithstanding its global interaction (see Figure 3), capital flow is the simplest of the three layers.

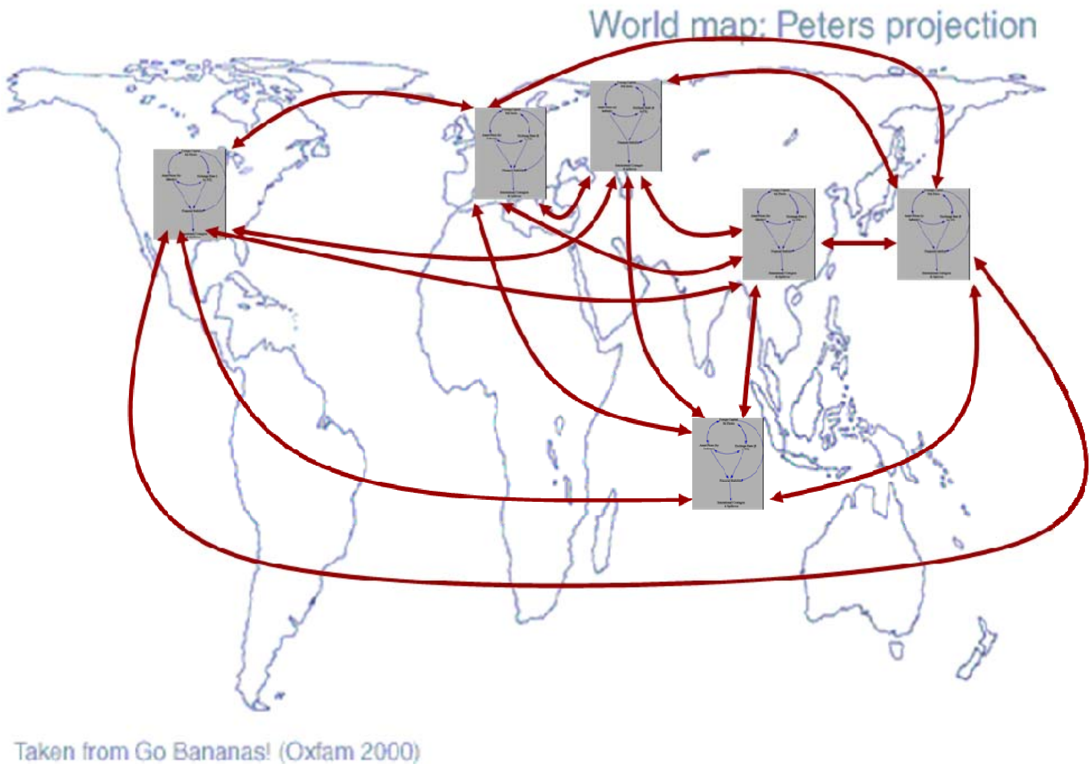


FIGURE 3 Global capital interactions

The exchanges and economy mechanism is national in scope and more complex in focus. The exchanges are mostly physical and operational in form and thus can be the direct target of attack with resulting disruption. In periods of crisis, exchanges have the protective mechanisms of circuit breakers and margin calls, which are explicitly modeled. There are payment and clearing infrastructures that can be a target of attack and thus can be disrupted, causing further downstream effects. Finally, the economy incorporates diverse industries with varied geographical distributions and effects.

The third mechanism, representing payment resumption, is the most local and detailed of the three. It represents firm-level decisions in the face of unanticipated and disruptive circumstances. To do so effectively, the payment resumption mechanism takes into account the factors to which decision-makers give weight, some of which are summarized in the preceding section. Two alternative decision models are available, and others can be incorporated as needed.

Scenario Generation

During the development of the present model, three scenarios were discussed, and one was developed in depth. The purpose was not only to shape the model mechanisms described above but also to lay the groundwork for the development of a more general scenario generator. The scenario generator will specify the nature and physical consequences of one or more possible attacks and the propensities of firms to return to their standard business practices, as mediated by industry, region, and firm size.

Notwithstanding the fact that warfare is presently largely asymmetric in form, a scenario generator is inescapably military in nature. It concerns the preparation and conduct of attacks designed to achieve maximum disruption of the financial system of the United States and, more broadly, the global economic system. Such attacks may be coordinated in complex ways and of extensive duration. Therefore, it is necessary to construct a scenario generation system that can probe the dynamics of the three financial layers and identify vulnerabilities that may cause such attacks to have a more disruptive impact than might otherwise be expected.

MODELING FINANCIAL INTERACTIONS

Market Index

The Market Index as used in the model provides a diagnostic assessment for the rest of the model. While we do not explicitly model the immediate affects of a terrorist attack on the market index (the user supplies scenario data regarding the initial impact), we do model market changes as the initial impact ripples through the rest of the model (Figure 4). A stochastic stream of daily market fluctuations based on historical data from 1975–2005 is used to prime the system and provide a “normal” operating environment for the model. As stated, the user must provide an estimated market index adjustment profile along with the scenario data that describes the attacks.

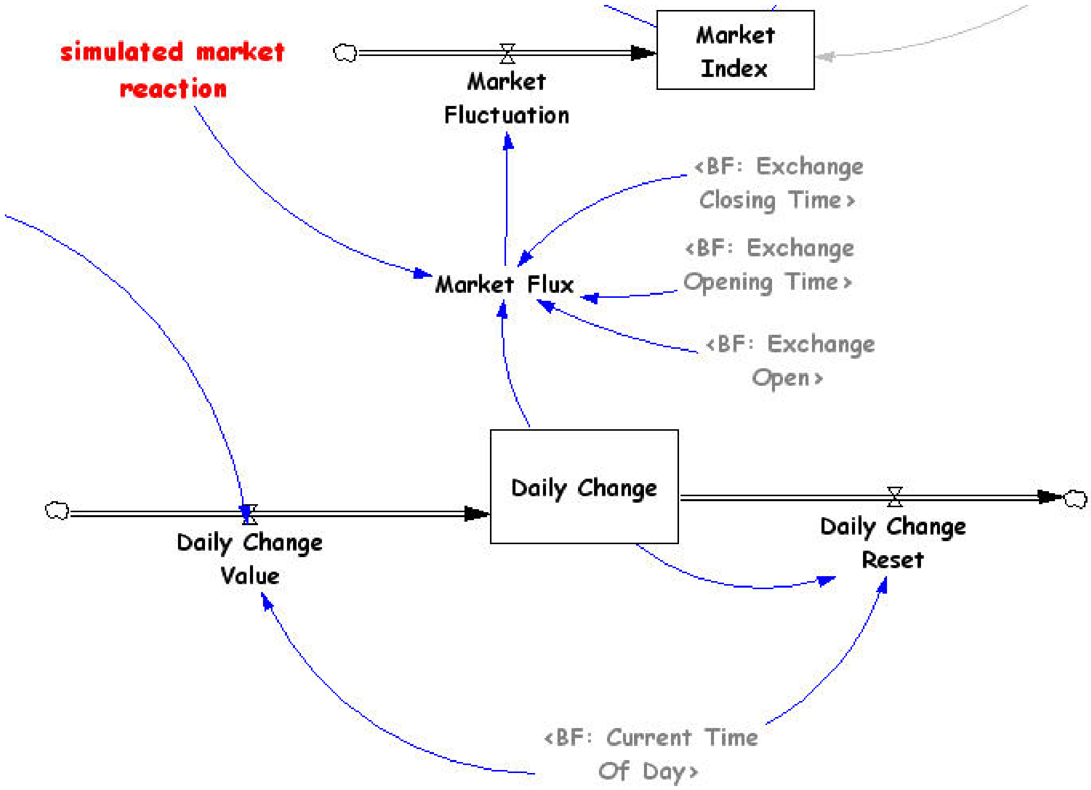


FIGURE 4 Monitoring the market index

Exchanges

The exchanges section of the model keeps track of the dollar volume flowing through the exchanges. The open hours of the exchanges are those of the New York Stock Exchange (NYSE): Monday through Friday, 9:00 A.M. to 4:00 P.M. Eastern time. During closed hours, no volume flows occur. The operability of the exchanges is also dependant on a number of other factors: workforce and infrastructure availability, secure communications availability, and automatic shutdown criteria (circuit breakers) defined by NYSE (each of which is described in detail in a later section). The total time required to complete a transaction is four days on average, with one day each utilized for execution and clearance and two days utilized for settlement. These dollar volumes "change state" as they flow through the system (Figure 5).

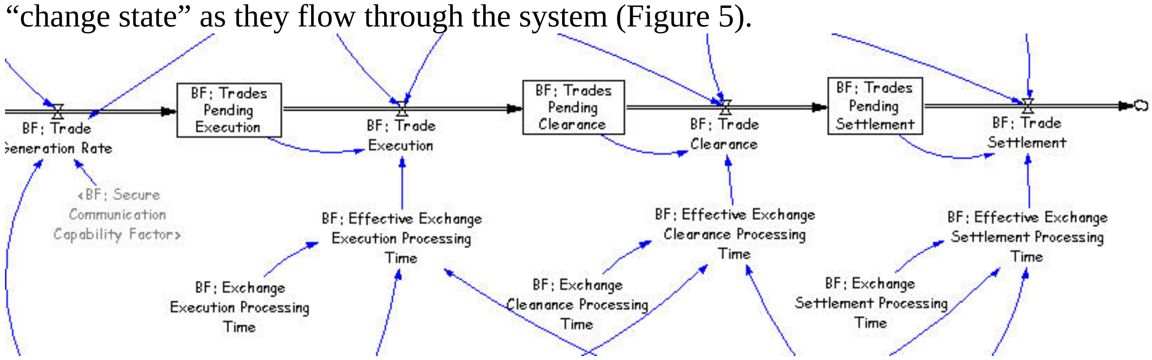


FIGURE 5 Monitoring the dollar volume flow through the Exchanges

Workforce Availability

We want to be able to model attacks against people as well as against infrastructure (e.g., as in the anthrax release). The operation of the exchange is dependant on the availability of the workforce. If the workforce is reduced in number for any reason, there can be a decrease in the capability of the remaining workforce. We define a capacity factor that ranges from 0 through 1 where 0 indicates no capacity and 1 indicates full capacity. This capacity is then used to adjust the processing times of the various stages of the exchanges. Currently, the user specifies this workforce capacity factor profile as input. However, another model could easily be integrated that would compute this capacity on the basis of scenario parameters. Note that the effects on workforce capacity are not limited to attacks: that is, the model could be used to analyze the impact of a pandemic flu outbreak. There are currently workforce capability factors for the exchanges, depository institutions, and payments systems. The factors for depository institutions and payment systems are structured identically to factor for exchanges.

Infrastructure Availability

As with the workforce, infrastructure must be available for the processing of the transactions through the exchanges. An infrastructure capacity factor similar to the workforce capacity factor is defined in the range of 0 through 1. This capacity factor is then applied to the processing times of the various stages of the exchanges. This capability factor can be used to model a variety of situations from actual infrastructure damage, to infrastructure contamination, to lack of required resources from outside sources (e.g., electrical power). Currently, the user specifies the infrastructure capability profile as input. However, it would be easy to integrate additional repair or decontamination models that would define this profile. There are currently infrastructure capability factors for the exchanges, depository institutions, and payments systems. The factors for depository institutions and payment systems are structured identically as for exchanges.

Secure Communications

We model FEDWIRE and other communications in the Secure Communications segment of the model. There are a number of steps required for completing a transaction. We explicitly model execution, clearance, and settlement, each of which contains multiple information and data flows between participants (Figure 6). Since we are modeling at a high level of aggregation in this segment of the model, we do not track the low-level detailed communications. We also only consider the communications generated by the exchanges and payment systems. The transactions cannot complete until all communications have completed. We created a Secure Communications Capability Factor with the range of values of 0 through 1 that defines the capability of the communications system and is used to adjust the data transfer times accordingly. Secure communications are available 24 hours a day, seven days a week.

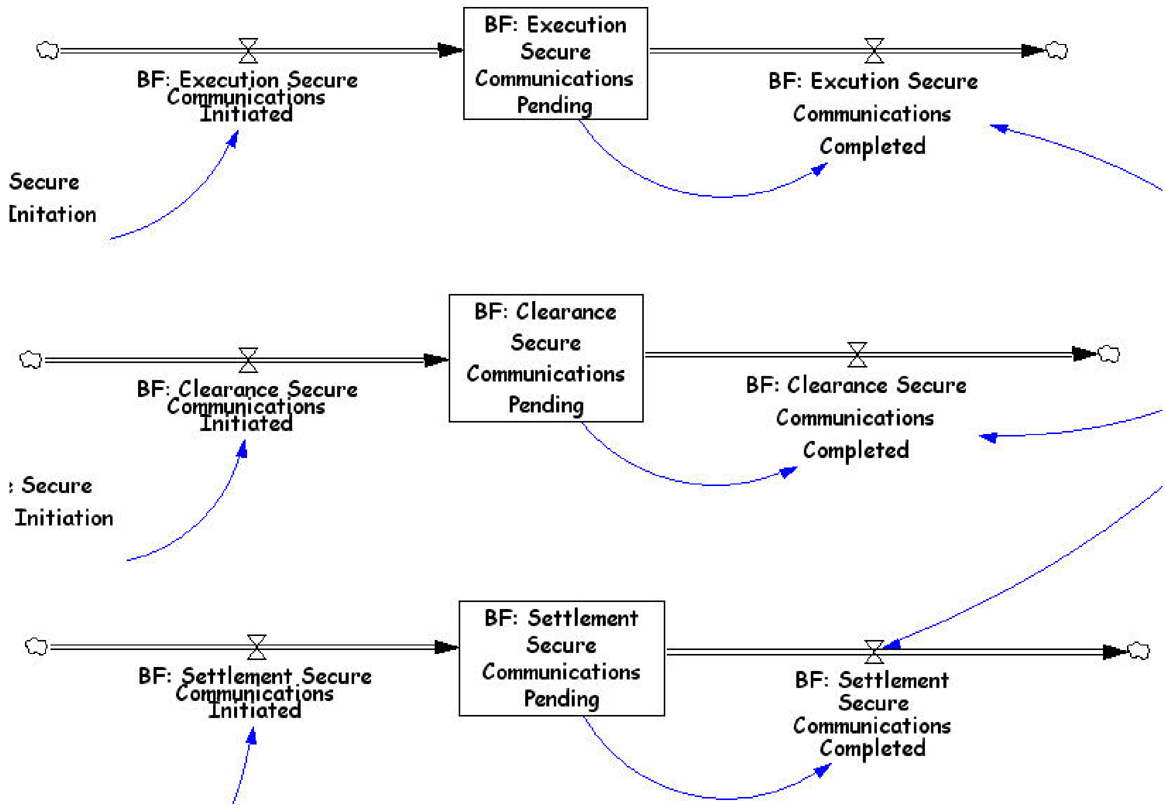


FIGURE 6 The state of secure communications awaiting completion (Note: BF=Banking and Finance)

Circuit Breakers

In response to dramatic drops in the market in October of 1987 and 1988, the NYSE instituted, and the U.S. Securities and Exchange Commission (SEC) approved, a set of circuit breakers to reduce market volatility and promote investor confidence (NYSE Euronext 2007). These circuit breakers are explicitly modeled and are tied to drops in the Dow Jones Industrial Average (DJIA). They are summarized in Table 1.

TABLE 1 NYSE circuit breaker policies

Event (measured from the start of the trading day)	**Time of Day** (Eastern)	**Halt Trading?**
Ten percent drop in the DJIA		
	Prior to 2:00 p.m.	For one hour
	2:00 to 2:30 p.m.	For 30 minutes
	After 2:30 p.m.	No halt
Twenty percent drop in the DJIA		
	Before 1:00 p.m.	For two hours
	1:00 to 2:00 p.m.	For one hour
	After 2:00 p.m.	Close exchange for the day
Thirty percent drop in the DJIA		Close exchange for the day

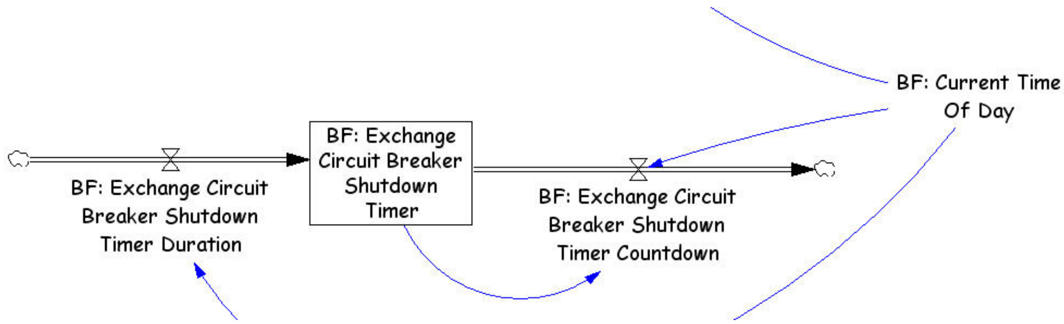


FIGURE 7 Circuit breaker shutdown timer

Margin Calls

Investors can purchase securities on margin using some personal cash along with cash borrowed from the broker. The investor intends that the value of the securities increase sufficiently so that the loan from the broker can be paid and a profit is realized. To protect the broker, the investor must keep cash or other securities in a margin account with the broker. The value of this account must be kept at or above a minimum requirement. If the value falls below this minimum requirement, a margin call is issued and the investor must provide additional cash or securities. The investor can accomplish this by providing additional cash or by selling securities. In the event the investor does neither, the broker himself can sell securities owned by the investor.

Since such forced sales have the potential to shift prices in the market, as well as have a negative effect on investor confidence, we explicitly model these margin calls (at a high level of aggregation). We use a mechanism similar to that used for circuit breakers with an additional component. While circuit breakers are triggered only during extreme market conditions, margin calls occur on a daily basis regardless of market conditions. So we add stochastic margin call transactions to the system based on historical data. We then monitor the changes in the market index and amplify the margin calls as the market index drops beyond the user-specified limits (Figure 8).

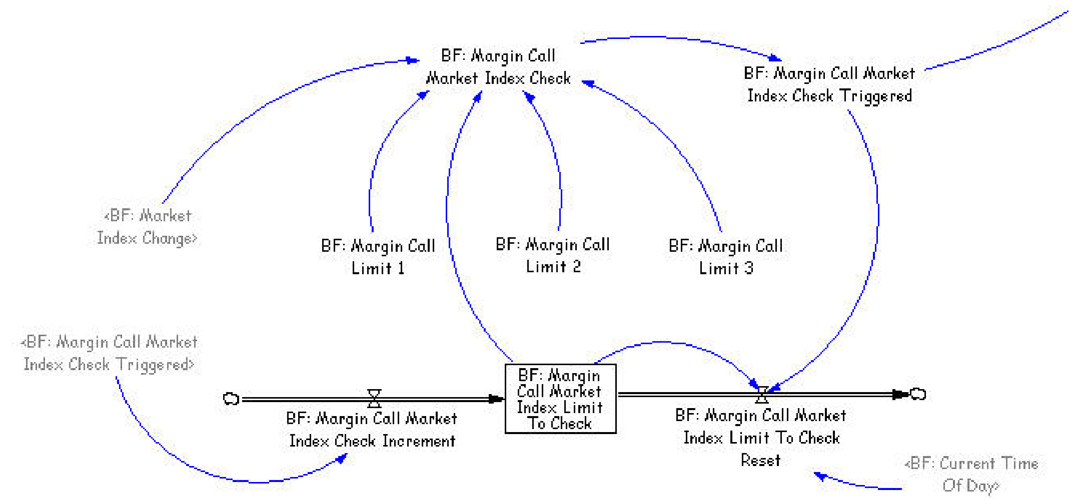


FIGURE 8 Monitoring market fluctuation with respect to margin calls

Payment System

The model explicitly models the Payment System (PS) of the U.S. economy. The dollar volume of payments that requires clearance and settlement (such as the issuance of a check) are tracked by the model. Three sources of these payments are available within the model: exchange payments, foreign indirect investment payments, and sector payments (payments by individual firms). The Payment System is dependant on four separate capability factors: PS Infrastructure Capability, PS Workforce Capability, Depository Institution Infrastructure Capability, and Depository Institution Workforce Capability. The integration of the System Dynamics model with the Agent models of Transactions and Cash Pinch occurs in the Payment System segment of the model (Figure 9). The agent models aggregate their data and provide it in a System Dynamics-compatible form.

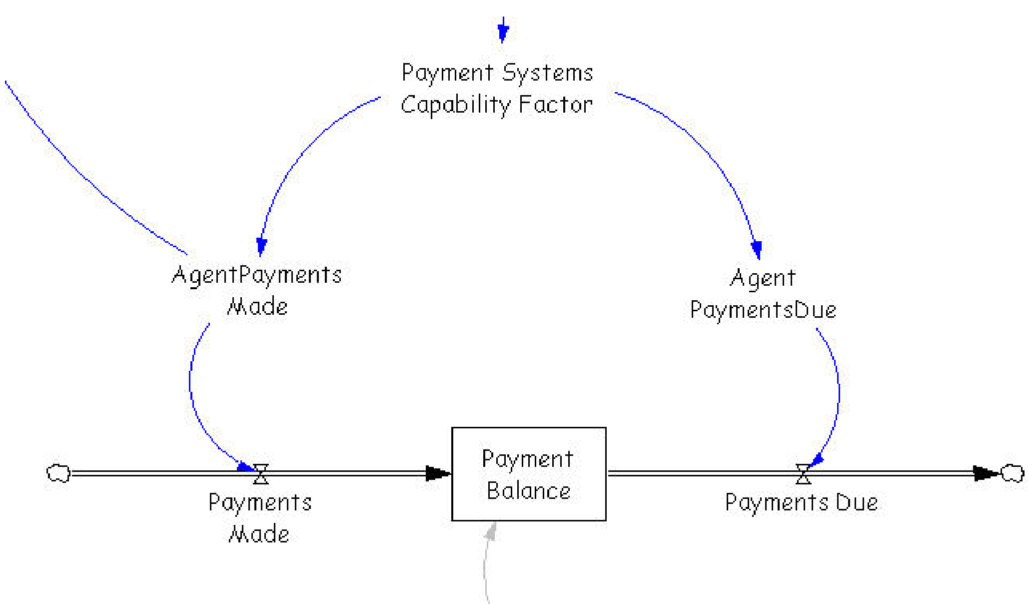


FIGURE 9 Payment systems linkage to agent models

International Cash Flow

The International Cash Flow segment of the model analyzes the indirect effects of a terrorist attack on the United States through the direct effects on Economic Productivity, Interest Rates, and Systemic Risk. The model considers both Return on Investment (ROI) Risk and prospective undermining of the Safe Haven assumption of foreign investors reacting to investment opportunities in the United States. When the opportunities are favorable, foreign investment tends to increase; when unfavorable, they tend to decrease (Figure 10).

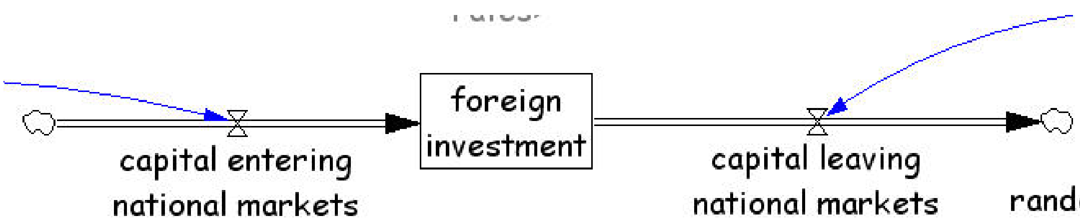


FIGURE 10 Monitoring foreign investment

Two agent models have been integrated with the System Dynamics. Each models the payments made and received by individual firms. They implement two different firm-level philosophies of payment resumption after a terrorist attack.

Transactions

The Transactions segment of the model generates the hourly payables from and receivables to the firms that are of interest within a geographic region, industry, and size for the scenario under consideration. The annual payments, receivables, and the number of firms are collected from the U.S. Census Bureau's online database. On the basis of this data, individual Firms are created that belong to a specific industry (i.e., are assigned a four-digit North American Industry Classification System [NAICS]) and geographic region (i.e., assigned a U.S. State) and that have expected annual payments and receivables using a Pareto distribution. The expected annual payments are distributed into expected daily payments and then into expected hourly distributions using Gamma distributions. Each hourly payment is categorized into mandatory, necessary, and contingent portions. The actual transactions (payments and receivables) can be modeled either by Field Effects or Cash Pinch mechanisms. The two mechanisms are described below.

Field Effects

The propensity of the Firms to pay any outstanding dues is modeled using the Field Effects. The field effects have components that are generic, regional, and industry-specific. The user inputs the field effects for the adverse conditions (e.g., terrorist attacks) and for the recovery period. The actual hourly payments are a function of expected hourly payments and field effects. Any payments that are not paid in the current hour are accumulated into a backlog and scheduled into the next month's expected payments.

Cash Pinch

The Cash Pinch agent model uses a Cash-On-Hand perspective to determine which, if any, payments will be made. Each individual agent (firm) starts off with a specific amount of cash in its possession along with a schedule of expected payments and receivables. The schedule is per month and is repeated for each month in the simulation.

All spending is classified as either discretionary or nondiscretionary. As the model executes, the firm compares the cash on hand with its payments and receivables for the day and determines whether there are sufficient funds to pay all bills. When there is a cash shortfall, the firm will determine which, if any, of its bills will be paid on time. Discretionary spending is first curtailed. Any discretionary payments in arrears are considered nondiscretionary at the point at which the new payable date is assigned. The goal is to remain solvent through the time frame. The model makes the simplified assumption that business makes a profit. During times of anticipated financial problems, the firms will divert available cash to run the business.

CONCLUSION

The banking and financial sector provides services to the American economy and increasingly to an integrated global economy. Accordingly, it is vast, dynamic, and interwoven in complicated, evolving ways. No model can do full justice to its complexity.

At the same time, the banking and financial infrastructure is a target for terrorism, both directly and as a collateral consequence of attacks on other primary targets. A successful attack on the financial infrastructure is likely to have ripple effects throughout the country and the world. Thus, for the sake of protection and mitigation, it is imperative that we model the financial infrastructure, including its vulnerabilities, interactions, and the threats it faces.

The present project assembles three interaction layers — global, national, and regional — with each having distinct dynamics in order to explore the types of risks the financial sector faces. The model that connects the three layers is a hybrid, with the first two mechanisms being systems dynamics, and the third being an agent simulation. Currently, a variety of scenarios is used to exercise the model, and a scenario generator that will perform sensitivity analysis is in the near-term plan.

ACKNOWLEDGMENTS

This work is supported by the U.S. Department of Energy, Office of Science, under contract number DE-AC02-06CH11357.

REFERENCES

Bin Laden, Osama, 2004, "Transcript of Osama Bin Laden's Speech," Worldpress.org.

Horwich, George, 2000, "Economic lessons of the Kobe earthquake," *Economic Development and Cultural Change* 48:521–542.

Johansen, Anders, and Dideriet Sornette, 2002, "Endogenous versus exogenous crashes in financial markets" in *SSRN*: Social Science Research Network.

NYSE Euronext, 2007; available at www.nyse.com/press/circuit_breakers.html.

Tirole, Jean, 2002, *Financial Crisis, Liquidity, and the International System*, Princeton, NJ: Princeton University Press.

U.S. Census Bureau, undated, available at http://factfinder.census.gov.

van Creveld, Martin, 1991, *The Transformation of War*, New York: Free Press.

MODELING THE TRANSITION TO HYDROGEN-BASED TRANSPORTATION

M.R. MAHALIK,* Argonne National Laboratory, Argonne, IL
G. CONZELMANN, Argonne National Laboratory, Argonne, IL
C.H. STEPHAN, Argonne National Laboratory, Argonne, IL
M.M. MINTZ, Argonne National Laboratory, Argonne, IL
T.D. VESELKA, Argonne National Laboratory, Argonne, IL
G.S. TOLLEY, RCF Economic and Financial Consulting, Inc., Chicago, IL
D.W. JONES, RCF Economic and Financial Consulting, Inc., Chicago, IL

ABSTRACT

At the Agent 2004 conference, Stephan and Sullivan reported on an agent-based model running under Repast J that describes how a personal transportation system might transition from petroleum to hydrogen. In this paper, we extend the original model in several ways. First, in place of a rectilinear grid of roads, we use the road topology of a real metropolitan area, the Los Angeles basin. Second, while the earlier model added hydrogen-dispensing stations on the basis of a rather simple algorithm, in the present model, station agents attempt to plan their investments on the basis of imperfect knowledge of driver fueling behavior, the expected penetration of hydrogen, the plans of competitors, and the amount of fuel they estimate will be dispensed. In addition, the driver agents are more complex. Their incomes vary, and they live and work in areas corresponding to that income on the basis of demographic data. They have attributes such as "greenness" that affect their willingness to pay more to operate a vehicle that has desirable environmental characteristics. We take advantage of Repast Simphony's networking capabilities to set up relationships between the agents on the basis of their home neighborhoods, their places of work, and common characteristics. The agents are influenced in their purchase decisions not only by their personal experience and a general "belief space" reflecting the attitude of society in general toward hydrogen but also by interactions with their peers. We examine how a transition to hydrogen succeeds or fails as these various attributes are varied and given greater or lesser influence.

Keywords: multi-agent modeling, hydrogen transition analysis, infrastructure investment modeling, social networks

INTRODUCTION

Hydrogen holds great promise as an automotive fuel. It emits no greenhouse gases either when burned in an internal combustion engine or when used to power a fuel cell, and ideally it can be made in a way that releases no net CO_2 to the atmosphere. Fuel-cell-powered vehicles can achieve energy efficiencies two or more times those of their internal combustion engine counterparts. Nevertheless, hydrogen faces formidable obstacles in replacing petroleum as the fuel of choice for automobiles. While many technical hurdles remain to be overcome, another

* *Corresponding author address*: Matt R. Mahalik, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439; email mahalik@anl.gov.

equally perplexing problem is how to induce both consumers to buy hydrogen-powered vehicles and investors to build refueling infrastructure when each depends upon the other for viability and neither exists today. This conundrum is often referred to as the "chicken and egg" problem and is especially severe for hydrogen because of the large costs involved, although the challenges are by no means confined to this problem.

At this point, it appears impractical and expensive to equip passenger vehicles with two separate powertrains, one for hydrogen and one for gasoline.[1] Thus, a transition from petroleum to hydrogen will likely have to succeed on its own through reinforcing feedbacks. We can envision a few strategically located hydrogen fueling stations (HFSs) that would induce an early adopter group to purchase hydrogen-powered vehicles (HPVs), and that those increasing numbers of vehicles would encourage more investment in infrastructure, and so forth. But how could this sustained transition be achieved, particularly in the early stages? Where should early HFSs be located, and how might investments be scheduled? What may be the range of business models and strategies employed by potential investors in the underlying infrastructure? What types of drivers, if any, should be targeted as the first customers for HPVs, and what inducements would be most effective? How does the early adopter group interact with the remaining majority of consumers, and how does that interaction affect the transition? Standard econometric models are at a disadvantage in answering these questions because a successful transition will likely depend upon a diversity of players. We use multi-agent modeling and simulation to address some of these key questions.

MULTI-AGENT HYDROGEN TRANSITION MODEL

Agent-based modeling (ABM) can be especially helpful in analyzing complex problems such as hydrogen transitioning that involve a diversity of players. Such modeling uses many "agents" which are heterogeneous autonomous actors and decision-makers. Agents can be given various and differing characteristics, and they interact with one another according to rules specified in a computer simulation. At the Agent 2004 conference and elsewhere, we reported on a Repast-based ABM of the transition from petroleum to hydrogen developed at Ford Motor Company (Stephan and Sullivan 2004a and 2004b, Stephan 2005), and similar work has been carried out by others (Schwoon 2007). This initial model gave interesting results, showing different transition behaviors depending, for example, upon what relative weights drivers put on worry and inconvenience. However, it had a number of serious limitations. For example, it did not model a "real" metropolitan area; the investor agents' criterion for investment, depending only upon weighted traffic counts, may have been overly simplistic; it did not take used car markets into account; and, finally, the driver agents, while capable of being influenced in their buying decisions by a global "belief space," did not interact individually with one another.

Under a U.S. Department of Energy (DOE)-sponsored project, a team drawn from Argonne National Laboratory, RCF Economic and Financial Consulting, Inc., Ford Motor Co., and other organizations is building a new analysis tool and continually expanding and enhancing it. The results reported here are from supplementary experiments that focus only on specific parts of the core model. In particular, the paper concentrates on the sensitivity of hydrogen vehicle

---

[1] Other schemes have been tried. On-board fuel reformers capable of producing hydrogen from gasoline were attempted in the 1990s but abandoned as impractical. BMW is currently testing a "dual-fuel" hydrogen/gasoline vehicle with a conventional internal combustion engine that can be switched to run on either fuel.

market penetration to differences in preferences and learning behaviors among drivers of hydrogen automobiles under simplified assumptions about infrastructure investors. The driver utility function considered here is a modified version of the one used in the core model, so that the experiments reported here should be viewed with these limitations in mind.

Agent Environment and Agents

In the present simulation, we choose as our model region a 100-by-50-mile rectangular area centered on the Los Angeles, California metropolitan area and divided into 5,000 square cells (Figure 1). In addition, there is a 25-mile-wide buffer zone surrounding this region in which agents do not live but into which they may travel. The roads on which agents travel include actual interstate expressways and "ubiquitous" local roads passing through every cell except where there are natural barriers. Driver agents are given a number of characteristics: income, "greenness," degree of concern about running out of fuel, and buyer-type "personalities." High-, middle-, and low-income driver agents are randomly located predominantly, but not exclusively, in neighborhoods of the same type on the basis of demographic data and are distributed such that the overall agent population density scales to the real population density (Figure 2). Similarly, agents are randomly but preferentially assigned to "jobs" in locations chosen to be relatively near their homes and the average salary levels of which preferentially match their incomes.

FIGURE 1 Los Angeles, California metropolitan area showing modeling grid structure (not including "buffer zone") and major expressways as laid out in the grid format

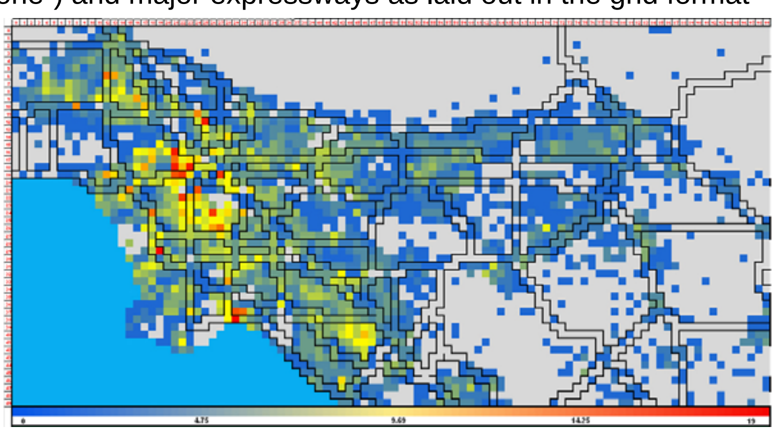


FIGURE 2 Population densities (households per square mile) in the model area

In addition to the driver agents, there are investor agents who build hydrogen refueling stations at strategic locations where anticipated fuel sales will be sufficient to make a profit. The results presented here are based on an investment decision strategy that is more simplistic, that is, investors place or remove HFSs on the basis of suitably weighted traffic counts and sales as opposed to exogenously supplied thresholds. We are currently testing more sophisticated investment algorithms. Our more advanced agents make their investment decisions with imperfect knowledge of driver vehicle purchase and fueling behaviors, the expected penetration of hydrogen, the plans of competitors, and the total demand for fuel. Given the uncertainties they face, the investor agents try to do the best they can, possibly make non-optimal decisions, and learn from their experience. Our advanced investor agents work with limited knowledge about what has happened in the past (such as fuel sales in particular locations) and have some crude assumptions on what might happen in the future. With this limited knowledge, they develop demand expectations for use in making their investment decisions. As the driver agents respond to the roll-out of the supply infrastructure by purchasing HPVs and hydrogen fuel, the investor agents receive feedback on the actual demand realization. They use this information in an ex-post analysis to revise their expectations and adjust their future investment plans by using a simple form of Bayesian learning. Rather than having single-value expectations, our investor agents have subjective probability density functions of sales. The investors use rules of thumb to determine whether and where to locate fueling stations. Similar to our driver agents, investor agents have a utility function they try to maximize. This approach allows us to distinguish between different types of investor agents with different attitudes toward risk, ranging from risk-prone to risk-averse.

Agent Decision Rules

For brevity, we will concentrate our discussion of agent decision rules in this section on the model's driver agents. In the course of a simulation, driver agents drive to and from work and to various destinations in the model region and the buffer zone. Drivers note the presence or absence of HFSs (whether or not they actually own an HPV). They accumulate either real (when driving an HPV) or potential (when driving a conventional vehicle [CV]) inconvenience and worry. On the basis of the increasing prevalence of vehicle-based global positioning system (GPS) units, we expect that drivers will have access to real-time data on the locations of HFSs and automated route planners that will show in advance whether it is possible to make a trip using an HPV and where to stop for fuel. Thus, drivers need not be concerned about the distance to the next HFS but nevertheless will suffer some worry if the distance between two HFSs takes their fuel below an agent-dependent "comfort level" for refueling. Drivers suffer inconvenience if there is no HFS in either their home or their work cells and also if they have to make a special trip to refuel before starting on a planned trip. Finally, agents are inconvenienced if their refueling habits must be altered as a result of owning an HPV. Serious inconvenience is suffered if the agent cannot make a desired trip with his or her HPV because of a lack of HFSs en route. An agent suffers inconvenience to a lesser extent if he/she must make a special trip to refuel or must refuel before he/she would otherwise want to in order to be able to reach a more distant HFS.

When it comes time to purchase a car, the agent weighs the pluses and minuses he/she sees of owning hydrogen versus conventional technology.[2] All factors are cast in terms of

---

[2] In order to engender the maximum number of purchases from a limited number of agents, our driver agents own fleets of 1,000 "millicars" (millicar = 1/1000 car) and buy/sell a specific number of these millicars every quarter

present-value dollars and summed up in the driver agent's personal "utility function." These factors include the difference in capital cost of the two types of vehicle as well as the difference in operating cost per mile. The latter term is converted to a present-value amount on the basis of the number of miles the agent expects to drive over the time period in which he wants to recoup any additional capital investment. The driver assigns dollar values to intangible factors. For example, he places a cost on the inconvenience of having to make a special hydrogen refueling trip based in part on the distance traveled. The agent sums up all such trips he made (or would have had to make if he had had a hydrogen car) over his driving experience, but weights recent experience more heavily. (Clearly, it is a rare driver who performs this math in the real world, but most drivers have some experience-based intuitive understanding of the inconvenience they can expect to suffer as a result of a scarcity of HFSs. It is this intuitive weighing of many different factors that we seek to simulate with a utility function.)

An important new feature is the ability of the driver agents to interact with one another and influence each other's buying decisions. Drivers are assigned personality types characterizing their buying behavior, as shown in Figure 3. An Early Adopter, for example, considers it a plus to be one of the first to buy a new technology; a Fast Follower also likes new technology but wants others to try it out first, etc. Thus, any agent's purchase of an HPV influences other agents' buying decisions either positively or negatively. Agents can interact with one another through four spheres of influence. The first is a global influence. Agents drive around and see others driving HPVs, see HPVs on television, and hear pundits expounding upon them. As more and more drivers switch to hydrogen, many personality types are persuaded to buy such vehicles themselves. Many, but not all, types want the new and uncommon: for example, if everyone has adopted a particular technology, "Techno-freaks" feel it is time to move on to something new.

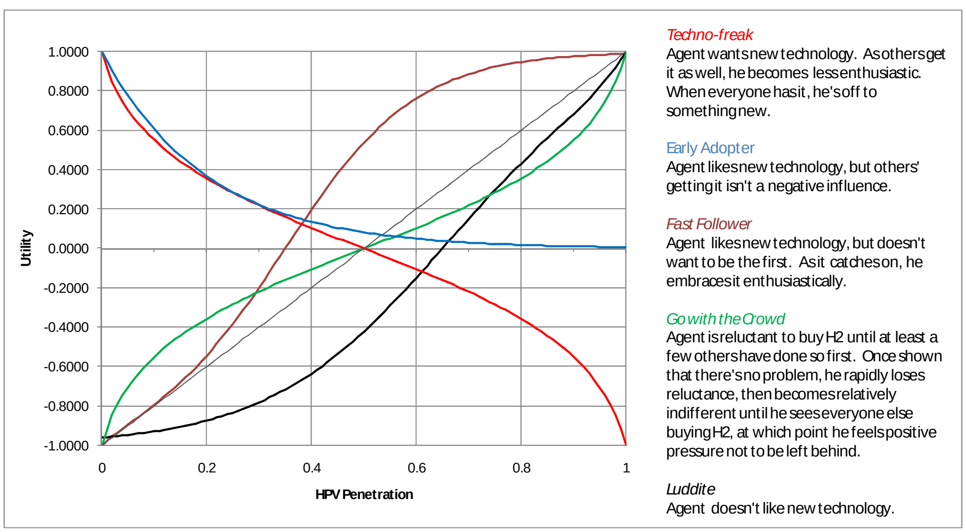


FIGURE 3 Driver agent "buyer personality types"

A second sphere of influence is the neighborhood. Talking with neighbors, seeing HPVs parked in front of their houses, and passing them on neighborhood streets likewise influences

---

on a schedule such that 1,000 millicars are replaced over a time period corresponding to the time that the agent would own a single car.

prospective buyers. In our model, the radius of such influence can be set exogenously. The third sphere is work. A driver talks with colleagues or sees them driving HPVs and is influenced by their choices. The fourth sphere is through social networks, where an agent is connected with friends or family not necessarily living in his neighborhood or working at his place of work. Some agents will be "opinion makers," having extensive networking connections and having high influence on others. This last sphere, however, is still being implemented and is not reflected in our current results. All these parameters are summarized in the agent's utility function as follows:

$$Utility_n(t) = d \cdot Utility_n(t-1) + (1-d) \cdot \left\{ \Delta CapCost - \sum_i w1_{n,i} \cdot InconvD1_{n,i} + \sum_s w2_{n,s} \cdot DrPers_n(fH_s) \right.$$

$$\left. + \; PVMultiplier \cdot \left[DistDriven_n \cdot \Delta OpCost - \sum_i w3_{n,i} \cdot InconvDD_{n,i} - w4_n \cdot Worry_n \right] \right\} \qquad (1)$$

In this equation, $Utility_n(t)$ is the utility of an HPV for agent n at timestep t. It is normalized to range from -1 (strong desire for a CV) to +1 (strong desire for an HPV) by a normalization factor omitted for clarity. It is based on the agent's previous hydrogen utility multiplied by an exponential time-decay factor d plus his utility during the current timestep multiplied by $(1 – d)$. In this way, older experience is discounted more heavily. $\Delta CapCost$ is the difference in capital cost between a CV and an HPV, and $InconvDI_{n,i}$ represents the various types of distance-independent inconvenience agent n has experienced. $DrPers_n$ is a function based on Figure 3 giving the influence agent n receives when a fraction, fH_s, of the agents in influence sphere s have HPVs. The terms in square brackets are distance-dependent parameters: $DistDriven_n$ is the distance agent n has driven in the last timestep; $\Delta OpCost$ is the difference in operating cost per mile between a CV and an HPV; $InconvDD_n$ are the various types of distance-dependent inconvenience the agent has encountered in his driving; and *Worry* is the worry he has experienced. The weights that the agent puts on the intangibles inconvenience, influence, and worry are represented by the terms *w1*-*w4*. (While the w_n's can vary from agent to agent, all results shown in this paper are based on common values for all agents). Finally, all the distance-dependent terms are multiplied by a factor *PVMultiplier* to convert the experience over the quarter to a present value on the basis of a given discount rate and the agent's anticipated driving over his desired payback time period.

As mentioned, agents have different levels of income. Income designation determines not only where they live and work (and consequently who their peer groups are) but also fixes the schedule on which they buy cars and whether those cars are purchased new or used. In the current model implementation, we assign agents to one of three income groups: high (20%), middle (60%), or low (20%). High-income drivers buy only new cars and then scrap or resell them as used cars on a regular schedule (where the age distribution has a median car age of ~4 years); middle-income agents also buy only new cars, but keep them until they are scrapped (median age ~9 years); low-income agents buy only used cars (those sold by high-income agents) and keep them until they are scrapped (median age ~14 years). The buy/sell/scrap distribution schedules were adjusted to match U.S. data for overall car ownership and scrappage

(Davis and Diegel 2007).[3] Annual driving distances for older vehicle are discounted to reflect the fact that older cars are driven less (*ibid.*). In the results that follow, we typically simulate about 7,000 driver agents (representing about 0.1% of the driving population of the area), although we have made runs with as many as 70,000 agents.

MODEL RESULTS

The model results presented here are based on agent-assigned characteristics that are not necessarily meant to be realistic but rather are held relatively simple to show how the model works. We choose values for model parameters such that a significant percentage of our agents switch over to HPVs during the 20-year simulation.

Base Case – No Social Interaction (i.e., No Influence)

Figure 4 shows the penetration of HPVs into the existing fleet of CVs over a 20-year period. At the end, 64% of the vehicles on the road are hydrogen-powered. However, this figure varies dramatically by driver income level. While 81% of the vehicles owned by high-income agents are HPVs, they account for only 45% of low-income agents' vehicles. This result emerges from the fact that in our model, low-income drivers buy only used cars and buy them only from high-income drivers. Thus, for a period of some years before high-income drivers have first purchased and then sold HPVs, low-income drivers are restricted by the used car market to CVs only. As HPVs enter the used car market, they buy them whether they want the technology or not, since they are restricted in their choice by market availability.

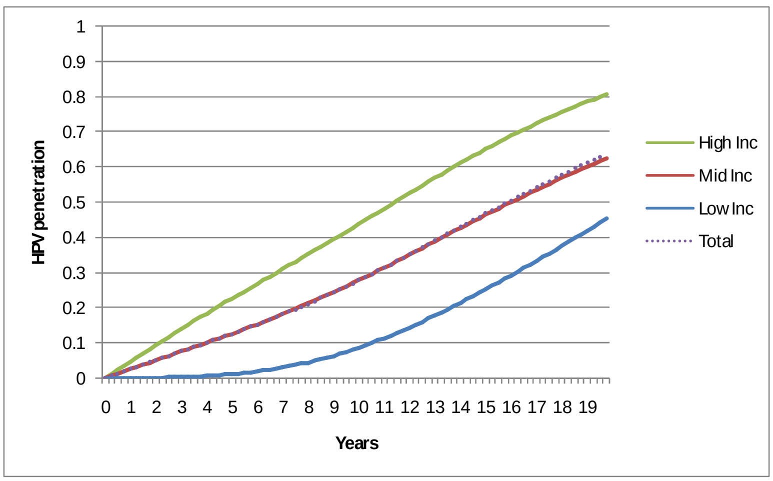


FIGURE 4 Fraction of HPVs owned by income group – agents have no influence on one another

[3] The schedules used are a compromise between a close approximation to reality and programming complexity. A possible future enhancement is to create a used car market in which all agents can participate and where prices are market-determined.

While this is an extreme case to a certain extent, it does reflect a real-world fact that used car buyers have to take whatever is available. When we run the model without this constraint, low-income buyers begin to buy HPVs as soon as they become available in their vintage, and in fact at the end of the 20-year period, they exceed the middle-income agents in percentage of HPVs owned. This (perhaps) counterintuitive result is explained by the fact (again, in our model) that low-income agents buy cars more frequently (and scrap them more frequently) than do middle-income buyers.

Early Adopters and Luddites

We now assign personalities to our agents and allow them to interact with one another. For purposes of illustration only, we deem all high-income agents to be Early-Adopters (hereafter referred to as HI-EAs), all middle-income agents to be Fast Followers (MI-FFs), and all low-income agents to be Luddites (LI-LUs), as illustrated in Figure 3. First we turn on Global Influence. That is, our agents our influenced by the fraction of HPVs they see on the road, regardless of where they see them or who owns them. We cut off the “population of influence” at an age of five years, so that agents ignore the technology of cars older than this age. The weight of this influence is deliberately chosen to be high. For MI-FFs and LI-LUs it ranges from -$2,000 to +$2,000 as the fraction of HPVs on the road increases from 0 to 1. That is, when only a small fraction of cars are HPVs, these agents regard an HPV as being worth $4,000 less to them than when everyone owns an HPV. For HI-EAs, the range is from +$2,000 to zero.

Figure 5 shows the results for this case. Because there are very few HPVs on the road at the start, the MI-FFs have no one to follow, so they stick with existing technology. HI-EAs start out strongly buying HPVs, but the total HPV vehicle fraction grows significantly slower than before, and this eventually pulls down their adoption rate. Finally the MI-FFs begin to adopt more rapidly as the HPV fraction rises above a critical threshold. The LI-LUs are prevented from buying HPVs early on because of their unavailability, then forced to buy them at the end when CVs are not available. When this constraint is removed, the adoption curve for this group becomes smoother.

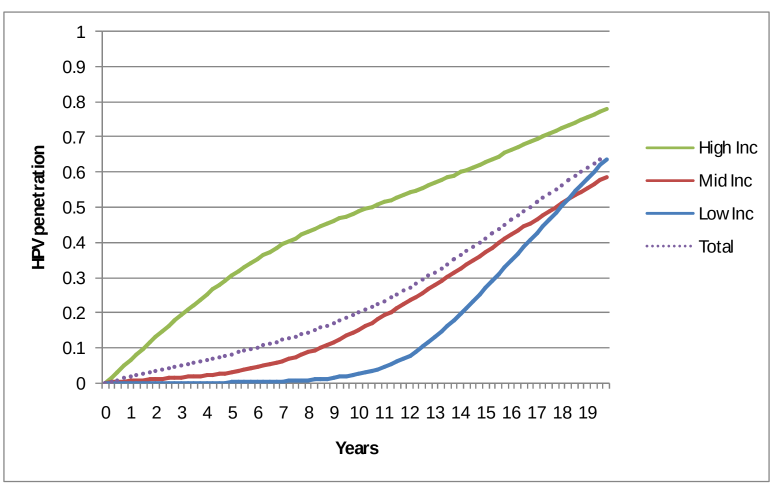


FIGURE 5 Fraction of HPVs owned by income group – agents exert global influence on one another

Neighborhood and Work Influence

In our final illustration of this simulation sequence, we remove Global Influence and substitute Neighborhood Influence and Work Influence. Figure 6 illustrates the Neighborhood Influence case, since the Work Influence results were very similar. Compared to the No Influence (Figure 4) and even the Global Influence (Figure 5) cases, the adoption of HPVs by MI-FF buyers is very slow, while that of HI-EA buyers has increased slightly back to the "no influence" level. Why should this be the case? Recall that, while neighborhoods are not completely income-segregated, middle-income agents, which constitute the majority of the population, are much more likely to see others like themselves as they look around their neighborhoods rather than the high-income agents needed to kick-start the MI-FFs' adoption. Consequently, adoption among middle-income agents never gets off the ground. Low-income agents are even less likely to see high-income agents in their neighborhoods, and consequently, when they are not forced to buy HPVs, these Luddites eschew the new technology almost completely.

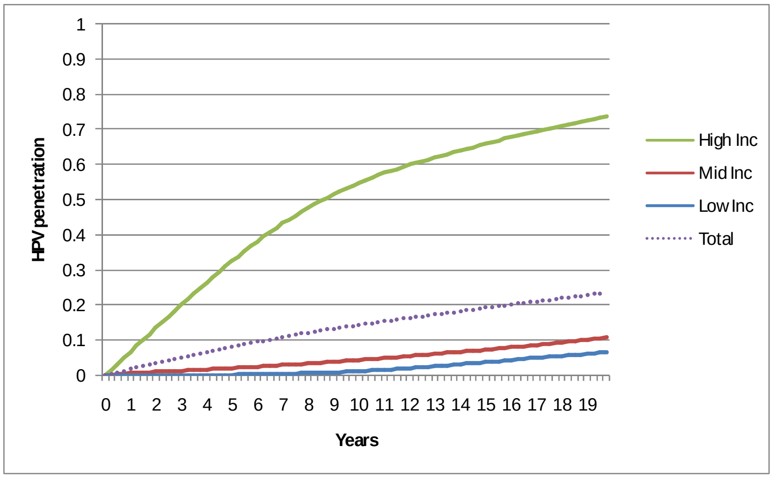


FIGURE 6 Fraction of HPVs owned by income group – agents exert neighborhood influence on one another

CONCLUSIONS

The examples illustrated above have given us confidence in the model design. As more sophistication is built into the model and driver and investor agents are given more diverse and realistic characteristics, we believe the model will allow us to gain insights into how the complex problem of interacting agents evolves in both realistic and extreme circumstances.

The purpose of the full model for the overall project is to study the development of hydrogen infrastructure. Infrastructure suppliers and purchasers of hydrogen vehicles react to one another's behavior in the chicken-or-egg interactions of the early transition to a hydrogen economy. Drivers will not purchase hydrogen vehicles unless hydrogen fuel is available conveniently, and the incentive to supply fuel depends on the existence of hydrogen vehicles to use the fuel, all occurring within a spatial context. The full model will allow the investor agent supplying infrastructure to decide whether to supply hydrogen from distributed production at fuel stations or from centralized facilities with pipeline or truck delivery to the stations. The investor agents make their decisions on the basis of profitability, but with allowance for risk aversion in their utility functions. They are “satisficers” (imperfect maximizers) who may make mistakes in their expectations about the future. A key feature is that they learn from their mistakes by using a Bayesian approach as events unfold. The mistakes drive the system off course, while learning acts as a corrective mechanism. The full model will contain multiple investor agents who compete with one another. The full model also explores whether government interventions are needed to help speed adoption in response to energy policy goals.

As noted earlier, the experiments reported here focus on the “driver side” of the model, using simplified assumptions about investors. The utility function of driver agents in the full model differs from that used in the experiments reported here. The driver behavior in the experiments here differs by using a weighted average of current and past utilities to change behavior over time. While the experiments reported here are intriguing, it is to be emphasized that they do not reflect any conclusions from the overall project.

ACKNOWLEDGMENT

This work is supported by the U.S. Department of Energy, Office of Science, under contract number DE-AC02-06CH11357.

REFERENCES

Davis, S.C., and S.W. Diegel, 2007, *Transportation Energy Data Book*, Ed. 26, ORNL-6978.

Schwoon, M., 2007, “A Tool to Optimize the Initial Distribution of Hydrogen Filling Stations,” *Transportation Research Part D*, 7082.

Stephan, C., 2005, “Modeling the Growth of a Hydrogen Transportation Infrastructure,” *Frontiers in Transportation*, Amsterdam, July 2–6.

Stephan, C., and J. Sullivan, 2004a, “An Agent-based Hydrogen Vehicle/Infrastructure Model,” *Conference on Evolutionary Computation*, Portland, June 19–23.

Stephan, C., and J. Sullivan, 2004b, “Growth of a Hydrogen Transportation Infrastructure,” *Agent 2004*, Chicago, October 7–9.

Saturday, November 17, 2007

Social Simulation Applications

Parallel Track II

Health Care and Epidemics

A SIMULATOR FOR CONTINUOUS AGENT-BASED MODELING

J. DUGGAN*, National University of Ireland, Galway. Ireland.

ABSTRACT

This paper describes a simulation environment that can be used to integrate population-level dynamics with those occurring at an individual, or agent-based, level. The benefit of this approach is that individual agent behaviour may be mapped at a detailed level, using differential equations, and aggregated over the entire population in order to determine population-level dynamics. Furthermore, individual agents can interact with one another, in terms of a social network structure. The environment is firmly grounded in the system dynamics approach, and, unlike conventional agent-based simulation environments, programming is not required in order to specify agent interactions and behaviours. The approach is validated by using the classic SIR model of contagion.

Keywords: System Dynamics, Agent Based Modelling, Simulation, SIR Models

INTRODUCTION

This paper proposes an extension to the System Dynamics (SD) method in order to provide a novel way for modeling multi-agent systems. Gilbert and Troitzsch (2005) comment that “a natural way of programming agents is to use an object-oriented programming language,” and frameworks such as RePast (North et al. 2006), AnyLogic (Borschev and Filippov 2004), and NetLogo are commonly used to achieve this goal. A characteristic of the ABM approach is the focus on agent heterogeneity, namely, identifying the differences in agents and simulating their interactions and behaviour over time. System Dynamics is an alternative approach to agent-based simulation, and employs as a robust and well-defined methodology to model the behaviour of decision making entities. The resulting simulations are run in continuous time, and, do not require programming expertise on behalf of the modeler. However, a disadvantage of the system dynamics approach regards the scale of agent models. Current system dynamics tools are not amenable to the construction of large scale agent societies, and so the possibilities for extending the heterogeneity of models is limited. The approach presented here addresses provides an approach and technology that allows large scale agent models to be built, based on sets of differential equations.

SYSTEM DYNAMICS

The systems approach to problem solving has many strands and influences, and originally emerged as a reaction to the reductionism advocated by the traditional “divide and conquer” approach to science, namely, that in order to understand a complex system, one must take it apart and understand its constituent parts. Systems thinking involves taking a holistic approach to problem solving, by identifying the manner in which systems interact in order to uncover insights regarding overall systems behaviour. System Dynamics (SD) is one of the most widely used systems approaches in the world, and was created and pioneered by Jay W. Forrester

* Corresponding author address: Jim Duggan, Department of Information Technology, NUI, Galway, Ireland.

(Forrester 1958, Forrester 1961) as a means of simulating the behaviour of social systems, explaining that behaviour, and crafting effective long term policies (Lane 2006). Sterman (2000) defines SD as “a perspective and a set of conceptual tools that enable us to understand the structure and dynamics of complex systems” and, because of its underlying mathematical foundation, SD is also “a rigorous modeling method that enables us to build formal computer simulations of complex systems and use them to design more effective policies and organizations.”

Forrester’s key insight followed on his study of manufacturing supply chains, and his observation that the chaos apparent in inventory levels throughout the supply chain could not be explained solely by external effects. He demonstrated that many problems were in caused internally, by the decisions of managers who were not fully aware of the feedback structures that were present in the overall system. Forrester argued that feedback is a critical component of human decision making, as many management decisions lead “to a course of action that changes the state of the surrounding system and gives rise to new information on which future decisions are made” (Forrester 1969).

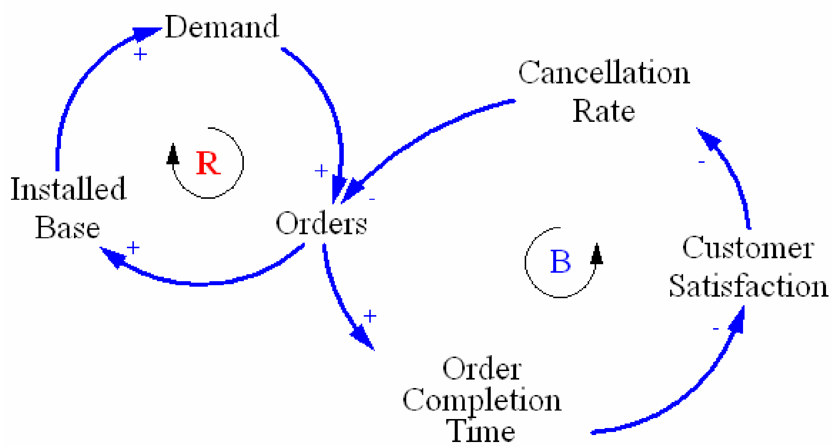


FIGURE 1 Limits to growth in a feedback system

Decision makers who do not take account of the feedback view often encounter policy resistance, where “well-intentioned policies are delayed, diluted and defeated by the unforeseen reactions of other people” (Sterman 2000). A simple example of this is captured in Figure 1, which shows two feedbacks. The first, a *reinforcing feedback,* models the “word of mouth” phenomenon, where demand fuels orders, which in turn increase the installed base, and this leads to further demand. The second loop, a *balancing feedback,* acts to dampen this growth, because capacity limitations in the system mean that as orders rise, so to does the order completion time, which reduces customer satisfaction, and this in turn leads to order cancellations. It is the interaction of these loops that determines how the system behaves over time, and while sales may grow exponentially at first, any serious capacity constraint will cause this growth to flatten over time. A possible solution to this would be to increase capacity early (a proactive strategy) in the product’s life cycle, rather than increasing the capacity at a later stage (a reactive strategy).

Figure 1 illustrates how feedback structures can be captured qualitatively, using causal loop diagrams. In order to formally model dynamic systems, these feedbacks must be represented quantitatively, through *stock and flow* diagrams. Flows are formal expressions of policy, where equations are formulated to capture the key decision rules that control the rates of flow through a system. Stocks are accumulations (e.g. number of employees, balance in a bank account, number of people queuing), and the completed models are run as a set of non-linear differential equations. However, a disadvantage of current system dynamics tools is that they do not provide the mechanism to construct large-scale agent models. The approach presented here addresses these shortcomings by providing an approach and technology that allows large scale agent models to be built, based entirely on sets of differential equations.

SYSTEM DESIGN

Figure 2 illustrates the conceptual design, and this is an adaptation of Sterman's (2000, p.515) representation of how decision rules govern the rates of flow in systems. The central idea is that each agent is represented by an individual stock and flow structure. An agent changes its state by processing information cues: in this generic model, these cues can be based on the agent's own state, information from other agents, and information from the aggregate system state. This aggregate system state is a summation of the individual rates of change for each agent in the population, and so feedback exists between the aggregate level and the individual agent level.

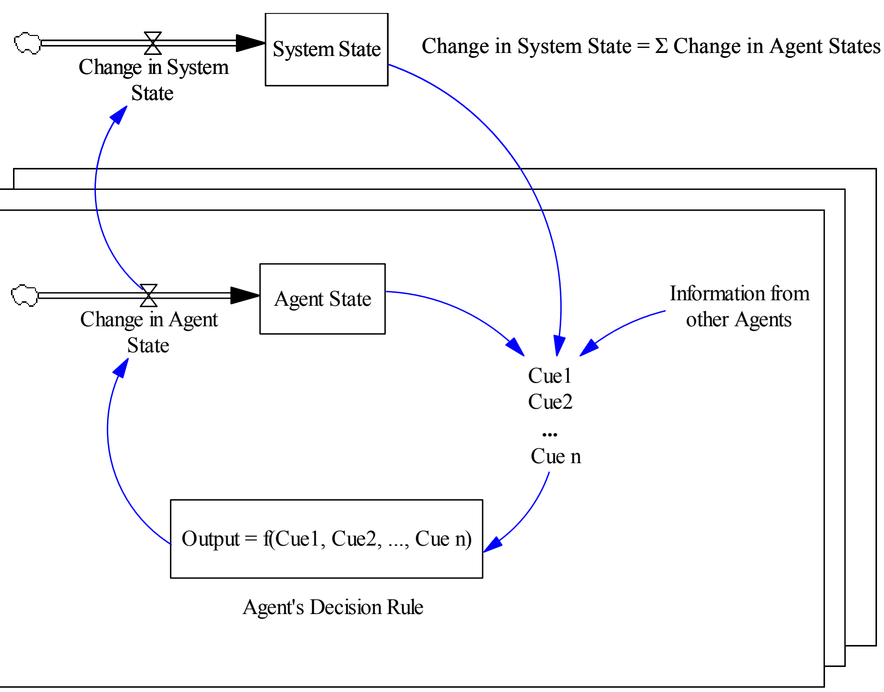


FIGURE 2 Conceptual design for the continuous agent based modeller

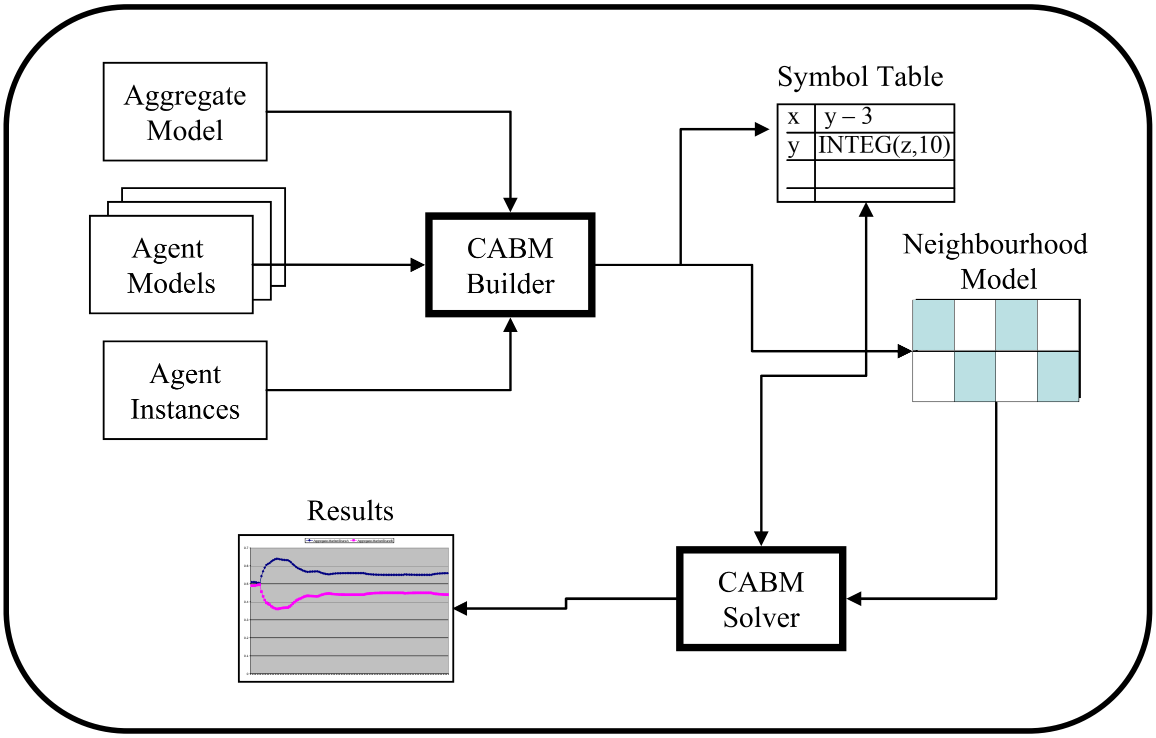


FIGURE 3 Architecture for the continuous agent based modeller

The system architecture is captured in Figure 3. The major software components are:

CABM Builder. This takes as input three model types, and creates: (1) a symbol table containing each equation to be solved, including stocks, flows and auxiliaries, and (2) a neighbourhood model that places each agent in a grid-like structure so that information cues from neighbours can be taken into account for an agent's decision making. The number of equations created is a function of the number of agents. For example, if each agent is represented by thirty equations, and there are one hundred agents, then there will be three thousand equations in the symbol table (excluding those equations that are specified as part of the aggregate model).

CABM Solver. This solves all numerical equations contained in the symbol table, and also has special-purpose routines that can aggregate variables in the model, and calculate neighbourhood values for each individual agent.

There are three main types of input for the CABM Builder:

The aggregate model, which corresponds to the "System State" element of Figure 2. These are the stocks and flows that capture the aggregate dynamics for the system of interest. The key states in this model will change based on an aggregation of all the changes at the agent (or lower) level of the model.

The agent models, which express important agent heterogeneities in the system under consideration. Each agent model can have different formulations for key decision equations.

The agent instances, which specify how many of each agent are to be created for a simulation run, and also can be used to vary the specified parameter values of individual agents.

In order for the system to work, three categories of mapping must be achieved when the models are combined. The first of these is mapping from the detailed level to the aggregate level, where a high-level flow is formulated based on an aggregation of lower level, or agent, flows. A stock and flow is identified as an aggregate by including the tag “<is_aggregrate>” as part of it definition (see Figure 4). For an aggregate flow, no equation is specified: instead, a purpose-built function – called AGGREGATOR() – is invoked, and during the simulation run this function will aggregate all the relevant agent flows.

```
<stock>
    <name>Aggregate.Susceptible</name>
    <is_aggregate>true</is_aggregate>
    <init>0.0</init>
    <outflow>Aggregate.InfectionRate</outflow>
</stock>

<flow>
    <name>Aggregate.InfectionRate</name>
    <is_aggregate>true</is_aggregate>
    <equation>AGGREGATOR()</equation>
</flow>
```

FIGURE 4 Defining stocks and flows at an aggregate level

At the agent level, any flow that must aggregate to a higher level is specified with the tag “<is_subflow>”, and the super flow, which is the flow that it aggregates to, is specified (Figure 5). In this example, the string “$NAME$” will be replaced by the specified agent name when the model created. Stocks at the agent level are defined in a similar manner (Figure 6). Furthermore, if a rate at the agent level is procedurally complex, a callout routine can be written (in C#) to evaluate it. Rates at the agent level that mirror an aggregate rate are usually programmed to flow over one time step, and so state switching occurs at discrete points in the simulation.

```
<flow>
    <name>$NAME$.RecoveryRate</name>
    <is_subflow>true</is_subflow>
    <super_flow>Aggregate.RecoveryRate</super_flow>
    <equation>DELAYFIXED($NAME$.InfectionRate, $NAME$.RecoveryDelay,0)</equation>
</flow>
```

FIGURE 5 Defining flows at an agent level

```xml
<stock>
    <name>$NAME$.Recovered</name>
    <is_substock>true</is_substock>
    <super_stock>Aggregate.Recovered</super_stock>
    <init>$RECOVERED_INIT$</init>
    <inflow>$NAME$.RecoveryRate</inflow>
    <capture_state>true</capture_state>
    <value_if_true>300</value_if_true>
</stock>
```

FIGURE 6 Defining stocks at an agent level

The final mapping relates to agent-to-agent communications, where an agent uses information from other agents in order to arrive at a decision. In order to facilitate this, the agents are modelled as a society in a grid-like structure. In figure 7, a society of 100 agents are shown, where each of these has a similar stock and flow structure. The purpose-built function NEIGHBOURHOOD_AVERAGE() will find the average for a given model variable from across all of its immediate neighbours, and this value that then be used as an important cue in an agent's decision making process. This value would play an important part in triggering whether or not an individual agent may change its state (for example, from susceptible to infected in a model of contagious disease).

```xml
<auxiliary>
    <name>$NAME$.NeighboursAvg</name>
    <equation>NEIGHBOURHOOD_AVERAGE($NAME$, $NAME$.Infected)</equation>
</auxiliary>
<flow>
    <name>$NAME$.InfectionRate</name>
    <is_subflow>true</is_subflow>
    <super_flow>Aggregate.InfectionRate</super_flow>
    <equation>EXTERNAL(0)+($NAME$.NeighboursAvg*0)+($NAME$.Random*0)</equation>
    <parameter>$NAME$.Susceptible</parameter>
    <parameter>$NAME$.Infectivity</parameter>
    <parameter>$NAME$.NeighboursAvg</parameter>
    <parameter>Aggregate.TimeStep</parameter>
    <parameter>$NAME$.Random</parameter>
</flow>
```

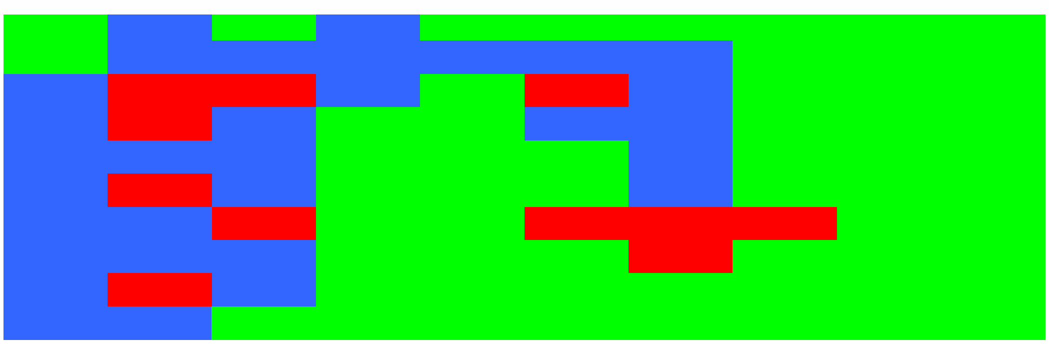


FIGURE 7 Information exchange between agents and the grid society structure

CASE STUDY: SIR MODEL

To illustrate how the system operates, a well-known case – the SIR Model - is selected. This has been widely modeled using both SD and agent-based methods. For this solution, the aggregate and agent components are shown in Figure 8. At an aggregate level the population is divided into three stocks: susceptible (S), Infected (I) and Recovered (R). The infection and recovery rates determine the rate of flow between these stocks. However, unlike the usual SD approach, these rate equations simply flow aggregators, and are determined by what happens at each individual agent level.

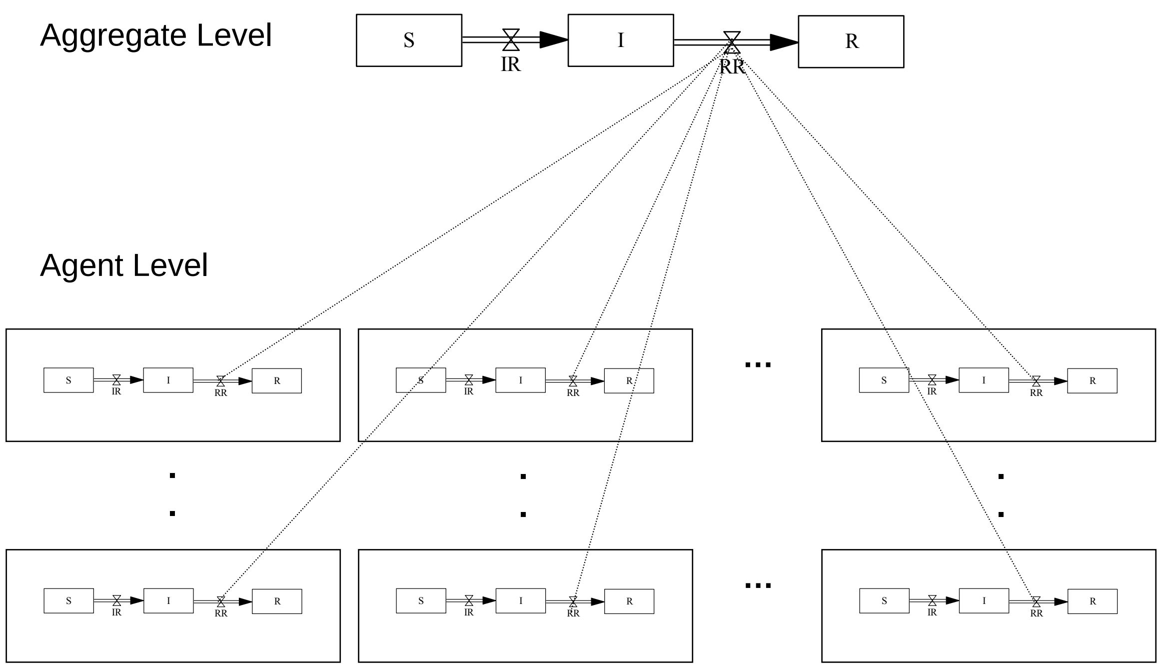


FIGURE 8 Two-level model of the SIR phenomenon

At each agent level, the SIR structure is employed, however in this case, the sum of all states must be 1 (i.e. an individual agent can only be one of S, I or R at any one time), and the rates are not continuous, so that states change at one point in time in a switching-type action. The recovery rate is simply a pipeline delay based on the infection rate (i.e. recovery follows infection after a certain number of time has elapsed). The infection rate is determined by: (1) the proportion of neighbours that are already infected and (2) the infectivity of the agent. An overall infection probability is calculated, and a [0,1] random number generated in order to decide whether an agent has been infected. If this happens, their state changes accordingly.

At a technical level, the complete model can be specified using mathematical equations, with a minimum of actual code. Underlying the model is a grid-based social network, where each agent occupies a cell on the grid, and is influenced by its neighbours, a structure that has parallels with a cellular automata model. Sample results from a simulation are now shown.

Figure 9 shows the overall aggregate behaviour of the agents. There are 100 in total, and the stock of susceptible starts at 95, and depletes as the infection spreads. As the recover rates pick up, the spread of the infection slows, and an equilibrium is reached.

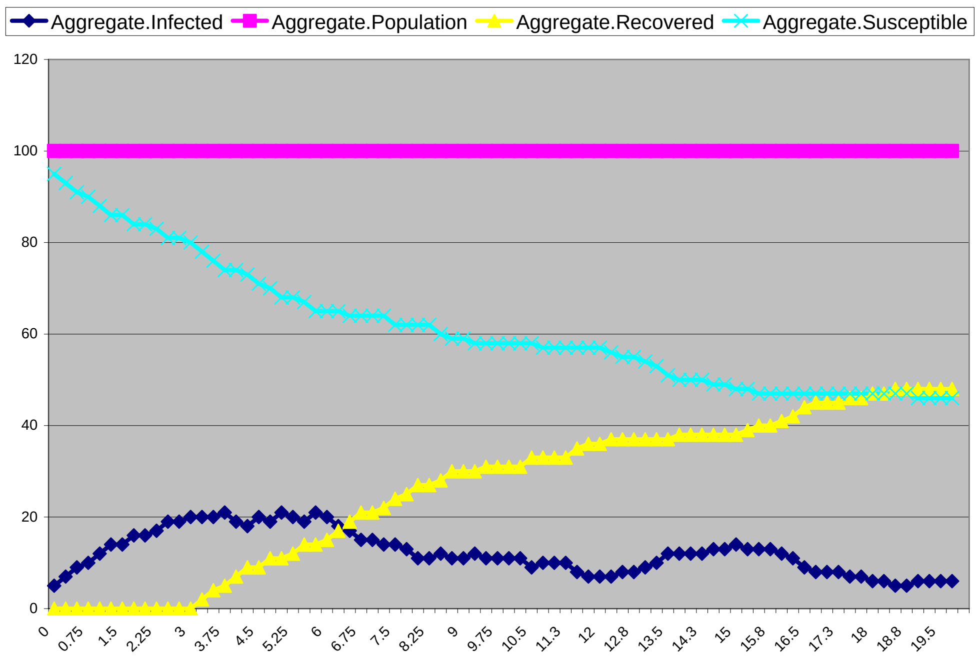


FIGURE 9 Aggregation of results by SIR categories

A more detailed analysis can also be viewed based on the overall state of the "grid" as time progresses. Figure 10 shows how the agent states have changed over a time interval of 5. Each rectangle represented 100 agents, and green maps onto susceptible, red is infected and blue recovered.

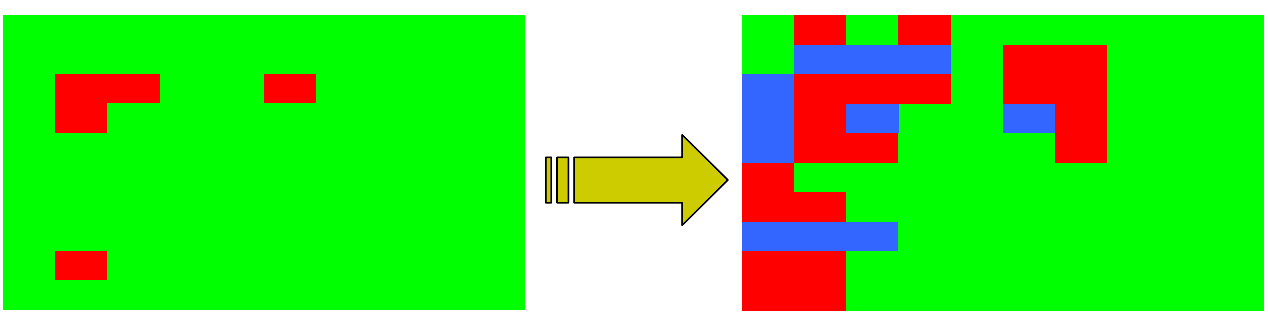


FIGURE 10 Change in agent states from time 0 to 5

Finally, Figure 11 shows a detailed trace of how individual states change over time. The time axis is vertical, and ranges from 0 through to 20. Each row across the diagram represents an agent's state at a particular point in time, and based on this it can be observed when an individual agent changed from one state to another.

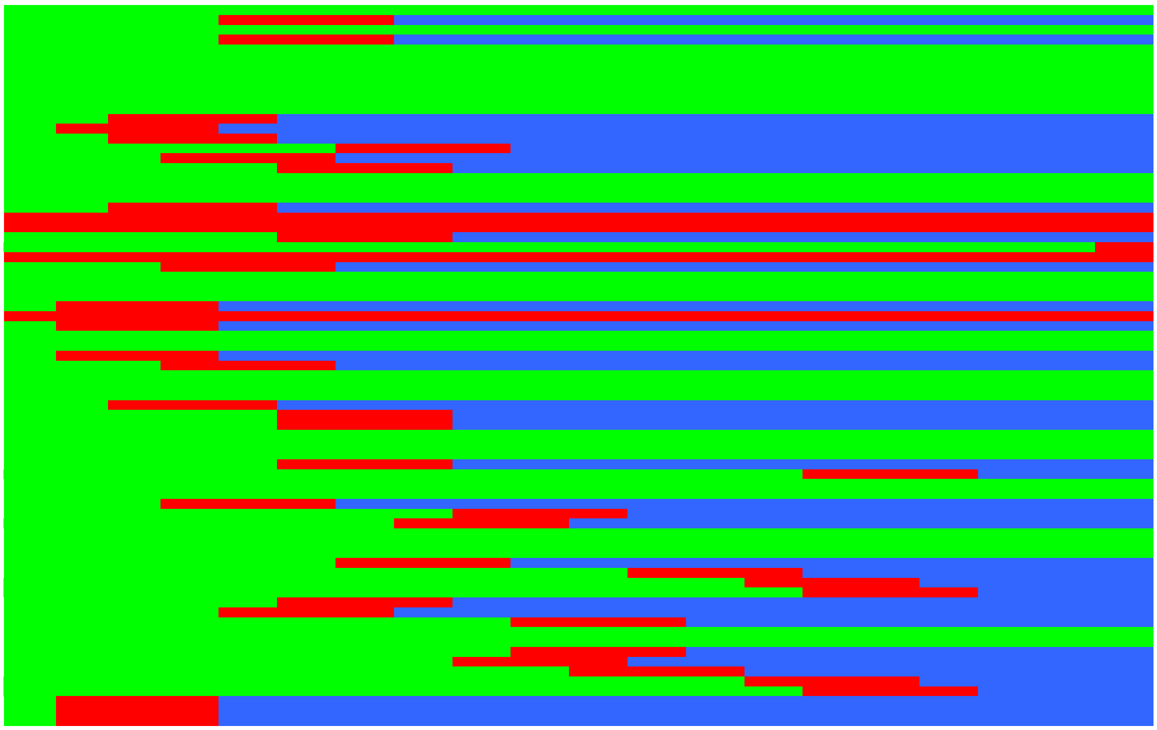


FIGURE 11 Mapping of agent state changes over time

CONCLUSIONS

This paper has presented an approach and a simulation system that can model agent-based systems using System Dynamics. There are a number of advantages to this, including:

- *Building on existing knowledge.* Models built using this approach have access to the rich body of knowledge within the field, including a wide variety of models that capture dynamic decision making processes across a range of disciplines.

- *Scalability and Performance.* Given a compact and lightweight numerical solver, this approach is scalable and should be able to accommodate a high number of agents and calculate results speedily.

Future work will include building a graphical user-interface for the current system, and also constructing a high performance numerical solver that will have the capability to simulate large populations of individual agents.

REFERENCES

Akkermans, H. 2001. "Emergent Supply Networks: System Dynamics Simulation of Adaptive Supply Agents." *Proceedings of the 24th Hawaii International Conference on Systems Sciences*, ISSN 0-7695-0981-9/01.

Borshchev A. and A. Filippov. 2004. "From System Dynamics and Discrete Event to Practical Agent Based Modeling: Reasons, Techniques, Tools." *Proceedings of the 22nd International Conference of the System Dynamics Society*. Oxford, England. 2004.

Forrester, J.W. 1958. "Industrial Dynamics: a major breakthrough for decision makers." *Harvard Business Review*. 36, (4), 37-66.

Forrester, J.W. 1961. *Industrial Dynamics*. MIT Press, Cambridge, MA.

Forrester, J.W. 1969. *Urban Dynamics.* MIT Press, Cambridge, MA.

Gilbert, N. and Troitzsch, K. 2005. *Simulation for the Social Scientist*, Open University Press.

Lane, D.C. 2006. IFORS' Operational Research Hall of Fame: Jay Wright Forrester. Intl. Trans. In Op. Res. 13 (2006) 483-492.

North, M.J., Collier, N.T., and Vos, J.R. 2006. "Experiences Creating Three Implementations of the Repast Agent Modeling Toolkit." *ACM Transactions on Modeling and Computer Simulation*. Vol.16, No.1, pp 1-25.

Sterman, J. 2000. *Business Dynamics. Systems Thinking and Modeling for a Complex World.* McGraw Hill Higher Education

Socio-Technical Systems

ENSEMBLE COMPUTING IN AGENT-BASED MODELING FOR TRANSCENDING PARADIGMATIC BOUNDARIES IN DECISION THEORY – UNDERSTANDING TRIBAL POLITICS

L.A. KUZNAR*, Indiana University – Purdue University, Fort Wayne, Fort Wayne, IN

ABSTRACT

The data outputs of agent-based models are extremely complex and that complexity is compounded by the uncertainty surrounding model parameters. Ensemble computing utilizes sets of models with varied parameter setting to explore the potential worlds a model could produce. This uncertainty is compounded when different models appear to derive from different scientific paradigms. I present a framework in which models derived from different paradigms are placed in an ensemble, whose varied outputs allow exploration of the various strengths and weaknesses of different decision theoretic paradigms. Specifically, theories and models derived from rational choice theory, risk sensitivity theories, social psychology, prospect theory, bounded rationality and culture norm theories are compared. Empirical data on political alliances in a patrilineal New Guinea tribe are used to measure the relative effectiveness of different theories. While ensemble computing can test competing hypotheses, its best use is in identifying the key elements any theory must have to be explanatory and predictive. This work applies to research on decision making and traditional tribal politics, including in regions of the world in turmoil such as Sudan, Somalia, Iraq, and Afghanistan.

Keywords: exploratory modeling; ensembles; decision theory; theory testing

INTRODUCTION

This paper is a follow-up to a modeling study of coalition formation in a New Guinea tribal village [Kuznar, 2006]. The evolution of coalitions among men in this village was used to test the efficacy of theories of political and economic decision-making drawn from rational choice, prospect theory, bounded rationality and risk sensitivity paradigms. The preliminary study identified four models that performed particularly well, sigmoid group (SG), full prospect theory (PT) (including effects of probability weighting, loss aversion and framing), and the smart agent prestige bias (sP3) and smart conformism 2 (sC2) models. In this paper, I present the results of more rigorous validation metrics that allowed finer discrimination among the competing theories.

In this paper, I present the results of more robust metrics used for exploring the strengths and weaknesses in decision theory models. The empirical case example used to test these models concerns the evolution of political alliances among men in a tribal village of New Guinea (present day Irian Jaya). This case provides an example of how exploratory modeling may enhance scientific evaluation, offers a preliminary test of decision theories, and suggests future hypotheses.

* *Corresponding author address*: Lawrence A. Kuznar, Dept. Anthropology, Indiana University – Purdue University, Fort Wayne 46805; e-mail: kuznar@ipfw.edu

EXPLORATORY MODELING

In exploratory modeling, the breadth of scientific ideas is captured in an ensemble of alternative models, rather than a single comprehensive model [Bankes, 2002:7264; Lempert, et al., 2006; Kleijnen, 1997]. Then, the resulting parameter space from these alternatives is searched for models that explain phenomena or models that are robust against perturbations of their parameters [Lempert, et al., 2006]. Exploratory modeling has been used for applied purposes such as weather forecasting [Palmer, 2000] and policy analysis [Bankes, 1993]. Since social scientists often propose theories derived from different paradigms, exploratory modeling may assist them in dealing with their own deep uncertainty. I present a relatively simple case where 24 decision models, derived from several different paradigms, are tested against one another to explore their relative explanatory power. I concentrate on only versions of the models that correspond to specific published propositions. A full exploration of each model's parameters and variables would require the use of the more sophisticated sampling strategies enumerated above.

MODELING THE KAPAUKU OF IRIAN JAYA (NEW GUINEA)

The Kapauku are a tribal people who live in the highlands of Irian Jaya. Their economy is based on growing yams and raising pigs, they control territories that contain their farmland and villages, they have cultural norms of patrilineal descent, and they practiced extensive warfare in the first half of the 20th century. The anthropologist Leopold Pospisil made detailed and extensive observations of Kapauku economy and politics during the decade of the 1950's, and he published data on the individual economics and political affiliations of the 55 adult men who comprised the political network of the Kapauku village of Botekubo [Pospisil, 1963, 1972]. Two prominent features of Kapauku culture are men's obsession with wealth acquisition and the intensely political nature of men's lives. Kapauku political coalitions center around *tonowi* (wealthy men), who are both economically successful and politically powerful [Pospisil, 1963:11, 48]. I use Pospisil's data on individual men's wealth and political affiliations to test competing theories of decision making by simulating men's decisions with theorized decision rules and examining which rules produce Kapauku-like alliances.

I have developed a general computational model of risk-taking in which agents interact via a coordination game with an optimal Nash mixed strategy of probabilistically cooperating and defecting with partners [Kuznar, et al., 2006]. This general model was adapted to represent the political behavior of the 55 men in Botekubo. The simulation begins with each man in his own alliance, and coalitions evolve as men join or defect on one another according to programmed decision models. Competing decision models are evaluated based on the speed and accuracy with which alliances structurally similar to those observed in Botekubo form.

DECISION THEORY

The field of decision theory is divided among several different paradigmatic lines, including traditional (canonical) rational choice, various bounded rationality approaches, and prospect theory. Sigmoid utility represents another alternative, in part derived but also departing from rational choice [Kuznar, 2007]. Each paradigm gives rise to numerous specific theories.

Rational Choice

Core elements of rational choice include the assumptions that individuals have full knowledge of their preferences and resources, that individuals maximize their utility, and that individuals are selfish [Cowell, 1986:Chapter 4]. Nash optimal solutions to competitive or cooperative interactions assume rational capabilities and so represent rational choice decision models.

Sigmoid Utility

Sigmoid utility theory maintains that an individual's position in a wealth distribution influences that individual's sensitivity toward risk [Kuznar, 2002; Friedman and Savage, 1948; Kuznar, 2007]. Individuals on the cusp of a class boundary, where increases in social rank (climbing the social ladder) bring large increases in wealth and status, are expected to be risk prone, or to take chances. I have applied this approach to understanding various forms of political behavior from voting, to political coups, to rebellions, to modern day terrorism [Kuznar and Frederick, 2003; Kuznar, et al., 2006; Kuznar, 2007]. Since joining a group of unknown individuals carries risk, risk prone individuals are more likely to join, and risk averse individuals are least likely to join. This approach is derived from rational choice, but departs by being particularly sensitive to others' payoffs and by allowing envy at others' well-being (rather than greed for one's self) as a motivator.

Group Affiliation

Social psychologists argue that small group dynamics can override selfish motives, especially in extremely risk-prone groups that tend to become highly socially isolated [Atran, 2003]. Therefore, the social psychological effect of small group dynamics on members of a group will be the reverse of the effects on individuals regarding risk sensitivity. Agents' probability of joining with non-members will be inversely proportional to their group's risk sensitivity; members of highly insular groups never join with outsiders. By using sigmoid utility theory and Arrow-Pratt measures, this model combines elements of sigmoid and small group psychology paradigms.

Prospect Theory

Prospect theory [Kahneman and Tversky, 1979, 2000] is a collection of propositions about human decision making that are derived from and empirically supported by experimental studies. Prospect theory's three core propositions are that people systematically distort probabilities (overestimating low probabilities and underestimating high probabilities), that people are loss averse (experiencing twice the disutility of a loss than the utility of an equal gain), and that framing profoundly affects decision-making with people (people are risk prone when considering losses and risk averse when considering gains) [Kahneman, 2000]. Prospect theorists have derived mathematical functions for probability weighting [Prelec, 2000:77] and the disutility of loss aversion [Tversky and Fox, 2000:104; Tversky and Kahneman, 1992:57] and I use these functions to model probability weighting (PW) and loss aversion (LA) respectively. I model framing (FR) by recording whether an agent's wealth has increased or decreased, assigning an adjusted Nash optimal join probability for agents in a frame of gains or the reciprocal probability for agents in a frame of decreases.

Prestige Bias

Prestige bias is the imitation of those with higher social status [Boyd and Richerson, 1985], and is a simple heuristic proposed by bounded rationality theorists. Prestige bias theories fail to specify the scales at which it operates. Therefore, I modeled prestige bias at different scales including imitating a higher-status partner (Prestige 1, P1), imitating the household patriarch (Prestige 2, P2), imitating the wealthiest member of a coalition (Prestige 3, P3), and imitating the wealthiest member of the society (Prestige 4, P4).

Conformist Transmission

Conformist transmission refers to the copying of normative behavior in a society [Boyd and Richerson, 1985], and is another bounded rationality decision heuristic. As with prestige bias theory, conformist transmission theory offers no guidance as to what social norms are copied, those of a neighborhood, a tribe, a nation, or the global village. Consequently, I developed alternative models of conformist transmission including conformism to one's household (Conformism 1, C1), to one's alliance (Conformism 2, C2), and to the entire society (Conformism 3, C3). Models assuming that probabilities were drawn on a [0,1] interval (naïve agents) vs. probabilities that bracketed the Nash optimum (smart agents) were run for both the prestige bias and conformism models. The models that bracketed the Nash optimum combine elements of quasi-rational choice with bounded rationality.

Validation Metrics

Evaluating the goodness of fit of computational models is challenging. To date, many validations consist of producing graphical outputs (or often 2-dimensional geographic maps) that look like some stylized fact the researchers are modeling [Kohler, et al., 2005; Lansing, 1993; Kuznar and Sedlmeyer, 2005]. Some computational methodologists label this "viewgraph validation," and point out that while intuitive visual aids can be a useful starting point for validating models, they fall short of rigorous and thorough validations of the model [Kleijnen, 1995; Oberkampf and Trucano, 2002; Oberkampf, et al., 2004].

It is necessary to use quantitative measures of a model's performance in order to evaluate the degree to which different models are successful and in what ways [Oberkampf, et al., 2004]. Simulation researchers use Thiel's Inequiality Coefficient (TIC) to compare model and empirical outputs and have expanded it to incorporate multiple dimensions of goodness of fit [Murray-Smith, 1998; Kheir and Holmes, 1978]. The single-dimension metric is calculated as:

$$TIC = \frac{\sqrt{\sum_{i=1}^{n}(y_i - z_i)^2}}{\sqrt{\sum_{i=1}^{n} {y_i}^2} + \sqrt{\sum_{i=1}^{n} {z_i}^2}} = \frac{n_{yz}}{d_y + d_z}$$

Where y_i is an empirical measure, z_i is a model output that corresponds to the empirical measure, i indexes the ith model run, and n is the number of runs. We use Thiel's Inequality Coefficient to compare the number of coalitions generated by a model and mean coalition size to Pospisil's data. Its values vary from 0 to 1, with 0 indicating a close fit (no difference). The normalized values of the TIC allow comparison of different models and different performance variables.

Point measures like means, variances and TICs are commonly used and provide one means of evaluating goodness of fit. However, one can have identical means drawn from very different data distributions. The Kolmogorov-Smirnov D – Statistic provides a non-parametric test for the equality of distributions [Blalock, 1979:266-269]. Models that produce statistically significantly different data distributions clearly perform poorly, and models with large p – values produce data distributions statistically indistinguishable from actual data.

RESULTS

An ensemble of 24 models represents the basic propositions of these theories, derived from four paradigms (rational choice, sigmoid utility, small group social psychology, prospect theory) (Table 1). Each model was run 100 times, and 10 model runs were selected from each run for analysis of how quickly the model converged to alliances similar to those empirically observed in the tribe. The performance of each model at iteration 15 was used to standardize the comparisons.

Table 1. Relationship between Decision Theoretic Paradigms and Decision Models Tested in Kapauku Simulation.

Paradigms	**Models**
Rational Choice	Nash optimum (N)
Modified Rational Choice	Sigmoid utility (S)
Modified Rational Choice / Social Psychology	Sigmoid utility+Group affiliation (SG)
Prospect Theory	Probability weighting (PW), Loss aversion (LA), Framing effects (FR), PW+LA, PW+FR, LA+FR, PW+LA+FR
Bounded Rationality	naïve Prestige bias 1 (nP1), naïve Prestige bias 2 (nP2), naïve Prestige bias 3 (nP3), naïve Prestige bias 4 (nP4), naïve Conformism 1 (nC1), naïve Conformism 2 (nC2), naïve Conformism 3 (nC3)
Bounded Rationality / quasi-Rational Choice	smart Prestige bias 1 (sP1), smart Prestige bias 2 (sP2), smart Prestige bias 3 (sP3), smart Prestige bias 4 (sP4), smart Conformism 1 (sC1), smart Conformism 2 (sC2), smart Conformism 3 (sC3)

Four models provided close fits to empirical data, including sigmoid utility (S), sigmoid group affiliation (SG), smart conformism 2 (sC2), and loss aversion (LA). These results differ from an earlier effort that did not use Theil's Inequality coefficient and Kolmogorov-Smirnov. In that earlier study [Kuznar, 2006], a prestige bias model and the full prospect theory model also performed well. The results from using more robust metrics are more discriminating. They also are more discriminating among the well-performing models. Two models consistently performed well across all metrics, and they were the sigmoid utility (S) and smart conformism 2 (sC2).

Table 2. Model Performance in the Kapauku Simulation.

Model	TIC Coalition Number	TIC Mean Coalition Size	Difference from Actual Coalition Number	Difference from Actual Coalition Size	No. Distribution Matches per 20 runs	Kologorov-Smirnov mean p-value
Nash	0.118	0.105	3.35	0.55	**15**	**0.258**
Sigmoid	**0.074**	**0.066**	**1.40**	**0.24**	**15**	0.252
Group Affiliation	**0.095**	**0.085**	**2.30**	**0.39**	11	0.199
Prestige I	0.109	0.096	3	0.50	13	**0.310**
Prestige II [0,1]	0.267	0.252	11	1.37	1	0.024
Prestige III [0,1]	0.240	0.220	9.2	1.21	4	0.072
Prestige IV [0,1]	0.242	0.223	9.35	1.22	1	0.034
Prestige II [0.3,0.9]	0.145	0.128	4.2	0.64	14	**0.275**
Prestige III [0.3,0.9]	0.150	0.141	5.15	0.81	13	0.173
Prestige IV [0.3,0.9]	0.245	0.225	9.5	1.23	3	0.050
Conformism I [0,1]	0.204	0.181	7.1	0.99	5	0.112
Conformism II [0,1]	0.248	0.240	10.2	1.32	0	0.019
Conformism III [0,1]	0.213	0.192	7.7	1.07	7	0.172
Conformism I [0.3,0.9]	0.130	0.113	3.15	**0.47**	13	0.202
Conformism II [0.3,0.9]	**0.083**	**0.073**	**2**	**0.35**	**15**	0.230
Conformism III [0.3,0.9]	0.120	0.107	3.4	0.55	**16**	0.181
PW (probability Weighting)	0.171	0.163	6.2	0.94	10	0.199
FR (Framing)	0.210	0.199	8	1.12	7	0.079
LA (Loss Aversion)	**0.090**	0.103	**1.7**	0.49	**17**	**0.371**
PW FR	0.187	0.175	6.8	1.00	5	0.088
FR LA	0.365	0.365	17.95	1.80	0	0.004
PW FR LA	0.284	0.271	12.1	1.45	3	0.042

(Full Prospect Theory)						
Mean	0.181	0.169	6.58	0.896	8.5	0.152
s.d.	0.077	0.075	4.15	0.422	5.9	0.106
Threshold	**<0.104**	**<0.094**	**<2.43**	**<0.474**	**>14.4**	**>0.258**

A more fruitful approach is to explore new hypotheses by asking what the successful models had in common. Successful models had two characteristics in common: 1) agents behaved in a quasi-optimal manner by selecting strategies that did not deviate far from Nash optimality, and 2) agents were not homogenous in their decisions. Therefore, the specific models derived from four different paradigms might not so much accurately represent reality as much as capture some essential elements that a model must have to be valid.

CONCLUSION

Computational models provide new and flexible capabilities for representing social theories from different paradigms. Exploratory modeling using ensembles of models provides a method by which competing theories can be tested. The result of the testing may not be a single correct answer, but insights into what essential elements better theories must contain. In the Kapauku case, theories related to rational choice, prospect theory, and bounded rationality each has some merit. In particular Kapauku men appear to have a general sense of what an optimal political strategy is, they may be imitating one another to refine their strategies, and their decisions appear to be conditioned by prospect theory biases, risk sensitivity, and group pressures to conform. Exploratory modeling with ensembles provides a method for more systematically searching the implications of these theories and suggesting new hypotheses that may aid in the search for more comprehensive and valid theories.

References

[Atran, S., 2003] Atran, S., 2003, "Genesis of Suicide Terrorism," *Science* **299**(5612):1534-1539.

[Bankes, S., 1993] Bankes, S., 1993, "Exploratory Modeling for Policy Analysis," *Operations Research* **41**(3):435-449.

[Bankes, S. C., 2002] Bankes, S. C., 2002, "Tools and Techniques for Developing Policies for Complex and Uncertain Systems," *Proceedings of the National Academy of Sciences* **99**(Supplement 3):7263-7266.

[Blalock, H. M., 1979] Blalock, H. M., 1979, *Social Statistics, Revised Second Edition*, New York: McGraw-Hill, Inc.

[Boyd, R. and P. J. Richerson, 1985] Boyd, R. and P. J. Richerson, 1985, *Culture and the Evolutionary Process*, Chicago: University of Chicago Press.

[Cowell, F., 1986] Cowell, F., 1986, *Microeconomic Principles*, Oxford: Oxford University Press.

[Friedman, M. and L. J. Savage, 1948] Friedman, M. and L. J. Savage, 1948, "The Utility Analysis of Choices Involving Risk," *Journal of Political Economy* **4**:279-304.

[Kahneman, D., 2000] Kahneman, D., 2000, "Preface,"in *Choices, Values, and Frames*, Kahneman, D. and A. Tversky (Ed.), pp. i-xvii, Cambridge: Cambridge University Press.

[Kahneman, D. and A. Tversky, 1979] Kahneman, D. and A. Tversky, 1979, "Prospect Theory: An Analysis of Decision under Risk," *Econometrica* **47**(2):263-291.

[Kahneman, D. and A. Tversky, 2000] Kahneman, D. and A. Tversky, ed. 2000, *Choices, Values, and Frames*, Cambridge: Cambridge University Press.

[Kheir, N. A. and W. N. Holmes, 1978] Kheir, N. A. and W. N. Holmes, 1978, "On Validating Simulation Models of Missile Systems," *Simulation* **30**(April 1978):117-128.

[Kleijnen, J. P. C., 1995] Kleijnen, J. P. C., 1995, "Verification and Validation of Simulation Models," *European Journal of Operational Research* **82**:145-162.

[Kleijnen, J. P. C., 1997] Kleijnen, J. P. C., 1997, "Sensitivity Analysis and Related Analyses: A Review of Some Statistical Techniques," *Journal of Statistical Computer Simulation* **57**:111-142.

[Kohler, T. A., G. J. Gumerman and R. G. Reynolds, 2005] Kohler, T. A., G. J. Gumerman and R. G. Reynolds, 2005, "Simulating Ancient Societies," *Scientific American* **293**(1):67-73.

[Kuznar, L. A., 2002] Kuznar, L. A., 2002, "Evolutionary Applications of Risk Sensitivity Models to Socially Stratified Species: Comparison of Sigmoid, Concave and Linear Functions," *Evolution and Human Behavior* **23**(4):265-280.

[Kuznar, L. A., 2006] Kuznar, L. A., 2006, "Exploratory Modeling of Ensembles for Testing Decision Theory Paradigms,"in *Proceedings of the Agent 2006 Conference: Social Agents: Results and Prospects*, Sallach, D. L., C. M. Macal and M. J. North (Ed.), pp. 157-166, Chicago, Illinois: Argonne National Laboratory.

[Kuznar, L. A., 2007] Kuznar, L. A., 2007, "Rationality Wars and the War on Terror: Explaining Terrorism and Social Unrest," *American Anthropologist* **109**(2):318-329.

[Kuznar, L. A. and W. G. Frederick, 2003] Kuznar, L. A. and W. G. Frederick, 2003, "Environmental Constraints and Sigmoid Utility: Implications for Value, Risk Sensitivity, and Social Status," *Ecological Economics* **46**:293-306.

[Kuznar, L. A. and R. L. Sedlmeyer, 2005] Kuznar, L. A. and R. L. Sedlmeyer, 2005, "Collective Violence in Darfur: An Agent-Based Model of Pastoral Nomad/Sedentary Peasant Interaction," *Mathematical Anthropology and Culture Theory* **1**(4):1-22.

[Kuznar, L. A., J. Toole and N. Kobelja, 2006] Kuznar, L. A., J. Toole and N. Kobelja, 2006, "Emergent Agents and the Simulation of Political Unrest: Application to Palestinian Political Coalitions," *Proceedings of the AGENT 2005 Conference: Generative Social Theory*

[Lansing, J. S., 1993] Lansing, J. S., 1993, *Priests and Programmers: Technologies of Power in the Engineered Landscape of Bali*, Princeton: Princeton University Press.

[Lempert, R. J., D. G. Groves, S. W. Popper and S. C. Bankes, 2006] Lempert, R. J., D. G. Groves, S. W. Popper and S. C. Bankes, 2006, "A General, Analytic Method for Generating Robust Strategies and Narrative Scenarios," *Management Science* **52**(4):514-528.

[Murray-Smith, D. J., 1998] Murray-Smith, D. J., 1998, "Methods for the External Validation of Continuous System Simulation Models," *Mathematical and Computer Modeling of Dynamical Systems* **4**(1):5-31.

[Oberkampf, W. L. and T. G. Trucano, 2002] Oberkampf, W. L. and T. G. Trucano, 2002, "Verification and Validation in Computational Fluid Dynamics," *Progress in Aerospace Sciences* **38**:209-272.

[Oberkampf, W. L., T. G. Trucano and C. Hirsch, 2004] Oberkampf, W. L., T. G. Trucano and C. Hirsch, 2004, "Verification, Validation, and Predictive Capability in Computational Engineering and Physics," *Applied Mechanics Reviews* **57**(5):345-384.

[Palmer, T. N., 2000] Palmer, T. N., 2000, "Predicting Uncertainty in Forecasts of Weather and Climate," *Reports of Progress in Physics* **63**:71-116.

[Pospisil, L., 1963] Pospisil, L., 1963, *The Kapauku Papuans of West New Guinea*, New York: Holt, Rinehart and Winston.

[Pospisil, L., 1972] Pospisil, L., 1972, *Kapauku Papuan Economy*, New Haven, Connecticut: Human Relations Area Files.

[Prelec, D., 2000] Prelec, D., 2000, "Compound Invariant Weighting Functions in Prospect Theory,"in *Choices, Values and Frames*, Kahneman, D. and A. Tversky (Ed.), pp. 67-92, New York: Cambridge University Press.

[Tversky, A. and C. R. Fox, 2000] Tversky, A. and C. R. Fox, 2000, "Weighing Risk and Uncertainty,"in *Choices, Values, and Frames*, Kahneman, D. and A. Tversky (Ed.), pp. 93-117, New York: Cambridge University Press.

[Tversky, A. and D. Kahneman, 1992] Tversky, A. and D. Kahneman, 1992, "Advances in Prospect Theory: Cumulative Representation of Uncertainty," *Journal of Risk and Uncertainty* **5**(4):297-323.

BUSINESS NETWORK TOPOLOGY AND RIGIDITIES IN PRODUCTION

G. CASTAÑEDA,* El Colegio de México, México D.F.
R. CHAVARIN, Universidad de Guadalajara, Jalisco, México

ABSTRACT

An agent-based model is built with the aim of explaining the effect that organizational architecture has on the performance (network size and stability) of a business group. An artificial network is grown where each node (firm) is subject to random income shocks while the degree of flexibility of the productive system (labor marker rigidity and technology's capita-labor ratio) is assumed exogenously. Two variants of network topology are considered: (i) decentralized management where each node makes local decisions and behaves opportunistically with a positive probability when a troubled partner needs to be rescued, and (ii) centralized management where financial transfers from other firm are made with certainty as long as it has available funds, although, the functioning of this topology entails some monitoring cost.

Keywords: Business groups, network topology, economic volatility, labor rigidities

INTRODUCTION

This paper attempts to understand why business networks may transform their topology or organizational architecture when the productive systems becomes less flexible, either because there is an increase in the rigidity of the labor market or because the participation of the firm's fixed costs increases due to an industrialization process based on capital-intensive production. The hypothesis to be theoretically validated here with an agent-based model (ABM) asserts that these rigidities make more cumbersome the operation of a decentralized network where decision-making is entirely local and, thus, entrepreneurs have the incentive to transform the network adopting a centralized architecture.

Decentralization is less problematic in an environment with minor rigidities since a firm (node) can respond by itself to a negative shock in demand by adjusting its variable costs which are a relatively large component of the cost structure. The adjustment in variable costs is mandatory when the financial system exhibits deficiencies that inhibit cash flow smoothness through credit borrowing. On the contrary, a centralized network fosters the use of financial transfers among firms facing uncertain shocks when cost rigidities preclude a solution to the cash flow problem at the level of individual nodes. In spite of the higher monitoring costs, the centralization of a network is advantageous in this economic setting since it can overcome to a large extent the opportunistic behavior that go with decentralization. These costs are paid to keep a stringent control in the allocation of the subsidiaries' cash flow surpluses and in the nodes' management.

The ABM presented in this paper is built having in mind some features of the Mexican experience in the XIX and XX centuries (Castañeda and Chavarín 2007), and the artificial facts generated with it are contrasted with such experience. However, it can be argued that the model is also adequate to explain the transformation of managerial practices and business alliances in any other country that undertook an industrialization process characterized by a steady increase in the rigidity of the productive system and the prevalence of institutional drawbacks that hampered the functioning of financial markets (Leff 1978, Weinstein and Yafeh 1995, Kali 2003, Khanna and Yafeh 2005, Feenstra, Hamilton and Lim 2002).

* *Corresponding author address*: El Colegio de México; Camino al Ajusco No. 20, Pedregal de Santa Teresa, Ciudad de México, 10740 México; (52-55) 5449-3000 ext. 4064; e-mail: sociomatica@hotmail.com

AN AGENT-BASED MODEL

When 'growing' a business network in the sense of Epstein (2006) it is possible to observe tick by tick how rapid the spin-offs expand and how the network responds to increased labor rigidities and higher volatility. In other words, the ABM allows the observer to discover the performance of a business network that changes from a decentralized status to a centralized one after the fixed costs have increased in the economy. Moreover, the simulation of the model with variations in critical parameters make possible to establish precise conditions that are required to validate the hypothesis theoretically.

Methodology

Artificial business groups are grown assuming two different network topologies: (i) a decentralized management where each node can fail to meet its commitment to transfer resources to firms facing financial difficulties, and (ii) a centralized management where an implicit holding structure precludes opportunistic behavior by network members. The performance or macroscopic pattern can be measured in terms of the mean network size, its standard deviation and the time span for the network to collapse.

The model is very simple to avoid excessive parameters that can confuse the understanding on how the aggregate outcomes emerge; therefore, the heuristic rule for surplus nodes in the decentralized network is to rescue troubled firms with a probability $p < 1$ (where $1-p$ is the probability of opportunistic behavior); likewise, the remaining cash flow is used to form a new spin-off that is connected to its parent firm, irrespectively of the amount earned in the previous tick. On the contrary, for centralized management there are monitoring costs that reduce earnings in each of the nodes, and the probability of being opportunistic is equal to zero since surplus firms are required by corporate offices to rescue member firms with a deficit, yet the new spin-offs are also attached to their parent firms.

Income Shocks Due to Changes in Firms' Demands

Starting with a central point in the grid of a torus, in each tick income fluctuates for the existing nodes according to a uniform distribution $U[a, b]$. Since fixed costs (k) are such that $a < k < b$ it is likely that a firm can experience a liquidity problem through its life span, when the opposite happens the node creates a spin-off that becomes a member of the network. Nodes can be differentiated in terms of income volatility; for instance, those firms whose lower-bound income is zero have a higher probability of facing financial difficulties than those nodes where $a > 0$.

Market Congestion and Diseconomies of Scale

Firms' earnings are reduced because of market congestion, that is, with more network members the market has to be split and this precludes the network to grow without bound. Since competition between different networks is not implemented in the model, market congestion can be interpreted as the cannibalization of the clients between subsidiaries, branches or franchises that belong to the same business group. This constrain of network size can also be interpreted as diseconomies of scale that appear in large conglomerates. In the case of a centralized topology, firms' earnings in each tick are also reduced through a monitoring cost that is proportional to firm's income ($0 < m < 1$); presumably, more economic activity in a subsidiary requires more stringent controls by corporate headquarters.

Financial Rescuing of Troubled Firms

The procedure for the rescue operation is as follows: (a) with decentralized management the troubled firm can ask for support to autonomous surplus firms, each of them provide it with probability $p < 1$, thus if there are three surplus firms the probability of not getting any financial support is given by (1-p)(1-p)(1-p) since each node acts locally; (b) with centralized management surplus firms with enough funds have a mandate to help deficit firms that ask for it and hence $p = 1$. Three types of matching between surplus and deficit firms are considered in the model: power-matching, random-matching and accumulated matching. With power-matching the troubled nodes with more ties coming out are rescued first, and surplus nodes with more ties coming out are asked last. This heuristic rule implies that powerful firms (those with more spin-offs) influence network's decisions on who is initially rescued and whose funds are used. However, with random-matching deficit and surplus firms are sorted randomly and, thus, the order of the rescue operation does not depend on any specific feature of the firms or hierarchical consideration.

Because these two types of matching are applied in the centralized and decentralized architectures, the only differences between these topologies have to do with the value of p and whether monitoring costs are relevant or not. In contrast, accumulated-matching is only used by networks with centralized management. Under this heuristic all funds of surplus firms are accumulated in the network's treasury and then used to rescue deficit firms sorted hierarchically, so that firms with more ties coming out are rescued first; therefore, when a centralized architecture adopts an accumulated-matching procedure its performance is compared with a decentralized topology with power-matching. Furthermore, the remaining funds after the rescue operation are used by the contributing firms to create spin-offs regardless of architecture and matching procedure.

Endogeneity of Opportunistic Behavior

In the case of decentralized management the probability of opportunistic behavior is endogenized. Since in each tick there can be a fracture in the network when a node vanishes due to liquidity problems and partners that are not capable or willing to provide financing, the out-coming nodes split from the rest of the group and give birth to different sub-networks. Therefore, the value of 1-p increases with the number of sub-networks to reflect the fact that under this new structure opportunistic behavior is more predominant. This artifact can be interpreted as increased fissures in the business group that make reciprocal funding less likely, the denial of funding can be understood as a decomposition in the social fabric of entrepreneurs involved in a decentralized network.

The Model's Pseudo-Code

A better understanding of the algorithm can be grasped by means of the diagrammatic pseudo-code presented in Figure 1. Notice that the exogenous parameters and procedures (identified with ovals and dashed arrows) that can be modified by the observer are the following: maximum income and degree of volatility that affect the nature of the random shocks; fixed costs and the selection of network topology that influence net earnings in each firm as a consequence of the socioeconomic environment and by making monitoring costs relevant, respectively; this latter selection, the possibility of making endogenous the probability of opportunistic behavior in decentralized networks and the sorting criteria for matching resources affect the rescue operation. The effect of market congestion is another exogenous parameter that is pre-specified in the program yet it is not relevant for the inferences of the model under a wide range of values.

Once the limits of the uniform distribution are specified by the observer, income shocks are seeded randomly in new and old nodes in each tick and, then, the proliferation of nodes determine the level of market congestion and net earnings for each firm. After the production process takes place with the given fixed costs and sales are defined in terms of market congestion, the financial situation of each firm is analyzed. Afterwards, resource matching takes place in terms of the sorting criteria defined by the observer, and in terms of the probability of opportunistic behavior.

Nodes that are rescued stay alive in the network while troubled ones that do not receive support die; under the latter scenario the probability of opportunism increases as a function of the number of sub-networks that the dying nodes generate. Firms with available cash flow net of financial transfers create new spin-offs that depart from the parent node and a new iteration in the program ensues. Finally, when the financial fragility of the nodes is generalized the network collapses and the program stops.

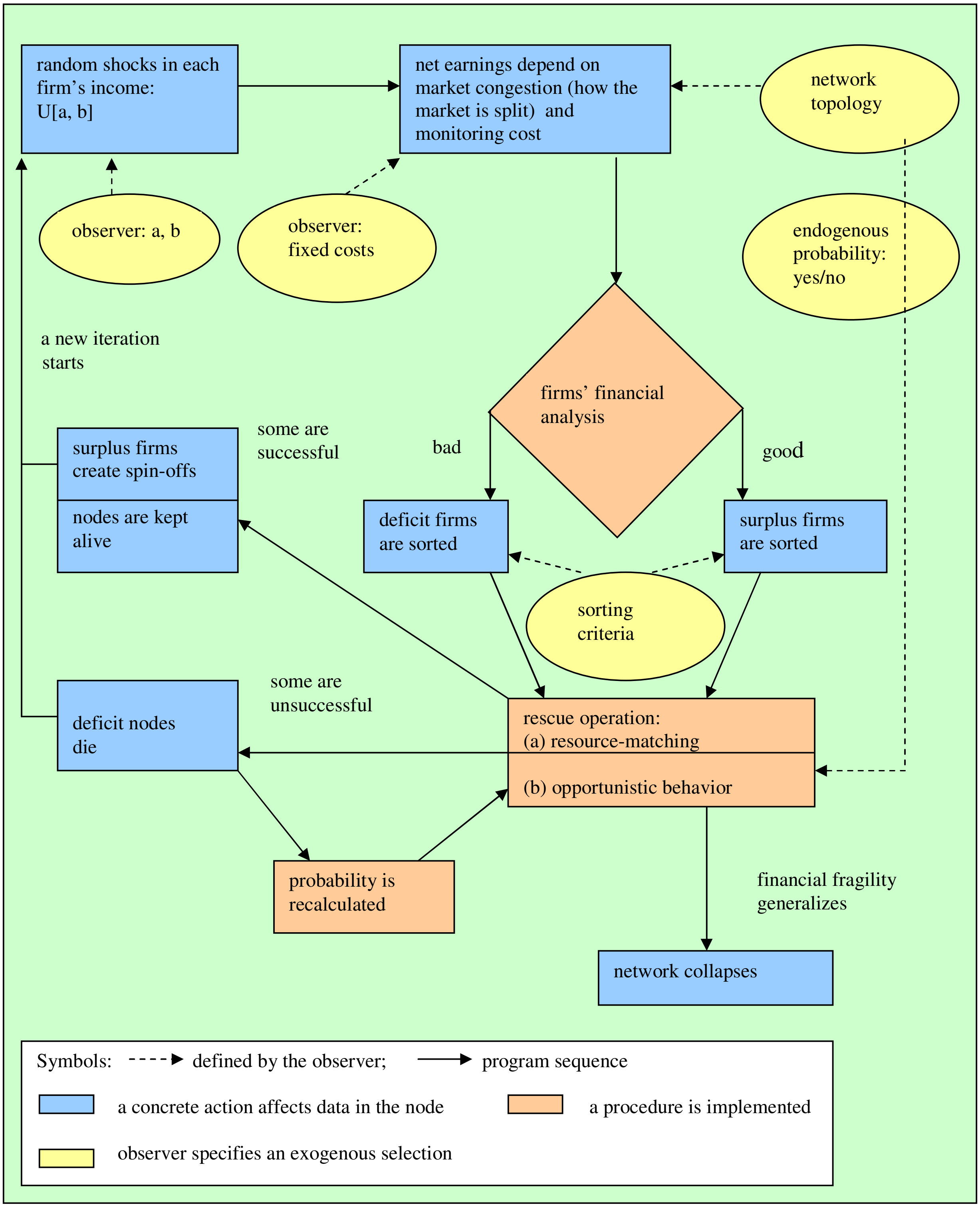


Figure 1 Diagrammatic Pseudo-code

SOME SIMULATIONS

The Main Hypothesis and Variations in Monitoring Costs

This hypothesis asserts that a centralized architecture has a better performance in a post-revolutionary setting (high fixed cost and high income variability) than the decentralized variant, while the opposite holds in a pre-revolutionary environment.[1] Accordingly, for this hypothesis to be theoretically valid the two statements have to be met simultaneously for a range of the monitoring costs in the centralized topology. Figures 2 and 3 present the mean network size, estimated with ten runs, and its confidence intervals for the pre-revolutionary and post-revolutionary settings, respectively. For the point estimates of mean values the decentralized topology performs better in the pre-revolutionary period in the range [0.08, 0.25] while the centralized one has a better performance in the post-revolutionary period in the range [0.05, 0.2]; that is, taken these two intervals jointly it can be argued that the hypothesis holds for the intersection [0.08, 0.2]. Although, the range is shortened to [0.1, 0.15] when considering non-overlapped intervals with 95% of confidence.

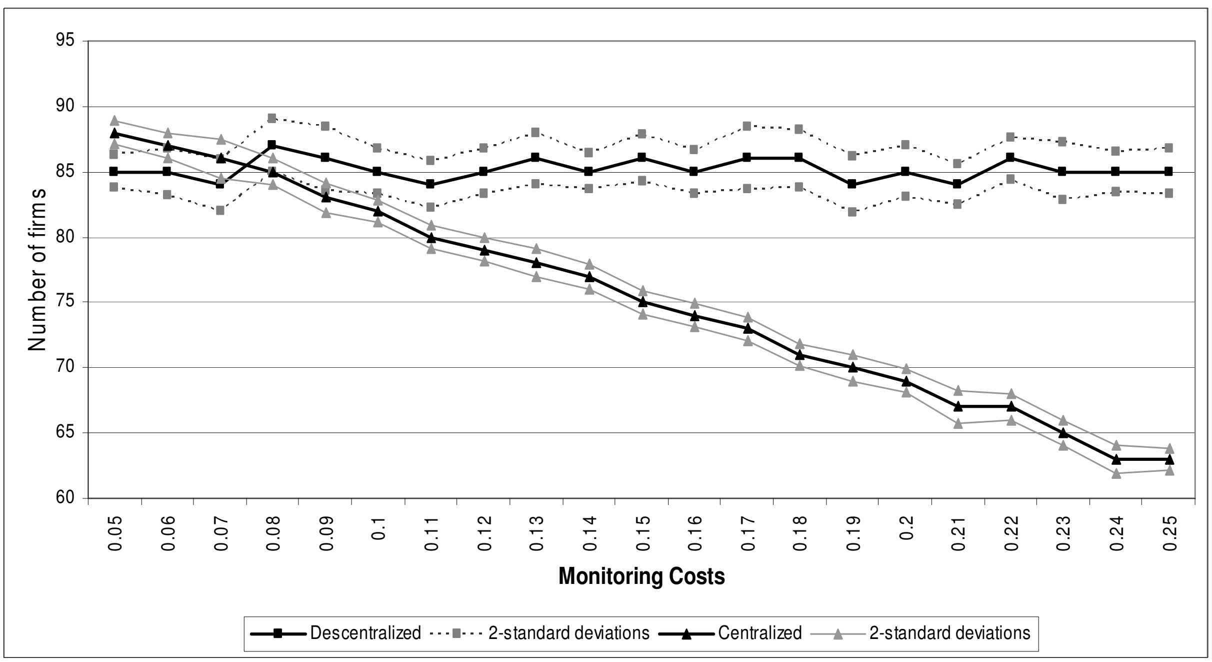


Figure 2 Mean network size for the two topologies in a pre-revolutionary context

Notes: (a) Each simulation is produced with three stages of 50 ticks each. The aim of the first stage is to let the network grow until it stabilizes due to market congestion. The first two stages are run with a decentralized topology and the third stage with a centralized topology, which is the historical sequence of events. (b) A partial mean size for each topology is calculated using the arithmetic means of the last 10 ticks of stages two and three for the series of moving averages with 30 values of the number of firms; then, the final mean size is obtained averaging the partial means of ten repetitions. (c) The intervals of confidence for the average number of firms are calculated with the mean plus-minus two standard deviations. (d) A pre-revolutionary setting is characterized by the following parameters: low fixed costs (k = 3), low volatility (a = 3, i.e. ratio-of-volatility = 1 in the *NetLogo* interface). (e) For the rescue operation firms were sorted using power-matching. (f) The probability of opportunistic behavior is endogenous and starting at 1-p=0.7.

In terms of the second indicator of performance (stability), results not shown in the paper indicate that the coefficients of variation of network size are higher with the decentralized topology

[1] The pre-revolutionary and post-revolutionary terms are chosen in reference to stylized fact of the Mexican history.

in both environments, irrespectively of monitoring costs; nonetheless, the gap between this indicator for the two topologies is much smaller in the pre-revolutionary setting (e.g. approximately 3 times lower). Therefore, in a pre-revolutionary environment network size seems to be a more important issue given the fact that the economy is less volatile and, hence, when this condition changes in the post-revolutionary environment a centralized architecture offers a better performance in terms of both indicators.

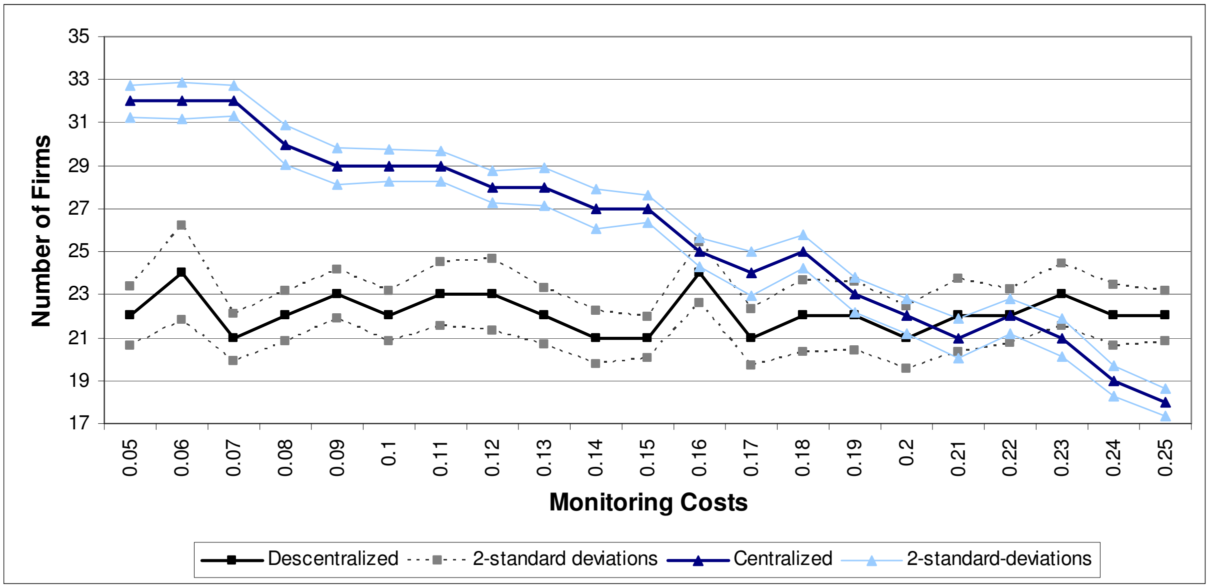


Figure 3 Mean network size for the two topologies in a post-revolutionary context

Notes: (a) A post-revolutionary setting is characterized by the following parameters: high fixed costs (k = 5), high volatility (a = 0, i.e. ratio-of-volatility = 0 in the *NetLogo* interface).

Stability of the Network and Counterfactual Exercise

In the following analysis the stability of each topology is tested in a volatile environment (a = 0) with high rigidities in the productive system (k = 6). The simulations are generated for a value of monitoring costs in the centralized topology which is within the range that validates the hypothesis (m = 0.13), and for an endogenous probability of opportunistic behavior starting at (1-p) = 0.7 in the case of a decentralized variant. An upper limit of 500 is set for the number of ticks and 25 simulations are run for each topology, 11 of them reach this limit when networks are organized with the centralized architecture and none with the decentralized topology. Likewise, the median number of ticks until the network collapses is 239 for the centralized topology and 55 for the decentralized one. The minimum and maximum ticks for these topologies are [13, 202] in the decentralized case and [16, 500] in the centralized case. Accordingly, these results indicate that the centralized architecture becomes critical for the survival of the network in a post-revolutionary environment characterized by important rigidities in the labor market and capital intensive production.

Next, history is re-played by growing a network in a socioeconomic environment that moves from the pre-revolutionary to the post-revolutionary setting in five stages described by the following parameter values [(a=3, k=3), (a=3, k=4), (a=3, k=5), (a=0, k=5) and (a=0, k=5)], each of them lasting 50 ticks. In the first four stages the decentralized architecture operates (with endogenous probability and 1-p=0.7) and, thus, 20 repetitions produce an average network size of

86, 53, 34 and 23,[2] while in the fifth stage the firms transform to a centralized system (with m=0.13) as the stylized facts of history indicate for a an average network size of 27. In contrast, the average size remains in 23 firms when in the fifth stage a counter-factual exercise is simulated with firms sticking to the decentralized topology; therefore, the adoption of a centralized architecture has economic sense in a volatile and rigid environment.

Phase Transition in the Opportunistic Behavior Space

The probability of opportunistic behavior in the decentralized topology (1-p) is modified within the range [0.5, 0.9] to calculate the average number of firms in the pre-revolutionary setting (k=3, a=3). Figure 4 describes the mean network size and its interval of 95% confidence calculated with 10 repetitions for each starting probability value. There is an initial plateau of the average number of firms for the starting values of 0.5 and 0.6, but network sizes drops sharply for a probability of 0.7 and, then, another plateau at a lower level is established for the values of 0.8 and 0.9.

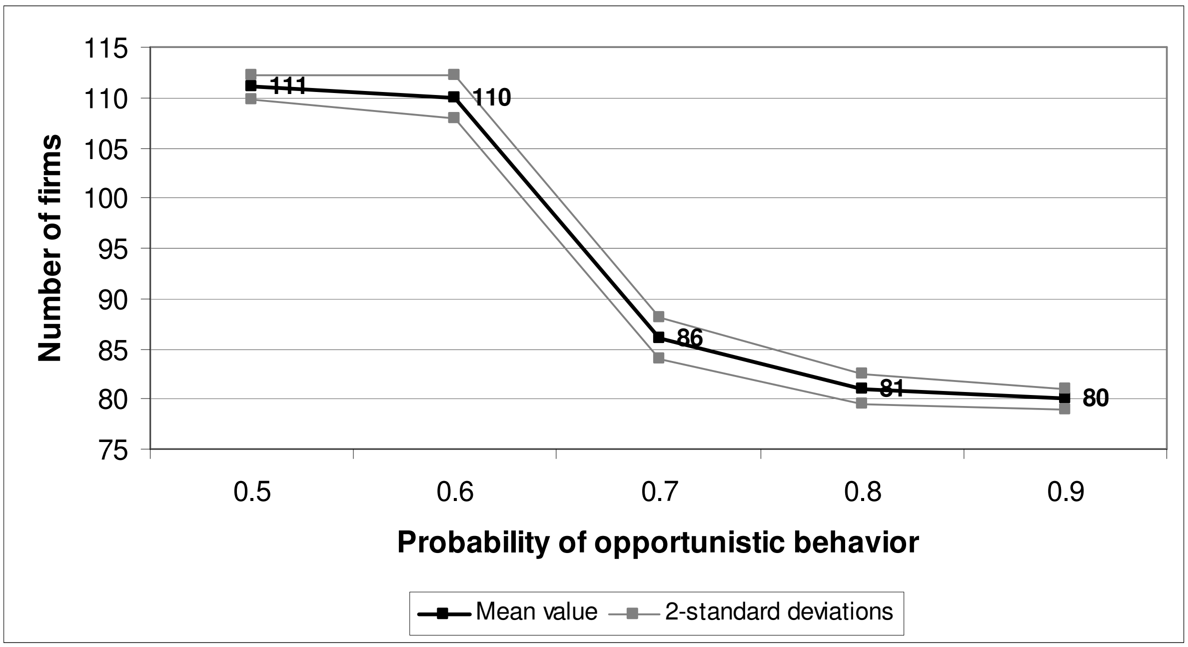


Figure 4 Mean network size for different opportunistic behaviors in a pre-revolutionary setting
Notes: (a) Arithmetic averages are calculated with the last 10 ticks of stages with 50 ticks. (b) The first stage in the simulations considers (1-p) = 0.5, and this probability is increased one decimal points in each of the subsequent four stages. (c) Ten simulations are run and, hence, mean network size for each parameter value is calculated with the mean of the arithmetic average estimated in each stage of the simulation. (d) In the pre-revolutionary setting: k = 3, a = 3.

There is a phase transition when the opportunistic conduct in the rescue operation starts at 0.7 since the emerging patterns in the dynamic system change drastically when crossing this threshold. The modification in the outcome of the model at this critical value is observed by running 10 more simulations in the post-revolutionary environment (a=0, k=5) for each of the two topologies. While for (1-p)=0.6 the average number of firms is 32 for the decentralized topology, this number is 29 for the centralized variant with monitoring cost of 0.13; means that are statistically different. This is the inverse result to the outcome obtained for (1-p) = 0.7 where the corresponding average sizes are 22

[2] First a partial mean size is calculated using the arithmetic mean of the last 10 ticks of the stage for the series of moving averages with 30 values of the number of firms; then, the final mean size is obtained averaging the partial mean size of the different repetitions.

and 28 (also statistically different), respectively. Accordingly, the hypothesis stated in this paper is theoretically validated only when the probability of opportunistic behavior in a decentralized architecture is relatively high, that is, for starting values of 0.7 or more.

Different Types of Matching in the Rescue Operation

Now the robustness of the hypothesis is checked with regard to the type of matching used to rescue troubled firms. Two more alternatives are considered: (i) random-matching where surplus firms rescue deficit firms sorted randomly, and (ii) accumulated-matching where all surplus funds are accumulated in the network's treasury and deficit firms are rescued in terms of their hierarchy (number of ties coming out) until there are no more resources available. In terms of network size the results obtained for power and random matching are practically identical for an scenario with m=0.13 and starting (1-p)=0.7 (see rows 3-6, Table 1). Again network stability is always better with the centralized topology and especially in the post-revolutionary context (see rows 9-12, Table 1).

Furthermore, with accumulated-matching the simulation results validate the hypothesis fairly well for both indicators of performance, network size and stability, in a scenario with m=0.13 and initial (1-p)=0.7. While in the pre-revolutionary environment the decentralized topology outperforms the centralized one with regard to stability (see rows 7 and 8, Table 1), in the post-revolutionary setting the centralized architecture is more stable and also generates larger networks. Notice, however, that the number of firms is statistically identical for both topologies in the pre-revolutionary setting (see rows 1 and 2, Table 1). Statistical results not presented here indicate that using accumulated-matching the hypothesis holds in the intersection interval [0.11, 0.23] for both criteria (network size and stability).

Table1 Network performance (size and stability) for different types of matching

			Pre-revolutionary	**Pre-revolutionary**	**Post-revolutionary**	**Post-revolutionary**
	Type of matching	Type of topology	Average number of firms	Interval of 95% confidence	Average number of firms	Interval of 95% confidence
1	**accumulated**	Decentralized	85	(87.23, 82.77)	23	(24.36, 21.64)
2		Centralized	86	(88.13, 83.87)	32	(33.00, 31.00)
3	**power**	Decentralized	86	(87.94, 84.06)	22	(23.32, 20.68)
4		Centralized	78	(79.08, 76.92)	28	(28.88, 27.12)
5	**random**	Decentralized	86	(87.55, 84.45)	23	(24.30, 21.70)
6		Centralized	78	(78.78, 77.22)	28	(28.49, 27.51)
			Coefficients of variation	Standard deviation	Coefficient of variation	Standard deviation
7	**accumulated**	Decentralized	0.17	0.008	0.33	0.022
8		Centralized	0.37	0.012	0.26	0.013
9	**power**	Decentralized	0.16	0.009	0.34	0.023
10		Centralized	0.12	0.004	0.18	0.008
11	**random**	Decentralized	0.17	0.006	0.34	0.018
12		Centralized	0.11	0.004	0.19	0.010

Notes: (a) The coefficient of variation is measured as the ratio of the standard deviation to the mean. A partial estimation is calculated with an arithmetic mean of the coefficients of variation for the last 10 ticks in the corresponding stage using moving means and moving standard deviations of the last 30 numbers of firms, then, a final estimation is calculated with the average of the partial estimations for 10 repetitions. (b) The probability of opportunistic behavior is endogenous and starting at 1-p=0.7. (c) Monitoring cost in the centralized topology is such that m = 0.13.

CONCLUSIONS

This paper shows that once labor market became very rigid and the industrialization took place in Mexico during the 30's-60's of the XX century there were valid reasons to establish a centralized topology. The high fixed cost and the volatile economic environment made the monitoring costs prevailing in this topology less relevant and, thus, entrepreneurs designed a mechanism to solve automatically the liquidity problems experienced by each node of the network. This situation contrasted with the stylized facts of the 1870-1910 period, where the labor market was flexible and fixed cost relatively small. Under this socioeconomic setting prevailed a decentralized topology where each firm decided independently whether to rescue or not the troubled firms in the group. These facts are consistent with the Monte Carlo simulations produced with an ABM of network formation, at least for some range of monitoring costs and opportunistic behavior.

REFERENCES

Castañeda, G., R. Chavarín, 2007, "The Transformation of Business Networks Due to Increased Rigidities in the Productive System. An Historical Analysis of Mexican Firms," unpublished manuscript, El Colegio de México.

Epstein, J., 2006, *Generative Social Science. Studies in Agent-Based Computational Modeling,* Princeton NJ: Princeton University Press.

Feenstra, R., G. Hamilton and E.M. Lim, 2002, "*Chaebol* and Catastrophe: a New View of Business Groups and their Role in the Korean Financial Crisis," *Asian Economic Paper*, 1 (2); 1-45.

Kali, R., 2003, "Business Groups, the Financial Market and Modernization," *Economics of Transition*. 11 (4).

Khanna, T and Y. Yafeh 2005, "Business Groups and Risk Sharing around the World," *The Journal of Business*, 78: 301–340.

Leff, N.1978, "Industrial Organization and Entrepreneurship in Developing Countries: The Economic Groups," *Economic Development and Cultural Change,* 26: 661-75.

Weinstein, D. and Y. Yafeh 1995, "Japan's Corporate Groups: Collusive or Competitive? An Empirical Investigation of *Keiretsu* Behavior," *Journal of Industrial Economics*, 43: 359-376.

Saturday, November 17, 2007

Social Simulation Applications

Combined Track

Spatial Agents

GEOSPATIAL EXOSKELETONS FOR AUTOMATA IN AGENT-BASED MODELS

P.M. TORRENS,* Arizona State University, Tempe, AZ

ABSTRACT

In this paper, I introduce a new approach to agent-based modeling in geospatial contexts. The novelty of the approach stems from introducing geospatial functionality as an exoskeletal wrapper around standard socio-communicative and goal-oriented agent-based AI. Operationally, I also introduce an integrated and symbiotic tight-coupling to motion capture and Geographic Information Systems, based on space-time Geographic Information Science. To prove the usefulness of the approach in simulation, I describe application of the model to a relatively well-understood (yet widely misrepresented) scenario involving crowd evacuation in constrained infrastructure.

Keywords: Geographic Information Science, geocomputation, geosimulation, multi-agent systems, agent-based models, geographic automata, urban simulation, crowd behavior, complex systems

INTRODUCTION

Geospatial functionality is essential in many agent-automata models. Geography is central to many agent rule-behaviors and is critical in defining agency. Indeed, for multi-agent systems that rely on environmental settings geography invariably occupies a pivotal contextual role in explaining and bounding actions and interactions between the system's constituent agents. There has been a recent surge in agent-based methodological development in Geographic Information Science and gecomputation and a steady growth in the application of agent-based models as experimental toolkits in physical and human geography studies.

Despite the flurry of activity in agent-automata modeling and enthusiasm for their potential in pushing the geographical sciences in new directions, both tool-forging and applied examples have, to some extent, missed opportunities to imbue the research agenda with geospatial science and spatial thinking. Geographers and other social scientists building spatial models have adopted tools and techniques often developed in non-spatial domains with the result that much of their work lacks real geospatial functionality or abstracts from the full geography of the systems or phenomena that they are actually modeling.

Chief among these problems is the use of cellular automata (CA) as agent-based models that require locomotion, even though they are relatively poorly suited to such applications. (As most developers of automata models know, gliders don't really exist outside the retina of the observing model-user (Faith 1998).) In many respects, this is a legacy of automata use in geographical models that was dominated by cellular models and a natural alliance between cells and digital geographic data structures and models based around rasters and raster-landscapes. Similarly, the first ABM development efforts in the geographical sciences were anchored in Geographical Information Systems that sported a slew of code libraries and common object models for reconciling raster layers over time, but treated dynamic vector models comparatively anemically. Similarly, spatial analysis is replete with methodology and algorithms for reconciling the composition and configuration of rasters through relatively straightforward schemes based on "map algebra". (These pattern-matching methodologies are

---

* Corresponding author address: Dr. Paul M. Torrens, School of Geographical Sciences, Arizona State University, Box 875302, Tempe, AZ 85287-5302; e-mail torrens at geosimulation dot com.

hugely popular in validating geographical models, despite widely-acknowledged problems and fallacies in their use.) Raster-based techniques and skills are standard in a geographer's skill-set, while relatively more complicated calculus for vectors and computational geometry is often not as well-developed. Geographers' (and Geographic Information Scientists in particular) natural comfort with conceptualizing systems in cartographic terms and the standard training for social-scientific geographers that often emphasizes qualitative methods and multivariate statistics over approaches like artificial intelligence (AI) or computable heuristics also contribute to these difficulties.

The unfortunate situation that ABM modelers in geography are left with is that physics-inspired random walks and potential-field-following heuristics dominate geospatial models, ignoring over one hundred years of behavioral geography cataloging the myriad ways in which people differ from particles in fluid-flows or mobile pebbles under the sway of gravity or their own kinetic energy. Automata-based modeling in geography, having enjoyed a period of infancy in its development up until relatively recently, has not had to face these issues. However, as the methodological and applied research agendas of agent-based modeling and the geographical sciences have grown more closely aligned, the inability of the tools that are available to answer the questions to which we would like to deploy them has become quite problematic.

In this paper, I introduce a novel approach to agent-based modeling by wrapping agents in a geospatial exoskeleton that affords them geospatial AI for their actions and interactions. (I refer to this as an exoskeleton because we leave certain core agent-based functionality intact, essentially “geospatializing” it via exoskeletal interfaces.) Semantically, the agent-automata that I deploy become geographic automata and the systems they form are better thought of as geographic automata systems. There is more to the scheme than semantic nuances, however. The conceptual foundation for the model is steeped in behavioral and urban geography theory. Mechanically, our modeling methodology is integrally bound to geosimulation, geocomputation, Geographic Information Systems, space-time analytics, and geovisulization. Introducing geographic agency from first principles in this way produces fantastic insight into the space-time processes driving even a relatively simple system such as crowd movement.

GEOSIMULATION AND GEOGRAPHIC AUTOMATA

Geosimulation (Benenson and Torrens 2004) sits in the background for the work that I will present, offering distinct advantages over conventional approaches. First, the traditional consideration of average and spatially-modifiable geographical units or (statistically) mean individuals is replaced in geosimulation. Instead, units are regarded as spatially non-modifiable entities, with individual descriptions and independent functionality. Where aggregates are considered, they are formulated generatively, built from the bottom up by assembling individual entities for the purposes of accomplishing an aggregate task or amassing an aggregate structure. Second, simulated entities are independent and autonomous in their geospatial behavior, turning attention to specification of individual-level behaviors when model-building. Entity behavior is not necessarily treated as spatially or spatiotemporally homogenous across the system being considered. Third, models are commonly designed as event-driven, rather than time-driven, and are built with packets of spatiotemporal change based on the independent internal clocks of simulated components. When put together to form a system, update of these clocks may be flexibly-defined and the methodology can reconcile diverse spatiotemporal scales.

The idea of geosimulation has caught-on in agent-based modeling, thanks in part to development of operational toolkits that allow people to build their own geosimulation models (Benenson and Torrens 2005), or to use of geosimulation as an interfacing mechanism (Bernard et al. 2002). In addition to its use in dynamic Geographic Information Science (Albrecht 2005), agent-based modelers in computer animation (Ali and Moulin 2005; Moulin et al. 2004), social geography (Koch 2003), location-allocation modeling (Ligmann-Zielinska et al. 2005), machine learning and data-mining (Filho et al. 2004), human-computer interaction (Furtado et al. 2007; Furtado and Vasconcelos 2007), criminology (Melo et al. 2006), and medical epidemiology (Ward et al. 2007) have developed geosimulation models. Geosimulation has also been used to extend neighborhood functionality for CA methodology (Zhao and Murayama 2007). The work reported in this paper is part of our efforts to develop more core geospatial functionality for geosimulation-based agent models.

Operationally, we use automata as the vehicle for geosimulation. We have developed a scheme for building Geographic Automata Systems (GAS), fusing the computational properties of automata with Geographic Information Science (GISci) functionality (Torrens and Benenson 2005). GAS are, fundamentally, automata and retain components from cellular automata and intelligent agents. We add the ability to express spatiotemporal relationships based on the full range of spatial analysis routines available in GISci. Geographic Automata (GA) may be located by any geo-referencing convention (L) and can also move through the spaces they occupy by any locomotion regime (R_L). L allow GA to be registered in space and time (i.e. $L = L_t$), either directly or indirectly, on a one-to-one, one-to-many, many-to-one, or many-to-many basis. A typology or ontology (K) of GA entities mediates the nature of L and R_L. Input to GA is considered *geographically* as neighborhoods of interaction and influence. GA neighborhoods may differ heterogeneously in extent and shape per GA and may change dynamically so that $N = N_t$. Neighborhood rules (R_N) determine these changes. Neighborhood relationships, expressed as cognitive filters, social relationships, lines-of-sight, and so on can be introduced and allowed to vary over space and time. Many GA (G below) may be combined in a systems context, with each GA in the collective GAS coded heterogeneously:

$$G \sim (K, S, R_S, L, R_L, N, R_N), \text{ where } R_S: S_t \rightarrow S_{t+1}\,;\ R_L: L_t \rightarrow L_{t+1}\,;\ \text{and } R_N: N_t \rightarrow N_{t+1}\,.$$

Among other things, this allows us to distinguish agent types based on their geospatial behavior. In the models to be described, five *entity types* (classes in an object-oriented sense) are used: World, FixedObjects, MobileObjects, Goals, and Probes. World represents the simulated environment (a city). It subsumes (contains, encapsulates) all other entities in the simulation. It also acts as a template for georeferencing, handling entities' absolute position in the World and their position relative to other entities in the World. FixedObjects is used to represent the urban infrastructure and thus handles buildings and obstacles (parked cars, trees). FixedObjects do not move, although they can influence the movement of other mobile entities. MobileObjects represent people in isolation, crowds and the groups that they form, as well as particles (smoke, embers, mobile toxins). MobileObjects move, as the nomenclature suggests. Goals are used to structure events. They are used as space-time anchors, operating as beacons for actions to be executed. Goals may be used by specific groups or individuals, in specific places and/or at specific times. Probes function as data-collecting entities that linger in the model world with the intent to capture attributes of the simulation (as an executable computer program), the simulated system (as a synthetic representation of phenomena of interest or under study), and model entities' states and actions. Probes are endowed with the ability to sift through data, sort it, and exchange it with caches for input/output to/from GIS as well as spatial and statistical analysis.

GEOSPATIAL FUNCTIONALITY FOR A GEOSPATIAL MODEL

The next step is to imbue this modeling scaffold with theory-driven heuristics. The sub-field of behavioral geography is replete with theory and explanation for human geospatial behavior. We take our cues, in developing geospatial functionality for our modeled agents, from decades of work in behavioral geography.

First, we acknowledge that geospatial behavior is largely determined by heterogeneous geospatial traits per-agent. While seeming to behave with universal behavior at a macroscopic scale of observation, people move and navigate through urban settings with a great deal of individuality and their movement behavior is formed heterogeneously from independent geospatial and geotemporal characteristics. Infrastructure characteristics are established and assigned to geometry in GIS. Pedestrian agents are endowed with characteristics from a synthetic data population, using statistical, geostatistical, and geodemographic inference to down-scale aggregate data sources to micro-levels. Motion capture and motion editing are used to produce realistic-looking movement, heterogeneously, per-agent. The technique also provides upper- and lower-bounds for pedestrian velocity, as well as free speed. This allows us to calibrate rates of acceleration/deceleration per agent, accounting for differences in habits, gait and allows us to encode body language into simulations geometrically.

Second, we base our modeled agents on an assumption that they plan their paths geographically and use waypoints to develop a general sense of how to get somewhere. Path planning is introduced to the modeling framework per-agent as a low-level search heuristic that determines (graph-based) nodal waypoints through which agents then ambulate using a second, higher-level way-finding heuristic, whereby agents plan their route between waypoints before mobilizing.

Third, we wrap agents with spatial cognition as part of their behavioral AI. A traditional automata-based socio-communicative and goal-oriented agent AI sits at the core of modeled entities' behaviors, but those routines are passed through a second layer of geospatial AI, thereby dictating when, where, and in what spatiotemporal contexts that functionality should be employed. Agent walkers are endowed with an individual-centric geography around themselves, used to filter the world as they move. This is formed as a vision cone centered on the agent and cast (as a ray) in a forward direction. Cone properties vary heterogeneously and spatiotemporally based on changes in pedestrians' behavior, characteristics, and surroundings. Potential collisions are registered in an array and sorted for relevance (angry dogs may get priority for some people, but not others, for example). Once free from encounters, pedestrians navigate to return to their shortest path route.

Fourth, we make use of the fact that people's activities strongly structure their use of space and time. Walkers may organize their paths through space-time as events fashioned around their activity goals and these serve as spatiotemporal anchors. We have developed an integrated GIS-based analytical toolkit for sweeping the parameter-space of simulations and for registering simulations to real-world conditions, based on space-time paths and prisms for individual, dyad, and group behavior and isochrones for individuals, groups, and events.

AN APPLIED EXAMPLE

I will now demonstrate the usefulness of this approach through discussion of the model's application to evacuation of agents through confined urban infrastructure. The physics of such effects have been well-modeled on the basis of association between the physical properties of crowd evacuation in such contexts and those of fluid and excitable media in granular or gas-kinetic environments (Helbing et al. 2000). However, true behavioral models have not been developed to model such situations, save a "social force" extension of Newtonian dynamics.

The simulation is set-up as follows. Synthetic pedestrians were arbitrarily loaded to an urban scene and run with innate behavior. Motion capture data dictate their free, upper, and lower speeds and acceleration/deceleration (figure 1). They possess one higher-level driver, an impulse to evacuate from their seed position to a goal at the end of the modeled world. Only one exit exists.

Time t　　　　Time t +1 second

FIGURE 1. Velocity data are derived from motion capture of a real-world actor's movement.

An A* algorithm is run to find the shortest route to that exit for each pedestrian. Pedestrians will follow this track unless their innate behavior dictates otherwise. At the onset of the simulation, pedestrians orient themselves in the direction of the exit. Taking a prompt from higher-level behavior (i.e., to proceed calmly to the exit, or flee at all costs), they quickly scale their velocity from a position of rest to their free speed, navigating, path-planning, and mobilizing to an assembly point on the other side of the exit (figure 2). Some pedestrians have line-of-sight to the destination, while others must steer clear of infrastructural objects (walls) before they can see the goal.

Agents that make it to the exit early are able to evacuate the enclosed space relatively freely. However, a bottleneck soon forms at the mouth of the exit. As the crowd of pedestrians builds up behind and to the sides of this obstruction, a characteristic arch forms at the entry to the exit corridor, which impedes escape further. The crowd begins to wedge at ± 45 degree angles on either side of the exit. This is a well-known emergent property (Helbing et al. 2001).

Probes continually report pedestrians' positions. Those data are run through a space-time GIS that builds space-time paths as graphics and geometry, allowing the parameter-space of the simulation to be swept. As seen in figure 3, Agent 1 exits (the space-time path becomes a vertical, indicating rest) quickly, following the shortest path relatively free of collisions. Agent 2 encounters crowded conditions, however, taking longer to exit. Agents 3 and 4 proceed well until their geospatial AI is impeded by a jam toward the end of their space-time path. Compaction and expansion of their space-time paths is evident, illustrating frustrated cycles of speeding-up and slowing-down. These signatures are similar to those in car traffic jams following accidents (Nagel and Schrekenberg 1995). The onset of a traffic jam is clear (figure 2c, 3a). The slope of the space-time path of individual walkers rises sharply as pedestrians enter the exit corridor, leveling out somewhat thereafter (figure 3).

Mapping textures to simulated pedestrians (cloth, hair) and infrastructure (concrete), lighting, and shadow-mapping is useful in generating realistic-looking simulations (figure 2). This is particularly important in conveying the themes of modeled scenarios to viewers of the simulation. Buildings may be rendered as transparent or reflective so that the dynamics of crowd flow can be viewed through dense simulated infrastructure as the simulation runs.

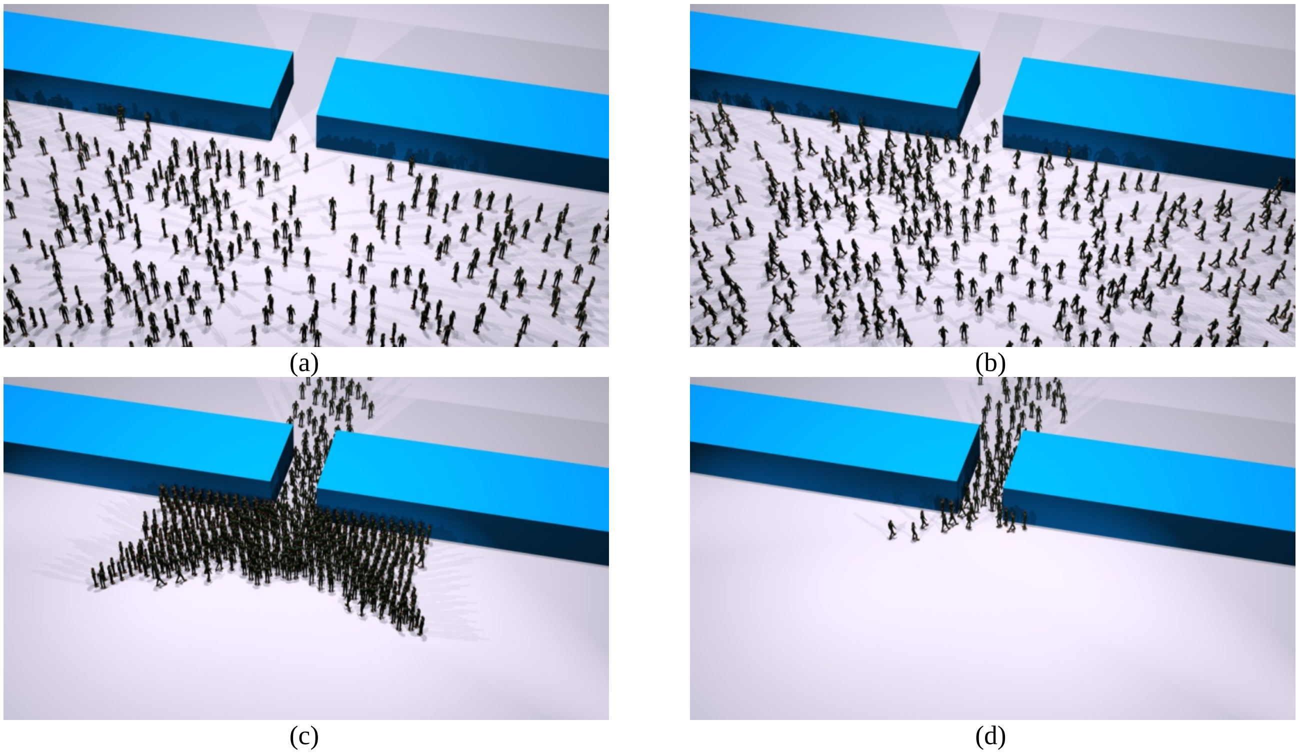


FIGURE 2. (a) Seed conditions. (b) Nearby agents exit quickly (t = 3 seconds). (c) Gridlock forms for agents that arrive at the exit later; pressure builds as the crowd compacts (t = 33 seconds). Characteristic lateral wedges of jammed agents form at the exit's sides, further obfuscating evacuation (t = 66 seconds). (d) As the pressure in the congested crowd subsides, evacuation proceeds more efficiently (t = 133 seconds.

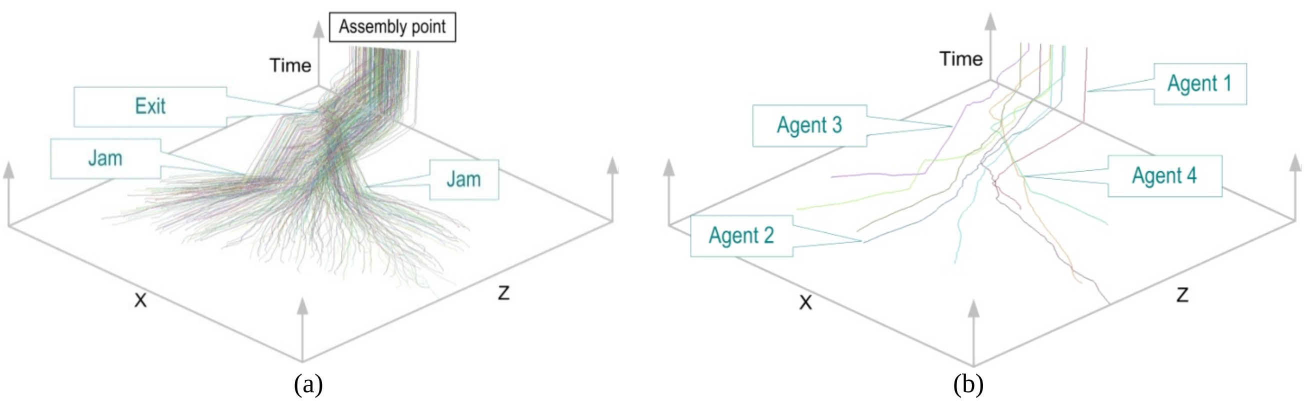


FIGURE 3. (a) Space-time paths for all agents. Spatial movement is shown in (x,z), temporal movement in y. (b) Agent 1 evacuates quickly and easily, as shown by the relatively straight and flat space-time profile. Agent 3 encounters some traffic over the last two-thirds of her journey. Agent 3 evacuates relatively slowly, while agent 4 has a difficult evacuation, as shown by the tortuous and steep space-time path of her movement.

CONCLUSIONS

A demonstration of the usefulness of behaviorally-driven models and analyses in developing realistic and intelligent synthetic representations of geospatial behavior in urban environments has been shown in this paper. The advantages of this approach have been illustrated through application of the scheme to evaluation of large-scale evacuation dynamics in downtown settings. The scheme that has been described is valuable in proving the underlying computation, but also demonstrates the potential for approaches of this kind in exploring and generating theory in studying spatial cognition, sociality, collective behavior, and human-environment interaction.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 0624208 from the Human and Social Dynamics program and Grant No. 0643322 from the Geography and Regional Science and the Methods, Measurement, and Statistics programs, and support from Autodesk, Inc. and Alias Research.

REFERENCES

Albrecht, J. 2005. A New Age for Geosimulation. *Transactions in Geographic Information Science* 9 (4):451-454.

Ali, W., and B. Moulin. 2005. 2D-3D Multiagent GeoSimulation with knowledge-based agents of customers' shopping behavior in a shopping mall. Paper read at Conference on Spatial Information Theory, at Ellicottville, NY.

Benenson, I., and P. M. Torrens. 2004. *Geosimulation: Automata-Based Modeling of Urban Phenomena*. London: John Wiley & Sons.

———. 2005. A minimal prototype for integrating GIS and geographic simulation through Geographic Automata Systems. In *GeoDynamics*, eds. P. Atkinson, G. Foody, S. Darby and F. Wu, 347-369. Florida: CRC Press.

Bernard, L., I. Simonis, and A. Wytzisk. 2002. Dynamic interoperable geoprocessing and geosimulation: An OpenGIS/HLA based interoperability architecture. Paper read at European Interoperability Workshop, June 24 to 26, at London.

Faith, J. 1998. Why Gliders Don't Exist: Anti-Reductionism and Emergence. In *Artificial Life VI: Proceedings of the Sixth International Conference on Artificial Life*, ed. C. Adami. Cambridge, MA: MIT Press.

Filho, E. V., V. Pinheiro, and V. Furtado. 2004. Mining data and providing explanation to improve learning in geosimulation. In *Lecture notes in computer science 3220: Intelligent Tutoring Systems*, eds. R. M. Vicari and F. Paraguaçu, 821-823. Berlin: Springer.

Furtado, E., V. Furtado, and E. Vasconcelos. 2007. A conceptual framework for the design and evaluation of affective usability in educational geosimulation systems. In *Lecture Notes in Computer Science 4662: Human-Computer Interaction (INTERACT 2007)*, eds. M. C. C. Baranauskas, P. A. Palanque, J. Abascal and S. D. J. Barbosa, 497-510. Berlin: Springer.

Furtado, V., and E. Vasconcelos. 2007. Geosimulation in education: A system for teaching police resource allocation. *International Journal of Artificial Intelligence in Education* 17 (1):57-81.

Helbing, D., F. Illés, and T. Vicsek. 2000. Simulating dynamical features of escape panic. *Nature* 407 (September 28):487-490.

Helbing, D., P. Molnár, I. Farkas, and K. Bolay. 2001. Self-organizing pedestrian movement. *Environment and Planning B* 28:361-383.

Koch, A. 2003. Sozialgeographische agentenbasierte geosimulation: Zur komplementarität von raumsemantik und raummodell. In *Klagenfurter Geographische Schriften 23: Multi-Agenten-Systeme in der Geographie*, eds. A. Koch and P. Mandl, 35-64. Klagenfurt: Institut für Geographie und Regionalforschung der Universität Klagenfurt.

Ligmann-Zielinska, A., R. L. Church, and P. Jankowski. 2005. Exploring multiobjective urban land allocation with geosimulation: Are optimized spatial alternatives doable in practice? Paper read at Geocomputation 2005, at Ann Arbor, Michigan.

Melo, A., R. Menezes, V. Furtado, and A. L. V. Coelho. 2006. Self-organized and social models of criminal activity in urban environments. In *Lecture Notes in Computer Science 4150: Ant Colony Optimization and Swarm Intelligence*, eds. M. Dorigo, L. M. Gambardella, M. Birattari, A. Martinoli, R. Poli and T. Stützle, 518-519. Berlin: Springer.

Moulin, B., W. Chaker, and J. Gancet. 2004. PADI-Simul: an agent-based geosimulation software supporting the design of geographic spaces. *Computers, Environment and Urban Systems* 28 (4):387-420.

Nagel, K., and M. Schrekenberg. 1995. Traffic jams in stochastic cellular automata. Los Alamos, NM: Los Alamos National Laboratory.

Torrens, P. M., and I. Benenson. 2005. Geographic Automata Systems. *International Journal of Geographical Information Science* 19 (4):385-412.

Ward, M. P., S. W. Laffan, and L. D. Highfield. 2007. The potential role of wild and feral animals as reservoirs of foot-and-mouth disease. *Preventive Veterinary Medicine* 80:9-23.

Zhao, Y., and Y. Murayama. 2007. A new method to model neighborhood interaction in cellular automata-based urban geosimulation. In *Lecture Notes in Computer Science 4488: Computational Science (ICCS 2007)*, eds. Y. Shi, G. D. Albada and J. Dongarra, 550-557. Berlin: Springer.

ENACTMENT SOFTWARE: SPATIAL DESIGNS USING AGENT- BASED MODELS

T. NARAHARA*, Harvard University, Cambridge, MA

ABSTRACT

This paper introduces a potential use of agent-based computational models into architectural spatial design strategies. Visualization of the human behaviors within architectural spaces is one of the uncultivated areas in the field of architectural design. The use of agent-based computational tools such as NetLogo has a great potential to overcome various shortcomings in the existing architectural design software.

Keywords: Architectural Design, Visualization, Agent-based Computation, NetLogo, CAD software, Image Rendering, Human Behaviors.

INTRODUCTION

Spatial qualities in architectural design cannot be fully evaluated solely by observing geometrical constructs without reference to inhabitants placed inside. However, imagining what happens to those inhabitants and appreciating their movement is difficult even for trained architects. Architects tend to underestimate the importance of the role of people inside their buildings; they are often not fully aware of the behaviors induced by the spaces which they design. The existing analytical means of architectural representations – plans, sections, elevations, axonometric, and perspective – are not sufficiently capable of visualizing the psychological behaviors of people.

Behavioral aspects of the spatial design have yet to be addressed well in any existing architectural design tools. As architectural projects become increasingly more complex in their formal manifestation as well as functional requirements, new methods are sought to address the complexities. Contemporary designs by today's leading architects are often filled with their signature expressions, and some of their design decisions seem to be executed based on their individual sensitivities and intuitions without enough concern for behavioral aspects of users. Today's advanced computational design tools (CAD software) can produce complex forms and sophisticated visualizations of light, materials and geometry. But they are not suitable for helping people to quickly study and understand a spatial design as it would be inhabited. The proposed method lays a foundation for developing a new kind of software that overcomes this shortcoming by the use of agent-based computation.

In architecture, many spatial problems are indeterministic. Solutions to particular design problems are usually constrained by multiple criteria and are far from obvious. Normally, architectural design has multiple evaluation factors, and this fact prevents it from having simple

* *Corresponding author address*: Taro Narahara, Harvard University, Graduate School of Design, 48 Quincy Street, MA 02138; e-mail: narahara@gsd.harvard.edu

straightforward equations or methods to reach solutions. Moreover, this ambiguity and invisibility of spatial qualities tends to hide the actual problems from designers' eyes. Aesthetic values may scarcely ever be quantified or objectively evaluated, but even mere functional aspects of architecture are hard to evaluate. For instance, architectural codes respond to constituent parts of dimensional constraint in architectural spaces. However, for a building as a whole, whether or not the end products of assembly of all locally constrained parts are efficient and comfortable, are very difficult to predict before actual user involvement. A "walking actual human" on full-scale mock-up can be one solution to check the level of comfort in architecture although time and cost that it takes to execute it is considerable. Another possible approach can be using a synthetic figure based on human behavioral patterns to test the performance of space before realization of the physical space.

The paper seeks to find a tool with which to capture human reactions in architectural spaces in an animated format. By suggesting another layer of architectural quality hidden behind the constructive forms, this paper aims to bring the designers' attention back to a man on the stage, and the eyes of a user. By walking a synthetic figure through architectural space, this tool will visualize the psychological response to architectural elements in motion.

The main difference between conventional architectural representations and this tool lies in the presence of autonomous behaviors embedded in the figures. Through agent-based computation, it moves inside the model and displays various behaviors in reaction to spatial characteristics such as transparent surface, opaque surface, perforation and furniture. Typically all the figures in architectural animations are controlled, prepared, and inserted afterward, on top of their 3D models in a top-down manner. For example, most of the walking figures merely follow the paths that were drawn and reinterpreted by the designer on their 3D models, and their motions are post-rationalized by the creator (animator) of the presentations. There are some levels of discontinuity between the motions of the figures and their surrounding architectural environments since they are results of reinterpretations by the observers who reside outside the environments. This new tool attempts to omit this final tweaking process by according some level of cognition to these figures from the outset of the process. These cognitive capabilities include collision detection, obstacle (walls, etc.) detection, cognition among others, and cognition of the attractors in architectural environments.

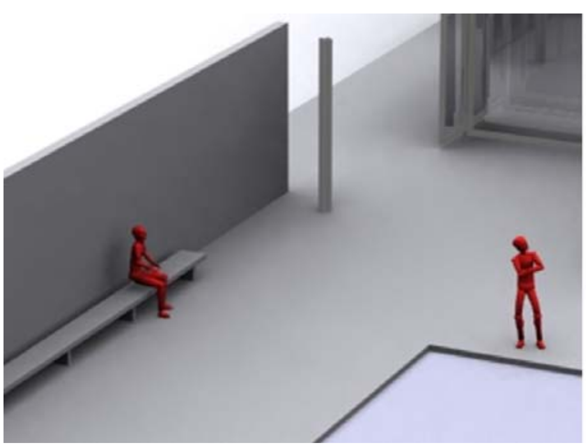
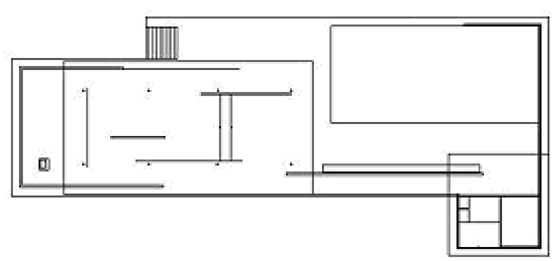



FIGURE 1. Proposed Tool (left), and Conventional design communication tools (right).

Re-Action to the architectural spaces, including the reaction to the different materiality of the architectural elements, is one of the most unique features implemented in this tool. These cognitions are directly embedded in the figures on the stage sets, and they are not from the re-interpretation or the post-rationalization.

METHODOLOGY

The proposed method uses two platforms; the agent-based environment which computes the behaviors of each figure, and the 3-D visualization environment inside a rendering application. Results from the agent-based computation will define all the behaviors and movements of the figures. In other words, all the intelligence about the autonomous behaviors is acquired from the simulation program developed by the author in the programming environment in NetLogo (Wilensky, 1999.) In order to achieve physical three dimensional qualities in representation, all these behavioral information will be once translated (encoded) into text files to export the information into external visualization environment. 3d Studio Max release 8 is selected for the final visual representational platform due to its high-end rendering capabilities, and its availability among the architects and students. Furthermore 3d Studio Max release 8 comes with CharacterStudio which provides standard bipedal character skeletons. This biped figure is used as an actor inside the Space Re-Actor's environment based on the results acquired form the agent-based computation.

In the field of architectural design profession, it is fairly typical for designers to build 3-d models of their own building designs within the CAD (computer aided design) software environments such as AutoCAD, 3d Studio MAX. The proposed method introduces walking scale figures in geometric models. The goal is to add a sense of place to the geometry, and augment the representation of its spatial quality for designers and audience. Through agent-based computation using NetLogo, they move inside the model and display various behaviors in reaction to spatial characteristics such as transparent surface, opaque surface, perforation and furniture.

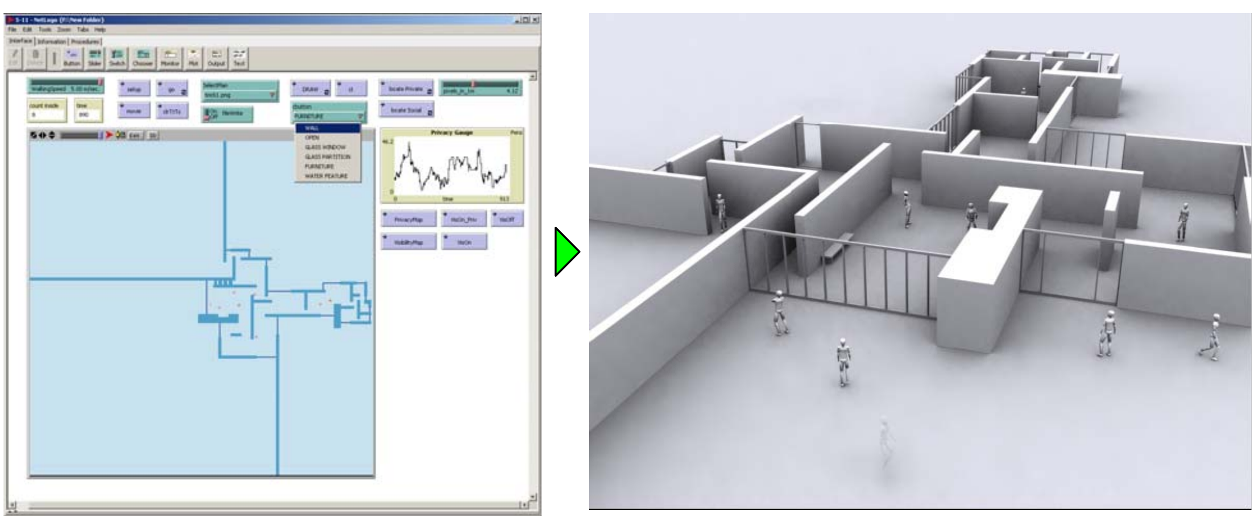


FIGURE 2. The Agent-based Environment in NetLogo (left), and the 3-D Visualization (right)

The figure is assigned a psychological profile with a different degree of sociability, and reacts to proximity and visibility of others in the same model. In order to provide cognitive and behavioral capabilities to the figures in the agent-based environment, individuals are equipped with series of internal variables such as vision, memory, energy, level of curiosity toward attractors, sociability, and so on. NetLogo's programming language is used to describe various architectural elements such as walls, glazing, doors, furniture, water feature, and so on as attractors which define the behavioral responses from agents on scenes. NetLogo's pixel-based graphic environment allows users to import any architectural floor plans' information from CAD software environments, and users can color-code preceding various attractors on their floor plans in real-time manner. Furthermore, computational results from NetLogo program can be exported into architectural animation software's 3-D environment using text codes indicating every agents coordinates information and series of ASCII codes representing agents' behavioral responses which are associated with proper motion capture files for bipedal character skeletons for cinematic representational purposes.

Series of built and un-built works by Mies van der Rohe are selected for the case studies. Mies is regarded as one of the pioneer of modern architecture and left series of minimalistically composed plans for his projects. The spaces designed by Mies are relatively loosely defined in terms of the programs and the specific usages of each individual spatial compartment, thus results of human responses are less predictable, and they are considered to be suited for the study. The German Pavilion at the International Exposition in Barcelona (1928-1929) by Mies, usually referred to as the "Barcelona Pavilion" is one building on which the functionality of this tool, the Space Re-Actor, will be tested. As a building type, a pavilion does not force any specific objective on the people visiting the building. Visitors' behaviors will be in direct response to the building's architectural features such as water features, benches, and sculptures, as well as to the transparency, opaqueness, or texture of the surface materials. They begin by wandering around the building and gradually find their ways toward what naturally attracts them. The visitors' behaviors are induced by the architectural elements present, and this condition is well suited for solely concentrating on evaluating the spatial conditions.

The followings are the examples of various reactions to architectural elements implemented in the synthetic figures. The tool also allows users to sketch and color-code the architectural elements to assign different qualities and characteristics to the materiality and features intrinsic to architectural spaces in real time manner.

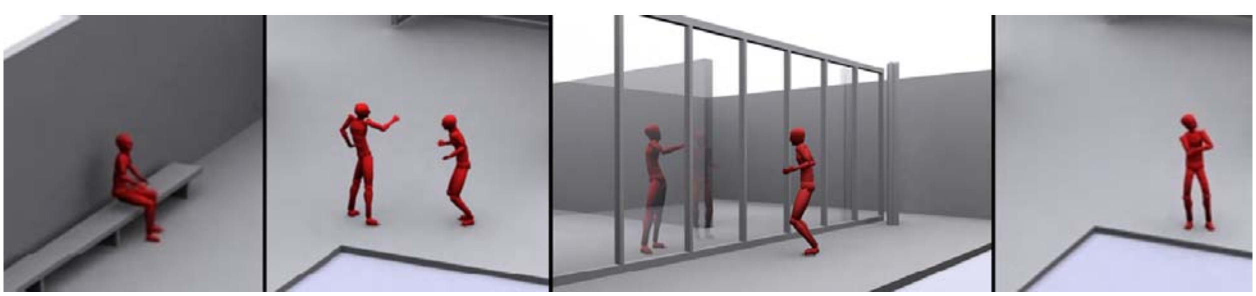


FIGURE 3. Re-Actions to Architectural Elements. Interactions among Agents. Series of Re-Actions gained from NetLogo are associated with various behavioral motion captured files.

• Attractor-based Reaction: users can assign attractors such as sculptures or water features. They attract visitors' attention, and the visitors may halt for a moment to enjoy the views. For instance, the Barcelona Pavilion has two water ponds as major architectural features. Spatial allocation of these elements inside the overall composition by definition reflects the master designer's intention and motivation to shape people's circulation and experience. How the specific placement of those attraction features can manipulate the visitors' behaviors and contribute to the overall experience of the pavilion are the questions to be explored by seeing the real-time reactions of walking figures.

• Agents' Variable-dependent Reaction: Depending on a visitor's activity level, she or he might find comfort by sitting on the benches. Figures have an internal variable to measure their activity level (energy level) to check whether they feel like sitting or not. This type of attractor will be conditionally applied based on the internal variables of the figures. For instance, if the energy level is lower than a certain threshold value, they will start to seek the place to rest (furniture: Bench). After regaining the energy, they will go back to respond with the normal level of curiosity to other attractors.

• Visibility-based Reaction: "Can see and can be seen" is an important concept for the behavior of figures. The synthetic figures have vision, and some of their reactions are based on visibility. Better visibility allows them to find not only the attractors but also the others inside the space who could influence their next moves. The tool calculates the visibility at every coordinate point of the accessible space.

• Agents-to-Agents Reaction: Besides the simple avoidance and collision detections, interactions between the agents are considered. Having different degrees of sociability for figures, the tool can start to render an event in the scene. Heterogeneity among the agents governs the

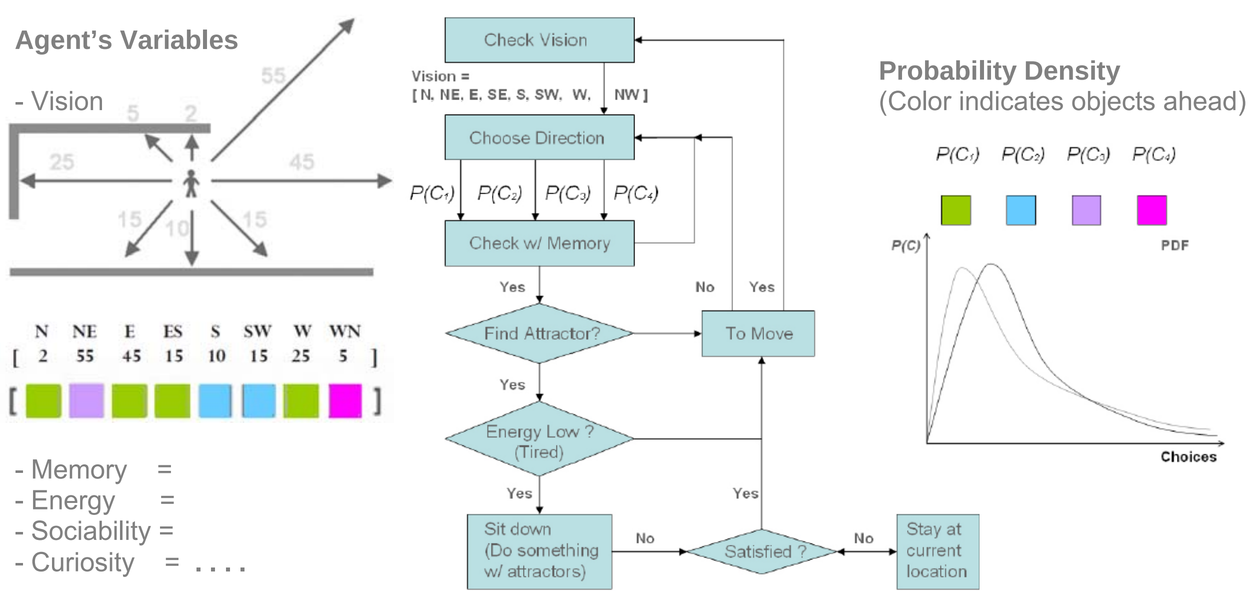


FIGURE 4. Agents' Vision (left), Example of Stochastic Decision Process (middle and right)

variations in interactions among the visitors. Some like to have more interactions with others and some do not. When two sociable people meet within a certain level of proximity on the street, conversation will start, and this begins to add a sense of place to the geometry.

Every trial of the tool produces different possible events, based on the stochastic decision making process which is implemented in figures. However, by seeing series of possible scenarios, users can identify behavioral tendencies and actions inherent in the space which they are designing. A series of behavioral choices are chained and branched out from the current states of the figure, and they will choose their next states based on situations from their surrounding environments, their internal variables, and the different probabilities with which possible next states are weighted. Likelihoods of one behavior over the other can be expressed by directing randomness to a certain manner. Their chances of occurring are not uniformly distributed. For the sake of simulation, these differing probabilities are interpreted and applied to the figure's tendencies to select one state over the others. The interpretation of the events' likelihoods should ideally reflect patterns of human behavior in real-life from available knowledge-based data. The aim of these exercises is to render the potential events and scenes of the building ideas, which had been left simply as states of geometry.

Another implementation for the figures' behaviors is called Privacy mapping. Privacy mapping is a concept for gauging the level of privacy inside the space. This method is particularly interesting for analyzing residential spaces. I based the concept on two proposed criteria: level of exposure to the exterior environment (public) and the numbers of others who can see within the house. The first criterion can be simply calculated by projecting invisible rays 360 degrees around every point within spaces. Areas of visual exposure to the exterior in elevations are calculated using trigonometry, and it is simply the sum of these areas from the entire surroundings that indicates the privacy level of this coordinate point. This measure is based on agents' locations and is highly influenced by the materiality (transparency) of the walls and partitions inside spaces. Any change of the materials and their locations may affect this measure which provides spatial hierarchy based on the privacy.

One other criterion that I proposed is to measure privacy based on how many others are seen. When the density of the space is higher, people are likely to be seen by others, and this value will indicate the capacity of spaces that can maintain a level of privacy. The relationship between the distance around oneself and one's perception of others has been introduced the

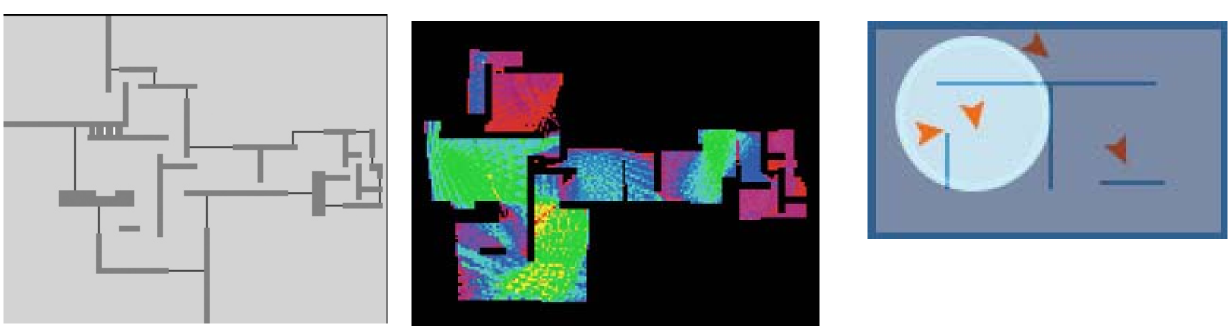


FIGURE 5. Plan of Country House by Mies, 1932 (left), Privacy level, from Red being high to Purple, being low (middle), Personal distance among Agents (right).

studies such as *Proxemics* by Edward T. Hall. In "*The Hidden Dimension,*" Hall suggests that four feet is considered to be "personal distance" which maintains a small hypothetical protective sphere between oneself and others. This dimension also varies, depending on one's cultural background, and is hardly ever objectified. It is an interesting idea to test the spaces with the different cultural dimensions possessed by various social groups. As a starting point of this study, any numbers of others within the dimension from Hall's studies, plus-or-minus four feet in radius as a threshold value, are counted as people who can greatly influence one's sense of privacy. Any others in visible areas outside of this distance are considered less influential; hence, those values decrease in inverse proportion to the distances from oneself.

CONCLUSION

The animated result from the tool brings a sense of place to the geometry by displaying the figures in motion. By observing the scales and numbers of footsteps that it takes to move figures from one location to the other, viewers can obtain a sense of speed and physical three-dimensionality. The tool's results offer a starting point for initiating users' conceptions of a space and help them to immediately study a spatial design as inhabited. Through considering the results that I have obtained from the series of experiments using the Space Re-Actor, I began to realize that the importance of the tool lies in the effort to integrate "human involvement" into a spatial design. A space seems to exhibit different characteristics according to the numbers of occupants, their objectives inside the space, and the proportions of groups with different degrees

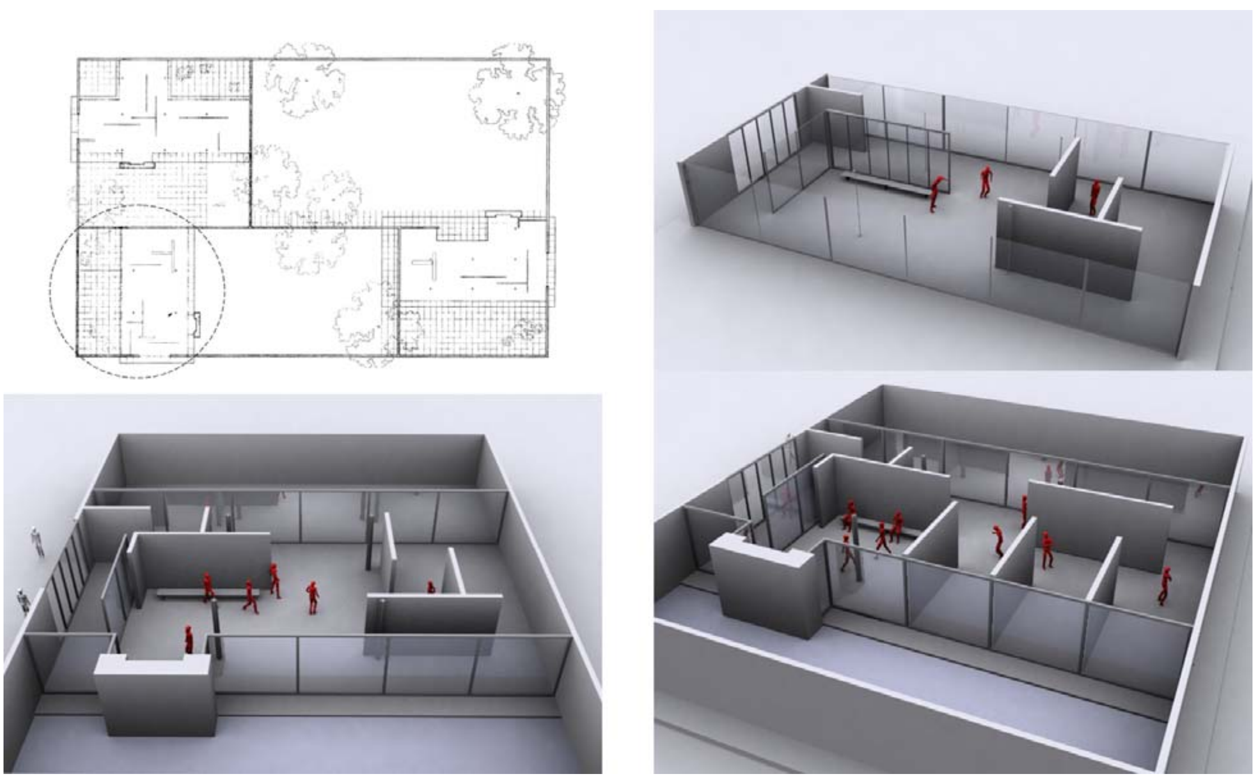


FIGURE 6. The group of court house by Mies (1931) and various Behavioral Responses from plan schemes with different degrees of Privacy using the Space Re-Actor

of sociability. Behavioral aspects of the spatial design have yet to be addressed well in any existing design tools. One may be able to obtain discrete numerical data about people's comfort levels as they depend on the density of a space through analytical means, but comfort and other characteristics in particular spatial layouts and conditions can be more fully grasped through the

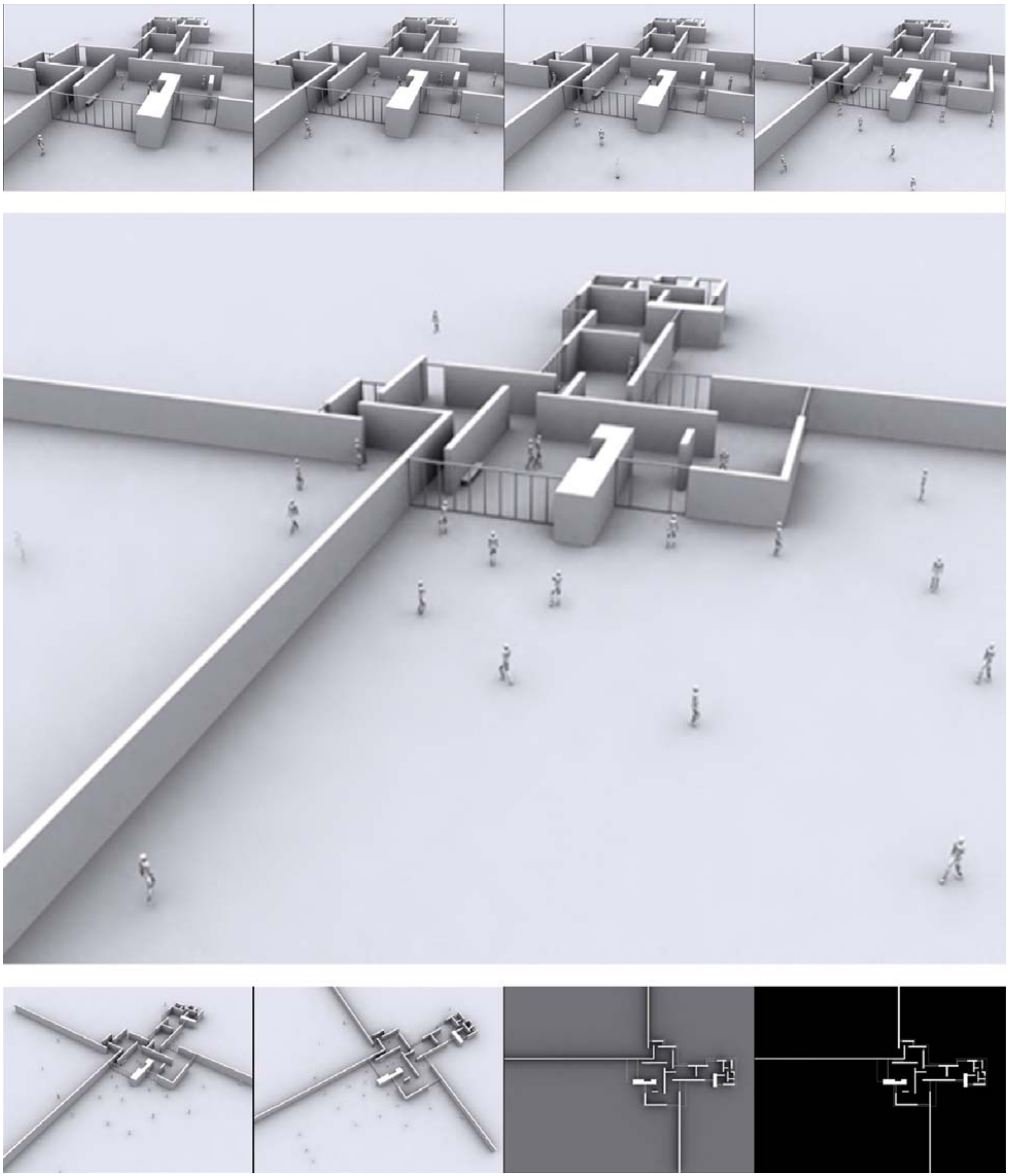


Figure 5. Brick Country House by Mies (1923) and Animated results from the Space Re-Actor

use of simulations. Aggregation of all the architectural components, such as doors, partitions, windows, staircases, and furniture in specific layouts can be understood through the use of the Space Re-Actor.

Describing the people's behaviors computationally is a controversial issue. That "an Agent-based model can never perfectly duplicate human social interaction" is a perennial critique and a genuine problem for the computer and social scientists involved in behavioral simulations. Simulations are always based on premise that human beings will behave in certain ways under specific conditions. Even though possible behaviors are incorporated from actual events and scenarios, there is no proof that the behaviors include all possible occurrences. The scope of reliability of results obtained through behavioral simulation is a critical bone of contention for researchers, not only in computer science, but also in operations research, social science, behavioral economics, and so on.

The fact that many scientists involved with neural-coding are actively using probabilistic approaches such as Bayesian analysis suggests that any study involving our "self" requires some degree of understanding of indeterminacy. Unlike discrete and determinant geometrical construction seen in architectural design, behavioral implementation can not be fully described without applying a notion of "indeterminacy." Describing a transition from one state to the other or one formal solution to the other by the decision of a human being may require a probabilistic treatment as is often true in nature.

"Is this really Simulation?" is an interesting question. The Space Re-Actor provides animated possible scenarios based on the qualities inherent in architectural spaces. The plausibility or persuasiveness of these results probably depends also on viewers' own attitudes and perceptions toward the "reality" around themselves. But the tool provides multiple interpretations of the spaces based on the stochastic decision process implemented in the figures, which is meant to reflect the indeterminacy of human behavior. "Reality," I believe does possesses indeterminacy. To determine types of people visiting the Mies's country house at particular times of the day is not realistic. Every trial of the tool produces different possible events, and the original intention was to identify tendencies in behaviors and actions by seeing series of possible scenarios. What we have witnessed in the several visualization results of the Space Re-Actor may not be certain to occur in reality. However, neither can one completely deny the potential occurrence of such events. It is, so to speak, dramatization tool for spatial designers, and I argue that the Space Re-Actor may more properly be considered as the first instance of a new category I am calling "Enactment software." One's attitude toward "reality" will surely influence how we regard this new tool, the Space Re-Actor.

Spatial qualities in architectural design cannot be fully evaluated solely by observing geometrical constructs without reference to inhabitants placed inside. The consequent emergent behaviors of people induced by characteristics of spaces may be impossible to predict, and indicate another layer of spatial qualities beyond the visible, formal, and aesthetic. A method for informing designers about the potential interactions between human behaviors and the spaces they are designing will constitute a valuable tool. As a future exploration, these figures' profiles or behavioral tendencies can be controlled by the users of the tool based on the conditions and the building types which they are considering.

ACKNOWLEDGEMENTS

I would like to sincerely thank, my academic adviser at the Massachusetts Institute of Technology, Professor Takehiko Nagakura for his insightful guidance and constant support during the development of this project. I would also like to thank my current academic adviser, Professor Kostas Terzidis at the Harvard University, Graduate School of Design for his valuable comments and feedback. This project is based on my thesis work, and it would not have been possible without their help.

REFERENCES

Benenson, I. and Torrens, P., 2004, Geosimulation: Automata-based modeling of urban phenomena. :Wiley.

Franpton, K., 1992, Modern Architecture: A Critical History. New York, NY.: Thames and Hudson.

Gilbert, N. and Troitzsch, K. G., 2005, Simulation for the Social Scientist. :Open University Press.

Hall, E. T., 1969, The Hidden Dimension, Garden City, New York: Doubleday.

Nagakura T. and Chatzitsakyris P., 2006, “A Synthetic Moviemaker for Spatial Representation.” Proceedings of ACM SIGGRAPH, Sketches, 2006, Boston.

Narahara T. and Terzidis K., 2006, “Multiple-constraint Genetic Algorithm in Housing Design,” [ACADIA International Conference, Synthetic Landscapes | Digital Exchange] Louisville USA. 12-15 October.

Werner, B. 1997, Mies van der Rohe. Berlin, Germany: Birkhauser Verlag.

Wilensky, U. 1999, “NetLogo,” Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL; available at http://ccl.northwestern.edu/netlogo/.

SHULGI: A GEOSPATIAL TOOL FOR MODELING HUMAN MOVEMENT AND INTERACTION

S. BRANTING, University of Chicago, Chicago, IL
Y. WU, Argonne National Laboratory
and Illinois Institute of Technology, Chicago, IL
R. SRIKRISHNAN, Argonne National Laboratory
and Illinois Institute of Technology, Chicago, IL
M. R. ALTAWEEL,* Argonne National Laboratory
and University of Chicago, Chicago, IL

ABSTRACT

Theories and models of movement are central to understanding a wide range of social phenomena that crosscut numerous disciplines. Despite its importance, walking is one form of movement for which models are badly underdeveloped within tools such as transportation geographic information systems (GIS-T) that modern planners and researchers use. Analysts and researchers are in need of tools that can integrate the complexities of nonmechanized modes of transportation and are also capable of easily producing appropriate models that can be studied in the service of making critical decisions and obtaining useful insights within different social-theoretical perspectives. This paper introduces the developing agent-based modeling tool called SHULGI. This tool enables multiple sociological and socio-physical processes to be incorporated in order to assess different factors that can impact transportation decision making and events. Particular attention is given to SHULGI's modeling capabilities for addressing pedestrian movements within an urban context. Models discussed in this paper include metabolic and human decision models that will enable different types of agents to develop satisficing or optimizing route selections. These selections can impact overall traffic and movement within the transportation networks addressed. Simulation functionality and outputs provide a preliminary demonstration of SHULGI's capabilities.

Keywords: Pedestrian Movement, Agent-Based, GIS-T, Repast Simphony, Archaeology

INTRODUCTION

Theories and models of movement are central to understanding a wide range of phenomena that crosscut numerous disciplines. Models of how people move and interact are complicated by the fact that people are thinking agents. We perceive the world and the actions of people around us and chose to alter our own movements and activities accordingly. This context is experienced by nearly everyone, every day, and it is something that scholars working within the broad interdisciplinary boundaries of location theory have strived to understand for centuries (von Thünen 1826; Weber 1909; Christaller 1933). A critical question in location theory is how does transportation and movement through space affect activities and the use of space at

* *Corresponding author address*: Mark R. Altaweel, Argonne National Laboratory, 9700 S. Cass Avenue Argonne, IL 60439; phone: 630-252-1379; e-mail: maltaweel@anl.gov, mraltawel@uchicago.edu.

particular places? This seemingly simple question belies a complex dialectic between transportation and land use that plays an important role in defining human-human and human-environment interactions.

This paper introduces an object-based geosimulation environment for modeling transportation, called SHULGI, which seeks to address this critical question in location theory. SHULGI uses the newly released Repast Simphony (Repast S) modeling toolkit as its foundational technology (North et al. 2005). With a modular object-based framework, SHULGI is designed as an easy to use, flexible simulation tool that can model multiple modes of transportation at multiple temporal and spatial scales. This paper will begin to demonstrate SHULGI's capabilities in modeling pedestrian traffic within an urban context. Despite its importance, models for addressing walking in existing spatial modeling tools, such as transportation geographic information systems (GIS-T), are largely underdeveloped. SHULGI incorporates a methodology for modeling pedestrian movement as developed by Branting (2004), which has demonstrable applicability in both the present and the past.

SOCIAL THEORY AND MOVEMENT

While movement creates opportunities for nearly boundless sets of choices and actions, it is at the same time self-limiting. Choices individuals make concerning what to do at a given point in time, along with the routes chosen to get to the required location to engage in that activity, will variously constrain what individuals can do in wider or shorter spans around the time of that activity. The realization of the importance of choices and movement within the finite framework of time has been used for decades to powerful effect within time-geography as a way to parse social and individual actions and interactions (Hägerstrand 1970; Pred 1986; Kwan 1999). The concept of the path taken by the individual through time and space is central to time-geography conceptualizations, accurately reflecting the importance of linear movement within this equation. Traffic patterns, as the aggregates of individuals' movement, can be seen through this lens of time-geography as constraining and, at the same time, as very revealing of the choices people make, including the activities they engage in individually and corporately.

In a complementary theoretical vein, location theory has long stressed the importance of movement and transportation in parsing choice, particularly in terms of land-use planning. Land-use and transportation form a symbiotic relationship in this regard, a point stressed by Blunden (1971). By the time of von Thünen around 1800, transportation was seen as a key factor in optimizing land-use allocation (von Thünen 1826). This was subsequently carried over into explicitly urban situations through work undertaken by Alonso (1964). Transportation and land-use played a key role in the calculus of industrial production and consumption put forth by theorists such as Weber (1909) and Isard (1956) and research addressing the transport of goods and services (Christaller 1933; Lösch 1954). The importance of transportation and movement seen in each branch of these basic schools of location theory continues to this day, with transportation and land-use playing key roles in human-to-human and human-to-environmental interactions (Rapoport 1977; Rapoport 1990; Johnston and de la Barra 2000; Stough 2004).

MODELING MOVEMENT

GIS-T Approach to Transportation and Pedestrian Traffic

A methodology addressing both the paths and patterns of traffic along with their symbiotic relationship to land-use offers enormous possibilities for exploring not only the form of the built environment, but also the social and individual thoughts and actions involved in its creation and alterations. GIS-T provides particularly good tools and methods to address both of these areas and indeed is being used by city and regional planners around the world. GIS-T is a domain within GIScience that only emerged during the 1990s, linking together established transportation models and methods with the display and organizational abilities of GIS (Thill 2000; Goodchild 2000). It provides network-based modeling and analysis tools that are particularly effective in forecasting and exploring traffic patterns both at various levels of aggregation as well as individual movement.

For situations where traffic is constrained, either through the framework of the built environment or through repeated practice (Helbing et al. 1997), GIS-T proves particularly effective. This applicability includes the case of pedestrian traffic, which might be considered much more variable than that of mechanized modes of transport (e.g., cars or trains) that require significant infrastructural investments (Branting 2004). There are a number of advantages to modeling pedestrian transportation using network-based approaches like those found in GIS-T (Branting 2004). Yet, surprisingly, this functionality is an almost unexplored area within the domain, with most GIS-T work focused specifically on modern mechanized transport. One of the key reasons why this is surprising is because walking is the most common mode of movement and transportation in the present as well as the past (Amato 2004; Solnit 2000; Branting 2004). For preindustrial cities, such as the ancient city at Kerkenes Dağ that is the focus of the case study below, modeling traditional modes of transportation like walking is crucial for understanding transportation. But even in present-day case studies, pedestrian transportation is a critical part of any transportation system and one that is growing in importance in light of issues with congestion and energy policies.

While the pedestrian GIS-T methodology developed by Branting (2004) has performed remarkably well, there is room for improvement. Three key difficulties with the methodology in its current implementation were identified. First, there are limitations in the flexibility of the method. Dynamic activity in time and space has always presented problems for GIS (Peuquet 2001). This was compounded by difficulties in implementing more complex scenarios, particularly those involving interactions between people in time and space. Second, the methodology became cumbersome as the range of simultaneous factors applied to a given scenario was expanded. This difficulty was manifested less in terms of computational processing speed but had more to do with the specification of the combinatory interactions between the factors. Third, the pedestrian GIS-T methodology was difficult and expensive to utilize. Several commercially available software packages, including ESRI's ArcGIS and Caliper's Transcad, were used to power both the modeling and the enormous amount of data formatting and preparatory work necessary between the various stages. Significant investments were needed in terms of both time and money to apply the methodology.

Agent-Based Modeling and Transportation Modeling

Agent-based modeling (ABM) provides a way beyond these difficulties with the pedestrian GIS-T methodology. Numerous cases have applied ABMs to transportation issues (Rickert and Nagel 2001; Batty 2001; Lake 2001), particularly because ABMs provide clear benefits to understanding human choice and decision making in movements within grids or networks. SHULGI has been designed to take full advantage of this flexibility and to allow for future extensions within a modular framework. In addition, SHULGI has been designed to make the modeling of complex transportation easier than in the GIS-T methodology by facilitating data integration, analysis, and model creation and use within this framework.

Among the benefits of SHULGI, the system allows users to load different forms of spatial and nonspatial data, either through a user's graphical user interface (GUI) or eXtensible markup language (XML) file reference, to instantiating simulation agents or other objects. Users can select from among the built–in least cost pathway algorithms, which currently include the A* and Dijkstra algorithms (Hart et al. 1968; Dijkstra 1959), or load those created by the user through XML references. In SHULGI, the A* and Dijkstra algorithms have been slightly modified so that they can factor multiple edge weights. Specific models that can be chosen in SHULGI include those that address pedestrian movements. These models address metabolic expenditures, velocities, and route decision making. The metabolism models that have been integrated include the Pandolf and McDonald models (Pandolf et al. 1977; McDonald 1961). These models take into account energy requirements over a given terrain. The velocities for these models are derived from studies of human walking (Sun et al. 1996; Kawamura et al. 1991). Time costs for the activity can also be determined from separate models such as Imhof (1968). Route decision models, which enable agents to decide where to go at a given time step, will be discussed shortly. In addition to using currently integrated models, users can create their own models, and then reference these within an XML structure to add the models within SHULGI's model library that provides the full array of model choices to analysts.

In SHULGI, models are not integrated directly with agent or entity objects; this aspect of its design is beneficial in that it allows various theoretical perspectives to be integrated using the same agent or entity[1] types. In summary, users can select a range of models addressing a specific behavior, with processes relevant for applied theoretical perspectives strictly implemented within models. Users can use SHULGI and Repast S tools that allow the construction of additional models and objects (North et al. 2005; Parker et al. 2006). In addition, SHULGI can be run within a computer cluster; the current implementation uses Terracotta software (Terracotta 2007). Current simulation models can be executed by using process-oriented discrete events, enabling contextual data to be passed to objects applying processes at relevant simulation time scales.

Object types that can be instantiated with spatial (e.g., shapefile) and nonspatial (e.g., CSV file) data include those listed in Table 1. After the appropriate simulation data have been loaded, users can map the data to an object type (e.g., TransportAgent) applied in the simulation. Data loaded and mapped include velocity values or default initial locations. Mapping the specific object parameters and data is accomplished through XML references or a GUI that

---

[1] Entity objects are software objects that have behaviors or processes associated with them. Entity objects can be agents; however, entities can also be noncognitive physical or environmental objects such as trees or river systems.

applies users' input. Regardless of the technique, users reference the locations (e.g., column titles, file locations) of the data values and the object variables that link to the data. Once the data mapping is complete, the simulation can be executed. Alternatively, simulation scenarios can be executed via batch runs, which can link to multiple computer nodes or a single computer. Batch mode is used for both verification and validation (V&V) as well as for producing a large number of results. Input data for batch simulations are all loaded via XML files using custom and Repast S batch mode capabilities. In batch mode, all mapping of object parameters with input data is performed automatically using XML files after the mode is launched. Figure 1 shows SHULGI's current interface used for data, model, and analytical tool integration as well as data mapping to simulation objects.

Table 1 Object types integrated in SHULGI

Object	Description
TransportAgent	Any type of moving agent
ShulgiEdge	Modified network edge
Structure	Compounds or fixed structures
Network	A Repast S network object
DestinationNode	A node a TransportAgent desires to reach
RoadNetworkNode	A node a TransportAgent can traverse

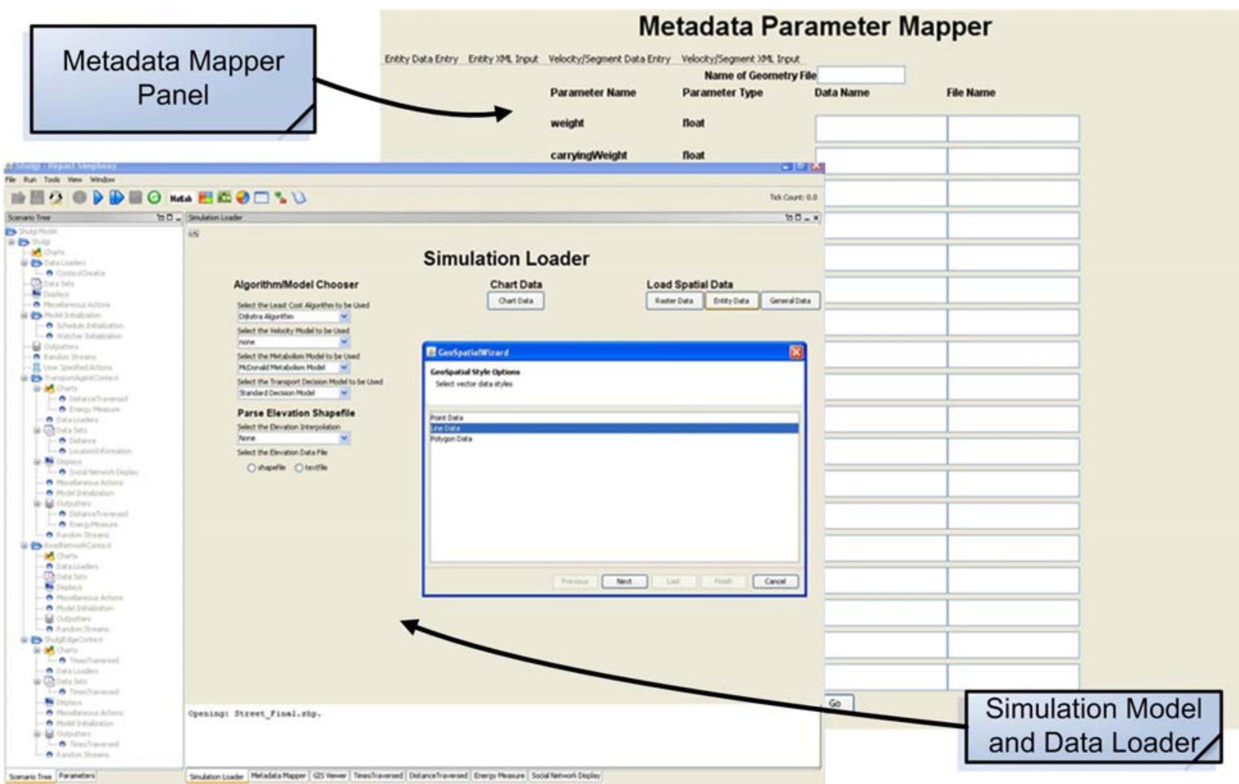


FIGURE 1 SHULGI's interface that loads models and data as well as allows mapping of data to objects

Kerkenes Dağ Case Study

The primary area to which SHULGI has been applied to date is that of pedestrian movements and route decision making. Current example scenarios derive from data obtained from the archaeological site of Kerkenes Dağ, an ancient city in Turkey dating to the 6th century B.C. (Summers et al. 2005; Summers et al. 2007). This example is one of the same datasets used in the development of the GIS-T methodology, allowing a direct comparison between the two approaches. As with the GIS-T methods, the type of agent is disaggregated from a generic human being into different genders and age groups based on the differing walking characteristics of each (Branting 2004). Six different types of agents have been used so far: young men (aged 10–34), young women (aged 10–34), middle-aged men (aged 35–55), middle-aged women (aged 35–55), older men (aged 56–75), and older women (aged 56–75). The movement space that agents operate within SHULGI can be either a grid or network. In the examples executed, SHULGI has been applied using road networks. Road network edges (i.e., ShulgiEdge object) can store multiple weight values or calculate a given weight as needed in runtime. The Standard Decision Model (SDM) is a SHULGI test model used by agents to decide where to move based on agent goals. Calculated metabolic expenditures using the Pandolf or McDonald models, with inputs for slope and speed based on localized internal and external factors, determines how costly an edge is for the given type of agent in the Kerkenes Dağ examples. In other words, the more energy required to cross a given route segment, the more costly that entire route becomes to an agent. In this first example, agents simply determine where to go by finding the nearest DestinationNode (DN) that was not previously visited by an agent. The agent then determines the shortest path route using a least cost algorithm. A DN represents an ultimate destination that agents want to reach and that are associated with the center of compounds or walled urban blocks. Agents, therefore, attempt to visit each DN only once or until the simulation ends. In contrast, a RoadNetworkNode (RNN) is a type of DN that forms the ShulgiEdge nodes, but an RNN can be traversed continuously in the SDM. The following Java code in Figure 2 illustrates the decision step used by agents to find the path to the nearest DN.

```
public void decideTargetLocation(){
       target = null;
       shortestPath = new ArrayList<ShulgiEdge>();
       RoadNetworkNode lastStep = null;

       //sort DestinationNodes, fined nearest node
       PriorityQueue<DestinationNode> destinationQueue =
               priorityQueue(ta.getX(), ta.getY(), ta.getZ(),
               destinationNodes.size());
       destinationQueue.addAll(destinationNodes);
       boolean got = false;
        while((!got || lastStep == null)&& !destinationQueue.isEmpty()){
              //potential node
               target = (RoadNetworkNode) destinationQueue.poll();
              if (!visited.contains(target))  //see if visited node before
                     got = true;
              else
                     continue;

              lastStep = getLastStep(); // edge prior to DestinationNode
        }
       removeEdges(); //removes other destination nodes
       if (lastStep != null)
              findShortestPath(source, lastStep); //gets shortest path
       addEdges(); //add all removed DestinationNodes
       if (shortestPath.size() != 0)  //connect to final edge
          shortestPath.add((ShulgiEdge) network.getEdge(lastStep, target));

 }
```

FIGURE 2 Decision step used by agents to find the path to the nearest DN

Currently, output produced by SHULGI can be viewed or analyzed in different formats and within different tools. This function includes a GIS viewer that used GeoTools 2.3 (GeoTools 2007) to output spatial changes and movements of agents. Chart tools and network layout displays can be used to visualize some of the results. Aggregate and agent outputs produced are also provided as comma delimited data, which can then be analyzed with Repast S or other tools.

Results

Results from SHULGI help demonstrate the modeling tools' capabilities discussed previously. Scenarios have focused on energy expenditure by agents as they traverse the street network of Kerkenes Dag. Figure 3 shows the street network layout of the city with compounds, urban blocks, city wall, streets, and young male agents. In this image, GeoTools is used to visualize agents as they move across the network based on the SDM. One of the key interests for the initial modeling trials is to determine which roads are the most primary used by agents in order to reach most destinations. Among other output results that can be displayed, the GeoTools viewer in SHULGI can show the number of instances that roads are used or the overall traffic volume. Other GIS output results can be selected by the user within a wizard editor coupled in SHULGI. Disaggregated and aggregated output, such as energy expended by agents, can be shown within graphs. Figure 4 indicates the mean energy expenditure of young men as they move along routes selected during simulation time steps. This graph is produced using Repast S plug-in tools for data output and display. In addition to mean energy expenditure, simulations record mean and maximum distance traveled as agents move along various routes (Figure 5); in this case, output is shown using Repast S chart tools. Figure 5's results provide an indication as to whether or not young men need to travel significantly further in a given time instance to reach more distant nodes from their original locations. Other visual displays include network layouts; however, this display has not been used extensively in the current SHULGI scenarios. At this stage of SHULGI's development, our efforts are focused on conducting V+V on the current models that have been integrated as well as on performing further tool enhancements to enable more detailed analyses of modeling results.

These results are preliminary for the agent categories to which the scenarios have been applied, but outputs produced do show the benefits of applying an approach that can couple metabolic and other relevant cost and decision models within an overall ABM approach. By using models for social and physical processes affecting route costs, results can be used to assess optimal or satisfactory route selections for agents as influenced by multiple factors.

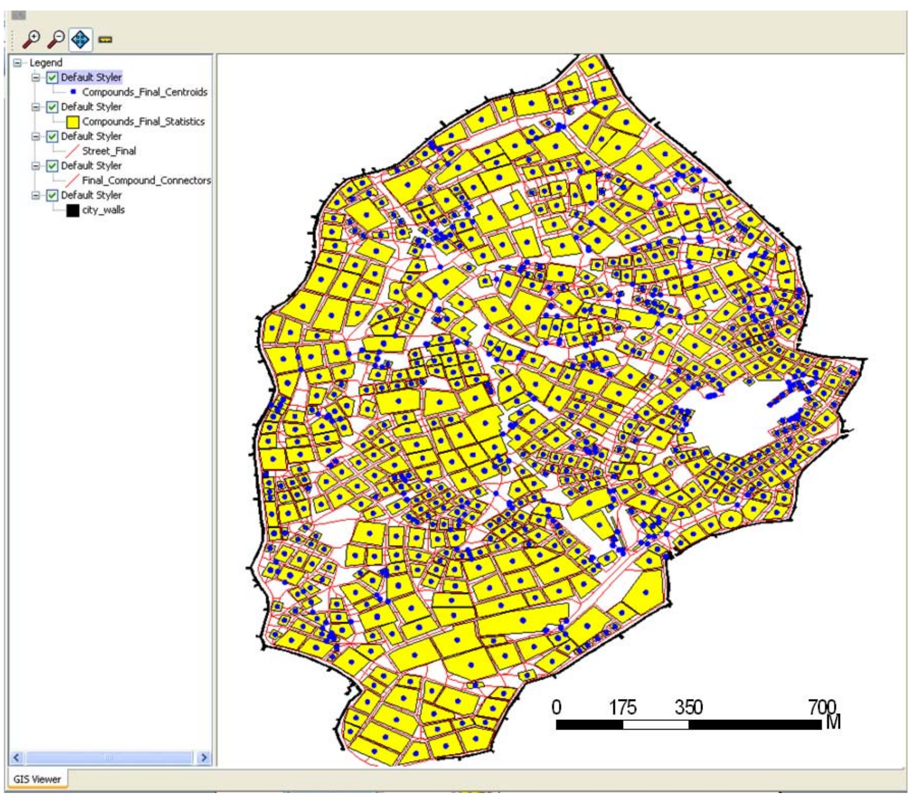


FIGURE 3 The street network (in red), compounds (in yellow), city wall, and young male agents (blue dots) during the simulation as they move and reach compounds

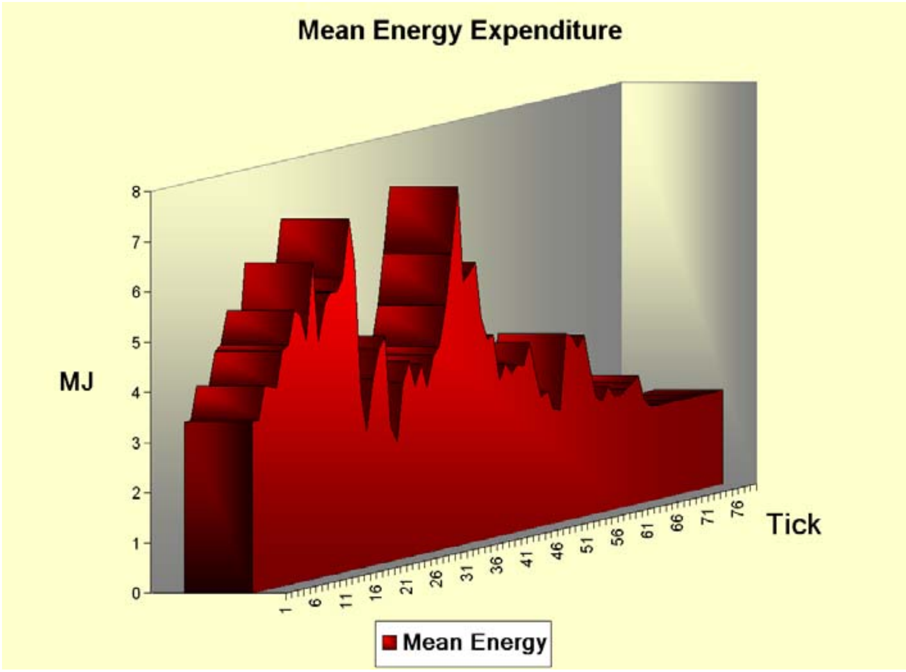


FIGURE 4 Mean energy expenditure (MJ/time step) for all young male agents

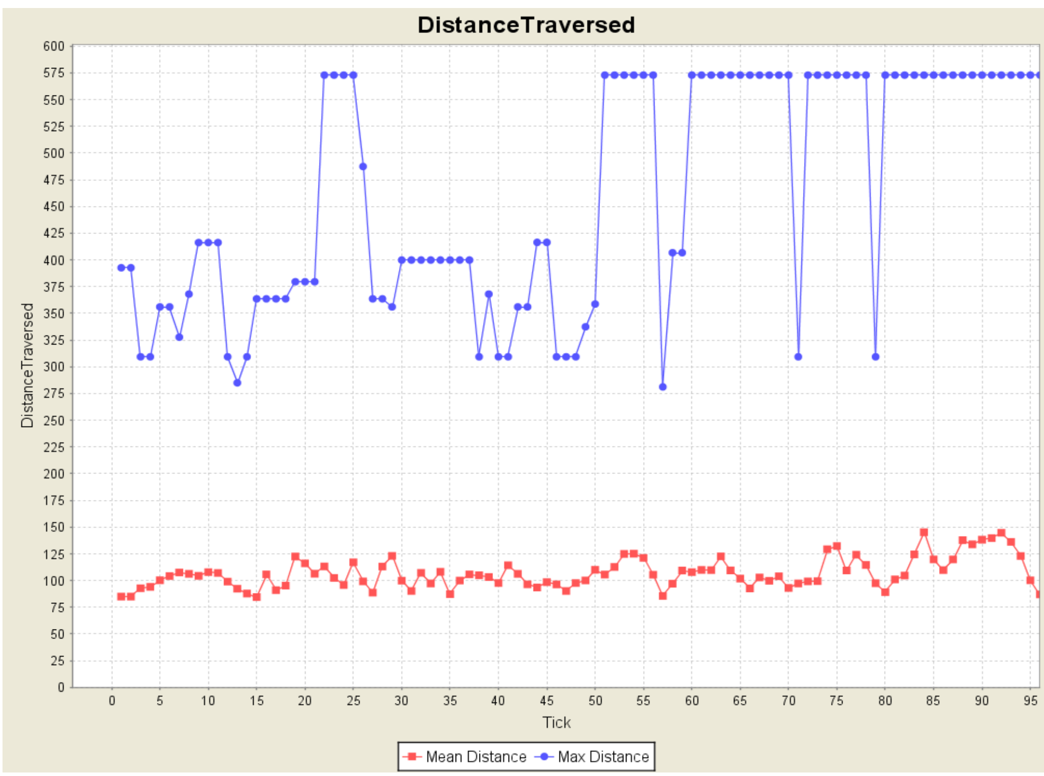


FIGURE 5 Mean (red) and maximum distance traveled (blue) during simulation time steps; distance measurements are in meters

DISCUSSION

Although SHULGI's full range of planned modeling capabilities have not been implemented, its tools have proven able to reproduce results developed within GIS-T applications and provide significant enhancements. By using an ABM approach, the modeling of pedestrian or other forms of movement can be far more flexible to theoretical applications and integration of alternative models and behaviors as compared to other geospatial approaches. Proper ABM design enables alternative models at varied temporal scales to be applied to different scenarios, allowing researchers to assess how individuals in a given transport structure may change or adapt their movements. The ability of SHULGI to simplify the integration of data and models for pedestrian movements addresses deficiencies in tools used by urban planners and geographers.

We anticipate that as SHULGI develops further, enhanced capabilities will be enabled. For instance, although SHULGI has been executed on multiprocessor clusters, our effort seeks to make the inclusion of multiple nodes a relatively simple process for the user by using appropriate GUI-based tools within SHULGI's main interface. We intend to apply and enhance SHULGI's capabilities to address other domains, including the modeling of pedestrian traffic in modern urban centers, as the tool further develops. The development of SHULGI as a freely available tool will insure its affordability for scholars and researchers interested in applying this ABM technology to their own transportation and land-use research.

ACKNOWLEDGMENTS

This work is supported by the U.S. Department of Energy, Office of Science, under contract number DE-AC02-06CH11357. This work was also supported by the Joint Theory Institute funded together by Argonne National Laboratory and the University of Chicago. Data for the modeling demonstrations were provided by the Kerkenes Dağ Project (http://www.kerkenes.metu.edu.tr).

REFERENCES

Alonso, W., 1964, *Location and Land Use: Toward a General Theory of Land Rent*, Cambridge: Harvard University Press.

Amato, J.A., 2004, *On foot: a history of walking,* New York: New York University Press.

Batty, M., 2001, "Agent-based Pedestrian Modeling," *Environment and Planning B: Planning and Design B* **28**:321–326.

Blunden, W.R., 1971, *The Land-Use/Transport System: Analysis and Synthesis*, Oxford: Pergamon Press.

Branting, S., 2004, *Iron Age Pedestrians at Kerkenes Dağ: An Archaeological GIS-T Approach to Movement and Transportation*, Ph.D. Thesis, State University of New York at Buffalo.

Christaller, W., 1933, *Die zentralen Orte in Süddeutschland*, Jena, Germany: Gustave Fischer.

Dijkstra, E.W., 1959, "A Note on Two Problems in Connexion with Graphs," *Nmerische Mathematik* 1:269–271.

GeoTools, 2007; available at http://geotools.codehaus.org.

Goodchild, M.F., 2000, "GIS and Transportation: Status and Challenges," *Geoinformatica* **4**(2):127–139.

Hägerstrand, T., 1970, "What about People in Regional Science?," *Papers of the Regional Science Association* **24**:7–21.

Hart, P., N.J. Nilsson, and R. Bertram, 1968, "A Formal Basis for the Heuristic Determination of the Minimum Cost Paths," *IEEE Transaction on Systems Science and Cybernetics SSC* **4**(2):100–107.

Helbing, D., J. Keltsch, and P. Molnár, 1997, "Modelling the Evolution of Human Trail Systems," *Nature* **388**(6637):47–50.

Imhof, E., 1968, *Gelande und Karte*, Zürich, Switzerland: Eugen Rentsch-Verlag.

Isard, W., 1956, *Location and Space-Economy: A General Theory Relating to Industrial Location, Market Areas, Land Use, Trade, and Urban Structure*, Cambridge, MA: M.I.T. Press.

Johnston, R.A., and T. de la Barra, 2000, "Comprehensive Regional Modeling for Long-Range Planning: Linking Integrated Urban Models and Geographic Information Systems," *Transportation Research Part A* **34**:125–136.

Kawamura, K., A. Tokuhiro, and H. Tahechi, 1991, "Gait Analysis of Slope Walking: A Study on Step Length, Stride Width, Time Factors and Deviation in the Centre of Pressure," *Acta Medica Okayama* **45**:179–184.

Kwan, M-P, 1999, "Gender and Individual Access to Urban Opportunities: A Study using Space-Time Measures," *The Professional Geographer* **51**:210–227.

Lake, M., 2001, "The Use of Pedestrian Modeling in Archaeology, with an Example from the Study of Cultural Learning," *Environment and Planning B: Planning and Design B* **28**:385–403.

Lösch, A., 1954, *The economics of location*, New Haven: Yale University Press.

McDonald, I., 1961, "Statistical Studies of Recorded Energy Expenditure of Man: Expenditures on Walking Related to Weight, Sex, Age, Height, Speed, and Gradient," *Nutrition Abstracts and Reviews* **31**(3):739–762.

North, M.J., T.R. Howe, N.T. Collier, and J.R. Vos, 2005, "The Repast Simphony Development Environment," in *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects*, pp. 69–79, Chicago, IL, October 13–15.

Pandolf, K.B., B. Givoni, and R.F. Goldman, 1977, "Predicting Energy Expenditure with Loads While Standing or Walking Very Slowly," *Journal of Applied Physiology* **43**:577–581.

Parker, M.T., M.J. North, N.T. Collier, T.R. Howe, and J.R. Vos, 2006, "Agent-Based Meta-Models," in *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms*, pp. 159–166, Chicago, IL, September 21–23.

Peuquet, D.J., 2001, "Making Space for Time: Issues in Space-Time Data Representation," *GeoInformatica* **5**(1):11–32.

Pred, A., 1986, *Place, Practice and Structure: Social and Spatial Transformation of Southern Sweden,* Cambridge: Polity Press.

Rapoport, A., 1977, *Human Aspects of Urban Form: Towards a Man-Environment Approach to Urban Form and Design,* Oxford: Pergamon Press.

Rapoport, A., 1990, "Systems of Activities and Systems of Settings," in *Domestic architecture and the use of space: An interdisciplinary cross-cultural study*, S. Kent (ed.), pp. 9–20, Cambridge: Cambridge University Press.

Rickert, M., and K. Nagel, 2001, "Dynamic Traffic Assignment on Parallel Computers in TRANSIMS," *Future Generation Computer Systems* **17**(5):637–648.

Solnit, R., 2000, *Wanderlust: A History of Walking,* New York: Penguin.

Stough, R.R., 2004, "Institutions, Land Use and Transportation," in *Handbook of Transport Geography and Spatial Systems,* D.A. Hensher, K.J. Button, K.E. Haynes, and P.R. Stopher (eds.), pp. 27–42. Amsterdam: Elsevier.

Summers, G., F. Summers, and S. Branting, 2005, "Kerkenes News 7, 2004"*, Kerkenes Haberler 7*, Ankara, Turkey: METU Press; available at http://www.kerkenes.metu.edu.tr/kerk2.

Summers, G., F. Summers, and S. Branting, 2007, "Kerkenes News 9, 2006," *Kerkenes Haberler 9*, Ankara, Turkey: METU Press; available at http://www.kerkenes.metu.edu.tr/kerk2.

Sun, J., M. Walters, N. Svensson, and D. Lloyd, 1996, "The Influence of Surface Slope on Human Gait Characteristics: A Study of Urban Pedestrians Walking on an Inclined Surface," *Ergonomics* **39**(4):677–692.

Terracotta, 2007, available at http://www.terracotta.org.

Thill, J-C, 2000, "Geographic Information Systems for Transportation in Perspective," *Transportation Research Part C* **8**:3–12.

von Thünen, J.H, 1826, *Der Isolierte Staat in Beziehung auf Landwirtschaft und Nationalökonomie*, Jena, Germany: Fischer Verlag.

Weber, A, 1909, *Über den Standort der Industrie: Reine Theorie des Standorts*, Tübingen, Germany: Mohr.

Decision and Information Sciences Division
Argonne National Laboratory
9700 South Cass Avenue, Bldg. 900
Argonne, IL 60439-4832

www.anl.gov