ANL/DIS-06-7

Argonne NATIONAL LABORATORY

THE UNIVERSITY OF CHICAGO

# Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects

*Co-hosted by*

**Argonne National Laboratory**
**The University of Chicago**

*In association with*
North American Association for Computational Social and Organizational Science

## Gleacher Center

450 North Cityfront Plaza Drive
Chicago, Illinois

September 21-23, 2006

ISBN 0-9679168-7-9

Agent 2006

**Suggested Citations:**

***Proceedings:*** Sallach, D.L., C.M. Macal, and M.J. North (editors), 2006, *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects,* ANL/DIS-06-7, ISBN 0-9679168-7-9, co-sponsored by Argonne National Laboratory and The University of Chicago, September 21–23.

***Paper:*** Jones, A., and B.C. Smith, 2006, "Title of Paper," in *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects,* ANL/DIS-06-7, ISBN 0-9679168-7-9, Sallach, D.L., C.M. Macal, and M.J. North (editors), co-sponsored by Argonne National Laboratory and The University of Chicago, September 21–23.

**About Argonne National Laboratory**

Argonne is a U.S. Department of Energy laboratory managed by The University of Chicago under contract W-31-109-Eng-38. 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.agent2006.anl.gov

Reproduced from best available originals.

ANL/DIS-06-7

Argonne NATIONAL LABORATORY

THE UNIVERSITY OF CHICAGO

# Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects

*Co-hosted by*

**Argonne National Laboratory**
**The University of Chicago**

*In association with*
North American Association for Computational Social and Organizational Science

**Gleacher Center**

450 North Cityfront Plaza Drive
Chicago, Illinois

September 21-23, 2006

Agent 2006

**A U.S. Department of Energy laboratory managed by The University of Chicago**

## CONTENTS

## SATURDAY, SEPTEMBER 23, 2006

## Social Simulation Applications

## FOREWORD

Welcome to Agent 2006, co-hosted by Argonne National Laboratory and The University of Chicago, in association with the North American Association for Computational Social and Organizational Science (NAACSOS). This is the seventh 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. These priorities have helped us attract quality papers and thus, keep the conference fresh and stimulating.

The broader theme of the 2006 conference is *Results and Prospects*. It has been a decade since Josh Epstein and Rob Axtell published their Sugarscape model, and it rapidly became an exemplar for the use of agent modeling to explore social simulation dynamics. It thus seems like an appropriate time to assess the achievements of this approach. What progress has been achieved? What near-term goals can be achieved within present assumptions? What should be considered the research horizons of the social agent simulation paradigm?

Our invited speakers — Uri Wilensky, Scott Page, and Noshir Contractor — are making creative contributions within, and to, social agent modeling. Their contributions have helped to shape areas that are diverse and significant, such as user-friendly development environments, dynamic models of culture, and network-based applications. Such themes provide a rich context for the array of papers that precede and follow the invited talks.

The overall combination of conference presentations will allow us to explore the present results and future prospects of this productive modeling paradigm. We hope that you will find the conference to be both educational and stimulating. We appreciate your participation and look forward to future contributions as well.

Finally, we hope you enjoy Agent 2006 and become increasingly committed to the kinds of social science progress that computational modeling makes possible. Once again, welcome.

The Center for Complex Adaptive Agent Systems Simulation
Argonne National Laboratory and The University of Chicago

David L. Sallach
Charles M. Macal
Michael J. North
Thomas D. Wolsko

## ACKNOWLEDGMENTS

We acknowledge the support of many people in organizing Agent 2006. Kathy Ruffatto handled registration, administration, and logistics, along with assistance from Dee Albarado. Argonne's Technical Services Division prepared the program book and the proceedings for publication. Margaret Clemmons served as project manager, with support from Michele Nelson in graphic design, Andrea Manning, and Argonne's Document Processing Center.

## ORGANIZING COMMITTEE

Charles Macal, Argonne National Laboratory and The University of Chicago
Michael North, Argonne National Laboratory and The University of Chicago
David Sallach, Argonne National Laboratory and The University of Chicago
Thomas Wolsko, Argonne National Laboratory

Thursday, September 21, 2006

# Methods, Toolkits, and Techniques

# Model Performance Optimization Techniques

# BREEDING FASTER TURTLES: PROGRESS TOWARDS A NETLOGO COMPILER

F. SONDAHL,* Northwestern University, Evanston, IL
S. TISUE, Northwestern University, Evanston, IL
U. WILENSKY, Northwestern University, Evanston, IL

## ABSTRACT

Despite the stereotype concerning their biological counterparts, NetLogo's turtles are fast. NetLogo (Wilensky 1999a) contains a sophisticated interpreter that has been highly optimized. Nevertheless, NetLogo turtles aren't as fast as they could be. Interpretation necessarily incurs a performance penalty. Thus, we are in the process of replacing NetLogo's interpreter with a compiler. This transition is happening in phases. In this paper, we discuss the architecture of NetLogo's interpreter and explain the first phase of the transition to compilation, which uses inlining to generate efficient bytecode from abstract syntax trees. This technique measurably reduces the interpreter overhead, while permitting a gradual transition to a compiled architecture. We approach the task of compiler design from the perspective of a powerful agent based modeling language with "low threshold" design goals. Preliminary benchmark results are presented, in addition to a forecast of further steps towards a full NetLogo compiler.

**Keywords:** NetLogo, compilers, performance, agent based modeling,
Java Virtual Machine

## INTRODUCTION

Despite the stereotype concerning their biological counterparts, NetLogo's turtles are fast. NetLogo (Wilensky 1999a) contains a sophisticated interpreter that has been highly optimized. Nevertheless, NetLogo turtles aren't as fast as they could be. We are working to remedy this. Because the use of even a sophisticated interpreter incurs a necessary performance penalty, we are in the process of replacing NetLogo's interpreter with a compiler. To better understand the current development focus, it is helpful to discuss the historical background and philosophical motivation of NetLogo.

The design of the original Logo language was guided by the slogan "low threshold, high ceiling" (Papert 1980). NetLogo upholds this tradition (Tisue & Wilensky 2004). It should be easy for new users to learn NetLogo and build models, but it should also be possible for advanced modelers to build "research-grade" models. There are inevitable trade-offs between these two design goals. NetLogo's adoption by thousands of modelers, from rank novices to veteran hackers, suggests that a healthy balance between these goals is being achieved.

Largely for reasons of "low threshold", NetLogo was originally implemented as an interpreted language. Even though a compiler would make models run faster, building a compiler is time-consuming and would not help lower NetLogo's threshold. Early development effort

* *Corresponding author address:* Forrest Sondahl, EECS Tech Institute C359, 2145 Sheridan Rd, Evanston IL 60208-0834; e-mail: forrest@northwestern.edu

was better spent in other directions (e.g., building an integrated development environment and adding features). As NetLogo matured and was more widely adopted by the research community, speed became a ceiling issue for advanced users. In response, the interpreter was substantially restructured and tuned for performance. This resulted in dramatic speed improvements, but eventually we felt that further significant improvements could only be achieved through compilation.

We should note that compilation and interpretation are not mutually exclusive approaches. The Java language is a prime example (Gosling et al. 1996). Java source code is compiled to an intermediate form (bytecode), which is interpreted by the Java Virtual Machine (JVM). Similarly, NetLogo source code is first transformed from text to an intermediate form (arrays of abstract syntax trees), which are then interpreted by the NetLogo interpreter (Tisue & Wilensky 2004). Although this system is quite fast when compared to naive interpreter implementations, it still results in measurable overhead costs when compared to models written in pure Java. To move beyond this performance barrier, we decided to compile the NetLogo language directly into JVM bytecode.

Building a new compiler from the ground up was problematic for several reasons. First, NetLogo’s code base is now large – there are over 300 built-in language "primitives" in NetLogo. Each primitive is implemented as a Java class. This large body of Java code represents a substantial investment of development time, which we wanted to leverage for use by the compiler. Second, there are features of the NetLogo language that can be smoothly handled by an interpreter, but would frustrate the implementation of a traditional compiler – for example, the frequent context switching as various agents execute their code, to simulate concurrent activity. We are not suggesting that traditional compiler-writing methods are inapplicable to the NetLogo language – in fact, we will be employing them later (see “Future Work” below). However, for the first phase of development we chose an alternative method which achieves significant performance gains, while integrating seamlessly with NetLogo's existing interpreter, and maintaining most of the flexibility of language development that the interpreted system provided. This integration is a strong first step in NetLogo's transition towards a complete compiler system targeting the Java Virtual Machine platform.

## IMPLEMENTATION

### Bytecode Inlining Overview

Our hybrid solution involves combining the existing interpreter with partial compilation. One important aspect of the new compiler is a technique we call “JVM bytecode inlining”. The NetLogo interpreter itself is running on the JVM platform, which means that each of the primitives accepted by the NetLogo interpreter maps to some sequence of JVM bytecode that gets executed. Our bytecode inliner extracts this sequence and inserts it into the compiled code. Inlining avoids the overhead of calling the sequence as a separate method, which is what the NetLogo interpreter had to do. The combined sequences of bytecode are then dynamically loaded as a single new Java method. The end result is similar to the output that would be given by a traditional compiler. However, we avoid some of the complexity of a full-blown compiler, because we are able to "steal" the bytecode that was pre-compiled by a traditional Java compiler (e.g., Sun's *javac* ). Another simplification is that at present we are only compiling individual NetLogo commands (and their arguments), not yet sequences of commands; the interpreter still moves from command to command and handles procedure calls. For the task of bytecode extraction and generation we use ASM, which is a small and fast Java bytecode manipulation framework (Bruneton et al. 2002).

## Bytecode Inlining Example

Conceptually, we can think of the bytecode inlining process as NetLogo dynamically extending the pool of primitives in the interpreter's repertoire, by replacing a command's entire abstract syntax tree with a single combined primitive that we synthesize to do the task more efficiently.

The text of a NetLogo program is first lexically parsed and tokenized. An array of abstract syntax trees is created, variable references are resolved, nested command blocks are linearized, etc. Eventually, the output is a NetLogo `Procedure` object, which consists of an array of `Command` objects, each of which is the root of a tree containing `Reporter` objects. All NetLogo primitives fall into these two categories, reporters (e.g., `+`, `sin`, `patch-ahead`) and commands (e.g., `rt`, `fd`, `print`). Reporters return ("report", in our terminology) values; commands do not – they simply "perform" some action.

For clarification of the bytecode inlining process, we will present a step-by-step example for a simple code fragment: "`rt (a + 5)`". This NetLogo code causes a turtle (agent) to turn (change its heading) `(a + 5)` degrees to the right, as shown in the right-hand side of Figure 1. On the left-hand side of Figure 1 there is a graphical depiction of the abstract syntax tree created by NetLogo's parser. As mentioned above, each node of the tree is a Java object. For instance, "`rt`" maps to an instance of class "`_right`", a subclass of `Command`. Similarly, "`+`" maps to the class "`_plus`", "`a`" to the class "`_turtlevariable`", and "`5`" to the class "`_constdouble`", subclasses of `Reporter`. These classes each define an execution method. Command classes define a "`perform()`" method (with a `void` return type), and reporter classes define a "`report()`" method (with an `Object` return type).

For this example, we will denote the instances of classes `_right`, `_plus`, `_turtlevariable`, and `_constdouble`, as R, P, T, and C respectively (see Listing 1). The NetLogo interpreter would evaluate our example tree by

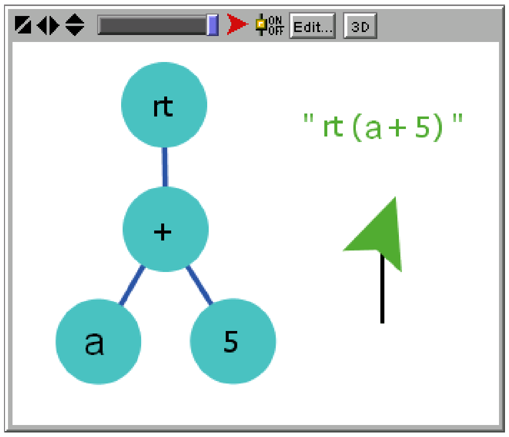


**FIGURE 1:** Abstract syntax tree

calling the `R.perform()`. This method in turn would call `P.report()`, which would call the `report()` methods for each of P's children nodes (T and C), add the results together, and return the result to R's `perform()` method, which would then change the executing agent's heading by the appropriate number of degrees. Pseudo-code is shown in Listing 2.

**LISTING 1:** Textual representation of the tree

```
_right (object R):   "rt"
    _plus (object P):   "+"
        _turtlevariable (object T):   "a"
        _constdouble (object C):   "5.0"
```

*Underscored words correspond to Java classes representing NetLogo language primitives.*

Instead of stopping with the abstract syntax tree, the new NetLogo compiler processes the tree to create JVM bytecode. First it performs a post-order traversal of the tree (e.g. order: `a 5 + rt`). For each node it visits, we use ASM's ClassReader to extract the bytecode from the `perform()` or `report()` method that would have been called by the interpreter (e.g. `_constdouble.report()`, `_turtlevariable.report()`, etc). We perform some minor transformations on the extracted bytecode before passing it to ASM's `ClassWriter`, to generate the `perform()` method of our new class. Instead of transferring "`return`" statements from the extracted method to the generated method, it leaves the result that would have been returned on the JVM operand stack. When the traversal is finished, the resulting class is written to a byte array, and dynamically loaded into the JVM using a custom `ClassLoader`. We create a new object G from the newly loaded class.

The pseudo-code that is representative of the transformation is shown in Listing 3. A textual representation of the JVM bytecode that is produced by the new bytecode compiler is shown in Listing 4. Note that the tree which originally consisted of four objects (R, P, T, and C) was replaced by a flattened version with just one object (G).

**LISTING 2**: Pseudo-code for interpreted system

```
R.perform():
  context.agent.turnRight( P.report() ) ;

P.report():
  return C.report() + T.report() ;

C.report():
  // C has a member field that holds the constant value, 5
  return C.storedValue ;

T.report():
  // The symbol "a" corresponds to an index into a variable array
  // For this example, assume the index, T.variable_number, is 7.
  return context.agent.getTurtleVariable( T.variable_number ) ;
```

**LISTING 3**: Pseudo-code for the bytecode

```
G.perform():

  context.agent.turnRight( 5 + context.agent.getTurtleVariable( 7 ) ) ;
```

**LISTING 4**: Simplified* JVM bytecode that results from compiling "rt (a + 5)"

```
G.perform():
ALOAD 1                                  // push the "context" onto stack
GETFIELD Context.agent : Lagent;         // get the current context's agent
BIPUSH 7                                 // push 7 onto stack
INVOKEVIRTUAL Agent.getTurtleVariable    // get Object stored in var 7
CHECKCAST Double                         // check that var 7 held a number
INVOKEVIRTUAL Double.doubleValue ()D     // convert Double Object -> double
LDC 5.0                                  // push 5 onto stack
DADD                                     // now "a + 5" is on stack
DSTORE 2                                 // store "a + 5" in JVM local #2
ALOAD 1                                  // push "context" onto stack
GETFIELD Context.agent : Lagent;         // get the current context's agent
CHECKCAST Turtle                         // make sure agent is a Turtle
DLOAD 2                                  // load "a + 5" back onto stack
INVOKEVIRTUAL Turtle.turnRight (D)V      // cause turtle to "rt (a + 5)"
```

_*Package names have been omitted for brevity. Actual compiler output contains additional bytecode for runtime type-checking and error handling, which has been omitted for clarity._

## Bytecode Inlining Advantages

Several aspects of this process increase performance:

1. Constant values. In the old system, constant values that are known at compile time -- such as 5 and 7 (the turtle-variable index) -- were stored in member fields. In the new system, they are hard coded as more efficient PUSH or LDC bytecode instructions.
2. Casting. The old interpreter's `report()` methods only return generic Objects, and the calling method must check the return type and cast it to the appropriate type. The new compiler is often able to perform this type checking at compile-time, and generate the appropriate bytecode, omitting the unnecessary casting.
3. Primitive type checking. Similarly, the new compiler is able to deal more efficiently with Java's primitive types – e.g. `booleans` and `doubles` – avoiding many cases where the interpreter was forced to "box" the results as `Boolean` or `Double` objects.
4. Method invocations. The old system required four perform/report method invocations, whereas the new system only requires one.

We conducted several tests to approximately measure the comparative influence of these aspects on increasing performance. Optimization of constant values (#1) accounted for around 4% of the performance improvement, whereas the type conversion aspects (#2 and #3) accounted for roughly 50%. Decreasing the number of method invocations (#4) is credited with the remaining 46% of the speedup.

Another aspect that could be contributing to the performance increase is synergy with JIT (just in time) compilers. Inlining method bytecode creates larger contiguous sections of bytecode in a single method, which can improve opportunities for standard intraprocedural compiler optimizations, particularly when the inlined method bodies are simple (e.g., Scott 2000; Bellotti et al. 2004 ). It is our hope that JIT compilers can better optimize our generated bytecode. As of yet, we have not measured the influence of bytecode generation on JIT compilers. Since Sun Microsystems' HotSpot compiler performs its own form of "class-hierarchy aware" method inlining (Paleezny et al., 2001), it is unclear whether synergistic interaction is occurring. Further benchmarking is required to examine this issue.

## PERFORMANCE RESULTS

### Performance Benchmarks

In our graphs, "Cur" denotes the current development build of NetLogo as of July 21, 2006, with compilation disabled, and "Cur+" denotes the same build with compilation enabled. All benchmarking was done on a 3.2 Ghz Pentium 4 with 2 GB of RAM, running Windows XP Professional. The results shown in Figure 2 used Sun's Java 2 Runtime Environment version 1.5.0_06, with the HotSpot(TM) Client VM. The results shown in Figures 3 and 4 used Sun's Java 2 Runtime Environment version 1.4.2_10, with the HotSpot(TM) Server VM. All benchmarks were run with the graphical display disabled, to better measure engine speed.

Figure 2 presents a view of the history of performance in NetLogo on one particular benchmark, the so-called "GasLab" benchmark. Our benchmarks are not synthetic micro-benchmarks; they are real models from NetLogo's models library. The GasLab benchmark is based on a model called "GasLab Gas in a Box" that demonstrates the Maxwell-Boltzmann distribution in an ideal gas (Wilensky 1997, 1999b). Figure 2 shows that performance improvements came quickly in NetLogo's early days. Between versions 1.1 and 1.2, the interpreter was restructured from a stack-based to tree-based (Tisue & Wilensky, 2004). Since that time NetLogo hit a

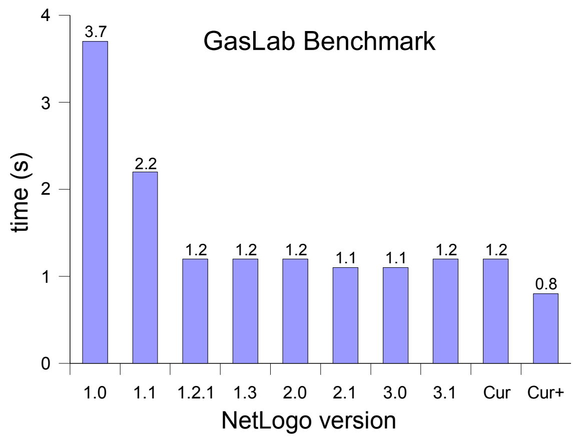


**FIGURE 2**: NetLogo performance history

performance barrier that persisted up until the creation of the new compiler. This breakthrough cut execution time on the GasLab benchmark to 66% of what it had previously been.

Figures 3 and 4 present a broader perspective on the performance gains attributable to the compiler. Figure 3 shows the results for each of the 13 benchmarks in NetLogo's benchmark suite. Note that the performance increase varies considerably between models. For instance, the bytecode compiler only shaved 5% off of the execution time of the Flocking benchmark (#7), while the time for the 1-D Cellular Automata benchmark (#13) was nearly cut in half. Figure 4 shows the performance gain across the board; on average, execution time was cut by 23%.

## Comments on Performance

For the purposes of this paper, we have limited ourselves to comparing NetLogo against its past performance. Although it would be interesting to do so, we have not compared NetLogo's performance against that of other popular agent based modeling platforms, or against models written in "raw" Java code without the aid of a specialized toolkit. It is often difficult to make such comparisons fairly, since various modeling platforms suggest different natural implementations of a given model, as well as different techniques for tuning and optimization. For further discussion on this topic, and a general review of several popular agent based modeling toolkits, see Railsback et al. (2006).

The new NetLogo compiler is still very much a work in progress. Not all of NetLogo's language primitives are yet taking advantage of the new system, and we expect continued performance increases. Some preliminary tests give us hope that the continuation of this project, in addition to further forays into bytecode generation (see "Future Work" below), may eventually lead to as much as 3x speed improvement over NetLogo 3.1 (i.e. a reduction of execution time to 33% of its previous value).

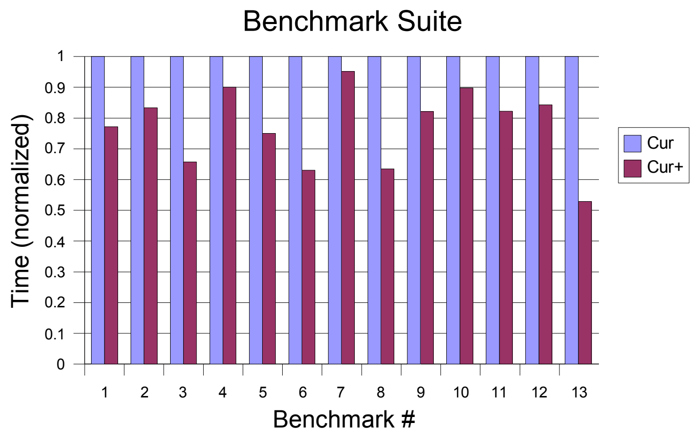


**FIGURE 3**: Bytecode inlining improvement

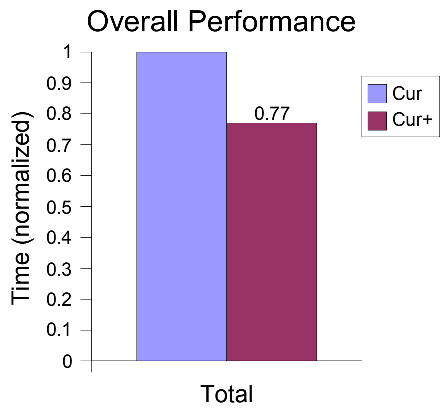


**FIGURE 4**: Benchmark average

Will NetLogo performance ever match/surpass the performance of raw Java code? If both Java and NetLogo are being compiled to bytecode, can't NetLogo language be just as fast? We expect not, because of differences in the source languages. For example, Java is statically typed (you must declare variable types – "`int n`", "`double d`"), whereas NetLogo is dynamically typed (any variable can hold any data type). The conciseness and flexibility of dynamic typing contributes to NetLogo's low threshold. Static typing allows more type-checking to be done at compile-time, and thus more efficient bytecode can be produced. Dynamic typing is one example of a trade-off between "low threshold" and "high performance."

Note that "low threshold" here isn't only relevant to novice programmers. Expert users, too, would pay a cost of slower authoring if type declarations were required. One reason NetLogo is popular among researchers and other "high-end" users is their ability to rapidly develop prototype models in NetLogo. In the end, the question is not which language is the fastest; nobody wants to write agent-based models in assembly language. The pertinent question is whether the language you want to model in is high level enough to ease development and maintenance, yet fast enough for your needs.

## FUTURE WORK

### Towards a NetLogo Compiler

As mentioned earlier, bytecode inlining is just the first phase in implementing a complete NetLogo compiler. The details we have discussed only involve creating the bytecode to deal with a single NetLogo command. Compiling the abstract syntax tree for each single command is effective at boosting performance if expressions are long (e.g., see Listing 5). However, many NetLogo models have a low command-to-expression-length ratio (e.g., see Listing 6), and in

**LISTING 5**: A procedure from the NetLogo "CA 1D Elementary" model (Wilensky, 1998a)

```
to do-rule  ;; patch procedure
  let left-on? on?-of patch-at -1 0
  let right-on? on?-of patch-at 1 0

  ;; each of these lines checks the local area and (possibly)
  ;; sets the lower cell according to the corresponding switch
  let on?-of patch-at 0 -1
    (iii and left-on?       and on?       and right-on?)        or
    (iio and left-on?       and on?       and (not right-on?))  or
    (ioi and left-on?       and (not on?) and right-on?)        or
    (ioo and left-on?       and (not on?) and (not right-on?))  or
    (oii and (not left-on?) and on?       and right-on?)        or
    (oio and (not left-on?) and on?       and (not right-on?))  or
    (ooi and (not left-on?) and (not on?) and right-on?)        or
    (ooo and (not left-on?) and (not on?) and (not right-on?))
end
```

**LISTING 6**: A procedure from the NetLogo "Flocking" model (Wilensky, 1998b)

```
to turn-at-most [turn max-turn]  ;; turtle procedure
  ifelse abs turn > max-turn
    [ ifelse turn > 0
        [ rt max-turn ]
        [ lt max-turn ] ]
    [ rt turn ]
end
```

such cases this technique is not as effective. These models should see a greater performance increase as we extend the compiler to generate bytecode for more than a single command at a time, which will be the next the phase of compiler work.

The first step in this direction will be to compile basic (that is, non-branching) blocks of adjacent commands. In the next step we will extend the compiler to process control structures — branches, loops, and procedure calls. Finally, whole procedures and entire NetLogo models will be compiled.

## Additional Optimizations

In addition to the core compiler plan outlined above, there are several other promising areas of optimization work:

- NetLogo language procedures could be inlined. This is a separate issue from the Java method inlining discussed in this paper, which occurs at the JVM level. The motivation, however, is much the same. When modelers write short NetLogo procedures that are called frequently, speed could be increased by inlining that procedure into the calling NetLogo procedure.
- A type inferencing system could be designed for local variables. Even though NetLogo is dynamically typed, there are situations where we could detect the type of a variable at compile time and optimize the code accordingly.
- We have already designed a peephole bytecode optimizer, which removes some inefficient code that is created during the bytecode generation process. More peephole optimizations could be introduced.
- Higher-level optimizations. NetLogo currently has a variety of sophisticated optimizations in place. For example, the code snippet "`turtles with [color = red]`" reports an agentset of all the red turtles in the world. The primitive "`any?`" tests whether or not an agentset is empty. A naive interpreter running the code "`if any? turtles with [color = red]`" would first find all the red turtles, and then see if that set is empty. NetLogo internally rewrites this code to stop looking for red turtles as soon as it has found one. There are a fair number of such optimizations already in place, but more could be designed.

### Caveats

So far, we have only discussed the benefits of inlining. There is also a drawback associated with inlining that becomes more salient as the amount of bytecode generation increases – namely "code bloat." As reported by Bellotti et al (2004), excessive method inlining in Java can result in decreased performance. Because our bytecode generation technique does not require the use of any extra JVM local variables, we have reason to hope that we avoid this negative effect of inlining. But performance issues aside, the JVM imposes a limit of 64 kilobytes for the bytecode of a method body, and so completely inlining the contents of a long NetLogo procedure into a single method will not be possible. We will need to find a balance between inlining and method invocation.

A second issue that arises is not particular to inlining, but is a consequence of generating bytecode. NetLogo allows models to be saved as Java applets, which can then be run in a web browser. Currently, the applet embeds our interpreter. With the compiler, the model would need to be compiled before it could be run; however, for security reasons unsigned applets may not load dynamically generated bytecode. We will resolve this issue by generating a custom JAR file for the applet, which will contain the compiled bytecode for the given model.

## CONCLUSION

Bytecode inlining provides greater flexibility than a more traditional compilation process. New primitives can still be added to the NetLogo language with the same ease as before – bytecode inlining extracts the compiled bytecode behind the scenes. This is particularly useful for NetLogo, which remains a rapidly evolving language. Our hybrid approach also allows some code to remain interpreted while other code is compiled. This intermingling of interpreted and compiled code provides the foundations for a gradual transition towards a full NetLogo compiler. Using the techniques described in this paper, we have already experienced a significant performance increase, and we expect future work on bytecode generation to result in further speedups. NetLogo's turtles are faster now than ever before, and they are still picking up speed.

## REFERENCES

Bellotti, F., Berta, R., & De Gloria, A., 2004, "Evaluation and optimization of method calls in Java," in *Software: Practice and Experience*, Vol. 34, No. 4, pp. 395-431.

Bruneton, E., Lenglet, R., & Coupaye, T., 2002, "ASM: A code manipulation tool to implement adaptable systems," in *Adaptable and extensible component systems,* Grenoble, France; available at http://asm.objectweb.org/current/asm-eng.pdf.

Gosling, J., Joy, B., & Steele, G., 1996, *The Java Language Specification,* Reading, MA: Addison-Wesley.

Paleezny, M., Viek, C. & Click, C., 2001, "The Java HotSpot Server Compiler," in *Proceedings of the Java Virtual Machine Research and Technology Symposium (JVM '01)*, pp. 1-12.

Papert, S., 1980, *Mindstorms: Children, Computers, and Powerful Ideas,* New York: Basic Books.

Railsback, S., Lytinen, S., & Jackson, S., 2006, "Agent-based Simulation Platforms: Review and Development Recommendations," *Simulation,* (manuscript in review).

Scott, M., 2000, *Programming Language Pragmatics*, San Francisco, CA: Morgan Kaufmann Publishers Inc.

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., 1997, *NetLogo GasLab Gas in a Box model*, Evanston, IL: Center for Connected Learning and Computer-Based Modeling, Northwestern University, available at http://ccl.northwestern.edu/netlogo/models/GasLabGasinaBox.

Wilensky, U., 1998a, *NetLogo CA 1D Elementary model*, Evanston, IL: Center for Connected Learning and Computer-Based Modeling, Northwestern University, available at http://ccl.northwestern.edu/netlogo/models/CA1DElementary.

Wilensky, U., 1998b, *NetLogo Flocking model*, Evanston, IL: Center for Connected Learning and Computer-Based Modeling, Northwestern University, available at http://ccl.northwestern.edu/netlogo/models/Flocking.

Wilensky, U., 1999a (updated 2006), NetLogo [Computer software] (Version 3.1), Evanston, IL: Center for Connected Learning and Computer-Based Modeling; available at http://ccl.northwestern.edu/netlogo.

Wilensky, U., 1999b, “GasLab: An extensible modeling toolkit for exploring micro-and-macro-views of gases,” in N. Roberts, W. Feurzeig & B. Hunter (Eds.), Computer Modeling and Simulation in Science Education, pp. 151-178, Berlin: Springer Verlag.

# DYNAMIC AGENT COMPRESSION

S.A. WENDEL,[*] University of Maryland, College Park, MD
C. DIBBLE, University of Maryland, College Park, MD

## ABSTRACT

We introduce a new method for processing agents in agent-based models, which improves the efficiency of certain models and facilitates the creation of hybrid agent-based / systems-dynamics models. Dynamic Agent Compression allows agents to shift in and out of a compressed state based on their changing levels of heterogeneity. Sets of homogeneous agents are stored in compact bins, making the model more efficient in its use of memory and computational cycles. Modelers can use this increased efficiency to speed up the execution times, to conserve memory, or to scale up the complexity or number of agents in their models.

The advantages outweigh the overhead of Dynamic Agent Compression in models where agents are unevenly heterogeneous: where a set of highly heterogeneous agents are intermixed with numerous other agents that are either frequently inactive or that fall into broad internally homogeneous categories. Sample applications include modeling the life-cycle of extremely large populations (fish-larvae, bacteria), or modeling the diffusion of ideas or diseases through a population. Dynamic Agent Compression is not appropriate in models with few, exclusively complex, agents.

We describe in detail an implementation of lossless Dynamic Agent Compression, where no model detail is discarded during the compression process. Lossless compression also allows modelers to readily implement hybrid models where systems-dynamics components co-exist with traditional agent-based methods. We contrast lossless compression to lossy compression, which promises greater efficiency gains yet may introduce artifacts in model behavior.

**Keywords:** Agent Based Modeling, Scaling, Homogeneity, Compression

[*] *Corresponding author address:* Stephen Wendel, 2181 LeFrak Hall, Department of Geography, University of Maryland, College Park MD 20742. Email: swendel@umd.edu.

## INTRODUCTION

This paper introduces a new method for processing agents in agent-based models, which improves the computational and memory efficiency of certain models and facilitates the creation of hybrid agent-based / systems-dynamics models. Dynamic Agent Compression is born out of the observation that in many models, agents are unevenly heterogeneous. A significant portion of the agents may be inactive at a given time tick, or be active but fall into a few broad categories of internally homogeneous agents. These inactive or homogeneous agents are intermixed with highly heterogeneous agents, and both groups may change their level of heterogeneity over time.

Dynamic Agent Compression allows individual agents to move in and out of a compressed state based on their changing levels of heterogeneity. Sets of homogeneous agents are stored in compact bins, leading to more efficient use of computational resources. Modelers can use this increased efficiency to reduce execution times, to conserve memory, or to scale up the complexity or number of agents in their models.

Dynamic Agent Compression is an extension of Stage's (Stage *et al.* 1993) work on static agent compression, which he used to tackle the problem of large-scale models with prohibitive resource requirements. His algorithm works as follows: Consider each agent as a point in a multi-dimensional attribute space, where dimensions include location, age, health status, etc. Find clusters of agents with similar attributes and replace them with aggregated agents. Give the aggregated agents attributes that represent the core of the clusters and an "expansion factor" to represent how many agents are within the cluster. Use these aggregated agents instead of the individual agents to save computation and memory resources, either throughout the entire model or within expensive sub-models.

In addition to Stage's (1993) COMPRESS algorithm, a number of related methods have been used to address the computational resource requirements of extremely large numbers of agents. As with Stage (1993), many of the early efforts occurred in the ecology field as researchers modeled massive numbers of trees, fish, or bacteria. The most popular method of handling this problem is the "Super Individual" method (Scheffer *et al.* 1995), where a single agent in the model represents multiple entities in the real world (such as bacteria). For example Rose (Rose *et al*. 1993) employed a sampling and re-sampling algorithm to represent varying numbers of fish larvae, juveniles, and adults in his models. Hellweger (2006) recently expanded upon this literature with a location-specific method, which addresses distortions caused by the scarcity of super agents in localized pockets of the model. Research into multi-scale models has also examined the creation of aggregate agents (e.g. Servat 1998), but without the particular use of formal compression techniques.[1]

This paper proposes a number of innovations on Stage's method in order to make the procedure more flexible and efficient. Where Stage (1993) utilizes a once-off static compression of agents,[2] we propose a dynamic method that adapts to agents' changing heterogeneity during model execution. Moreover, we describe an extensible architecture where the individual

[1] As a side-note, the agent-compression library described here could readily be used to support emergent multi-scale modeling.
[2] Which can be repeated for each execution of expensive sub-models, but is nonetheless an inherently static method.

modeler needs only specify a limited set of parameters, and where multiple compression algorithms (including Stage's COMPRESS algorithm) can be made available. We then discuss in detail a "lossless" implementation of Dynamic Agent Compression, where no model detail is discarded during the compression process. Lossless compression allows modelers to readily implement hybrid systems-dynamics / agent-based models or limit resource requirements on existing large-scale models without biasing model behavior. We then discuss a more challenging but potentially rewarding "lossy" variant.

## METHODOLOGY

The key to Dynamic Agent Compression is that agents are compressed and decompressed as the simulation progresses, but that the model interacts with all agents as if they were in a traditional agent-based environment. At the start of the model, similar agents are grouped into compact bins, while agents with particularly unique attributes are left un-grouped. During the simulation, these bins ("Agent Containers") dynamically compress or de-compress one or more component agents, depending on the agents' attributes. If non-compressed agents join an existing group of agents in attribute space, they can also automatically join an existing Agent Container or form a new Agent Container. When a change in a compressed agent's attributes causes it to become heterogeneous, it decompresses into a separately instantiated agent or joins another Agent Container, as appropriate.

Dynamic Agent Compression can be implemented in a "lossless" or "lossy" manner. In a "lossless" compression, only agents that are strictly identical from the perspective of model behavior are combined into a single entity.[3] In a "lossy" compression, *similar* agents are combined based on the degree of compression desired.[4] Lossless compression provides an inexpensive way to make a model more efficient, but the efficiency gains are limited by the degree of homogeneity in the model. As will be discussed below, there is a class of models where this approach can be extremely valuable. Lossy compression discards limited information about the agents in pursuit of greater efficiency. It can be applied to any agent based model, but researchers must balance the efficiency gains against the potential for bias in model outcomes.[5]

Dynamic Agent Compression relies on the actions of a Compression Manager which manages the agent and Agent Containers. The Compression Manager filters calls from the model to create, modify, and query agents. When a set of homogeneous agents occurs, it creates an Agent Container to represent them more efficiently, and redirects calls from the model to the

[3] Agent "metadata" that is relevant to the modeler but which otherwise does not affect model behavior need not be homogeneous. For example, agents can have different names or unique identifiers that do not directly affect the model. As demonstrated in the implementation below, this agent-specific metadata can be stored and used as needed.

[4] For a general discussion of lossy versus lossless compression, see for example: http://en.wikipedia.org/wiki/Lossy_data_compression#Lossy_vs._Lossless_Compression. The method discussed here is conceptually a type of "transform codec," where existing agent information is processed into compressible pieces.

[5] Bias in model behavior would be considered a "compression artifact." As with any lossy compression technique, the goal is to provide modelers with the fewest artifacts for the desired level of compression. However, artifacts can never be completely avoided, and the modeler should perform a thorough sensitivity analysis to evaluate tradeoffs.

container. Each Agent Container holds a single agent with attributes that represent the set and a counter for the total number of agents in the set. It passes queries from the model to the sample agent, and monitors changes in the agent. If an agent differentiates itself from the set, the Compression Manager extracts it from the Agent Container and makes it a unique, individual agent. Thus, with the supervision of the Compression Manager, Agent Containers behave like their component agents; they accept time ticks, can be visualized, can be check-pointed, and can answer queries from data collection probes. In each circumstance, they respond like a set of individually instantiated objects.

In general, Dynamic Agent Compression is appropriate (i.e. the increased efficiency in agent storage outweighs the overhead of compression) in models where agents are unevenly heterogeneous: where a set of highly heterogeneous agents are intermixed with numerous other agents that are either frequently inactive or fall into broad internally homogeneous categories. Sample applications include modeling the life-cycle of extremely large populations (fish-larvae, bacteria), or modeling the spread of ideas or diseases through a population. By compressing the sets of otherwise homogeneous agents, and handling updates to the compressed agents appropriately, the model can save significant computational cycles and memory resources without compromising the model's behavior.

## SAMPLE JAVA IMPLEMENTATION

Here, we describe a lossless Dynamic Agent Compression implementation in which we applied a relatively straightforward change to an existing model to utilize our Agent Compression library. This implementation places the Agent Compression Manager directly between the main model class and the agents. The Agent Compression Manager is generic,[6] and knows nothing about the model or agents beyond their implementation of two simple class interfaces.[7] Our library has two Java classes, in addition to the two class interfaces:

- Core Agent Compression Library:
    - AgentCompressionManager Class. The AgentCompressionManager acts as the filter between the model and the agents, as described above. It receives calls from the model for creating, updating (stepping), and querying of agents, and then passes the call to the appropriate AgentContainer or Agent.
    - AgentContainer Class. The AgentContainer represents a set of CompressibleAgents in the model, and contains a sample agent from that set. On each time tick, it calls the sample agent, and removes it from the set if it diverges from the rest of the group.
- Interfaces required of the existing model:

[6] Dynamic Agent Compression can be implemented in a much more specific manner. In fact, we have two implementations of the AgentCompressionManager, one which is tailored with specific knowledge about the agents, and one which blindly creates Agent Containers from the supplied list of parameters. Here, we discuss the generic implementation.
[7] Note, most of the required functions were already present in our model, and are generally common among agent based models. Thus, it would not require much additional programming to adapt other models.

    - CompressibleAgent Interface. In order to use the library, a model's existing agent classes must implement the CompressibleAgent interface, which allows for simple communication between the agent and the Agent Compression Manager.
    - CompressibleModel Interface. In order to use the library, the modeler must implement the CompressibleModel interface, which allows for simple communication between the Agent Compression Manager and the model. The Interface has only one function, which allows the Manager to ask the model to instantiate an Agent (of its own desired subclass).

The creation of agents requires the most modification. In the model, calls to the Agents' constructors are replaced by calls to the AgentCompressionManager. AgentContainers are then generated dynamically as individual agents are added to the model. The agent creation proceeds as follows:

- The model tells the Agent Compression Manager that it would like to create Agents with a given set of initial parameters. The Compression Manager examines these parameters and either
    - Asks the model to instantiate the Agent as it normally would (using its pre-existing agent classes and initialization procedures). The model then gives the Agent to the AgentCompressionManager for its records; or
    - Chooses not to instantiate the Agent, since it either has a group of agents that is sufficiently similar to the newly desired Agent to warrant the creation of an AgentContainer or it already has a relevant AgentContainer. In the first case, the AgentCompressionManager creates an AgentContainer to wrap the group of individual Agents. In the second case, the AgentCompressionManager merely increments the relevant AgentContainer.

Once Agent initialization is complete, the model continues through the rest of its startup procedure. When the simulation begins, the flow of execution is as follows:

- At each time increment, the model calls its step() function. First, it updates its internal information and makes any other changes to the environment.
- The model then calls AgentCompressionManager.step(), which:
    - Calls step() on each unique Agent (i.e. each agent not within an AgentContainer).
    - Calls step() on each AgentContainer. The AgentContainer calls step() on its internal CompressibleAgent *for each "agent" contained in the container*. The first time it calls step(), it is a normal function call, unmodified from the Agent's normal procedure. The agent updates its internal information and interacts with the environment. The subsequent calls within the same time tick are "timeless steps", where the Agent is asked to interact with its environment but *not* update its internal information based on the passage of time.
- After each step() within AgentContainers, the AgentCompressionManager checks to see if their CompressibleAgents have changed attributes. If so, it extracts the Agents from the AgentContainers.

- It then checks the attributes of the newly-extracted Agents, to see if they can join another existing AgentContainer, trigger the formation of new AgentContainers, or remain as an individually instantiated Agents.

When the model needs to gather information about the agents (for data-logging or other purposes), it similarly queries the AgentCompressionManager instead of the individual agents. As with the step() function, the AgentCompressionManager queries the AgentContainers and the individual agents in order to gather the required information.

Figures 1 and 2 below illustrate the program execution flow of a typical model respectively before and after adaptation for Dynamic Agent Compression.

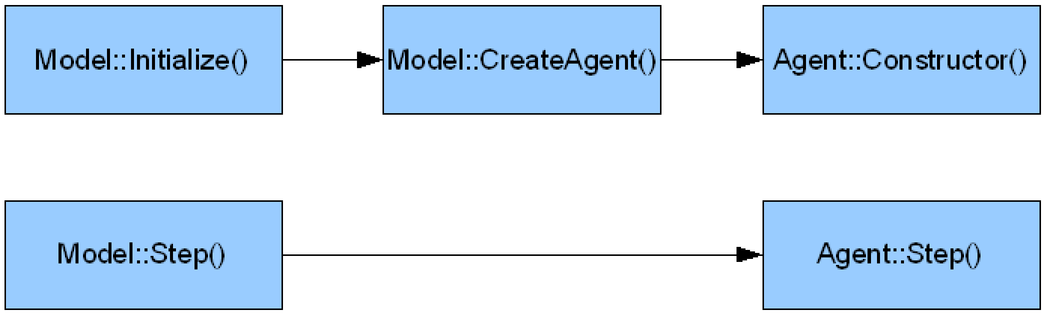


**Figure 1:** Program Execution Before Implementing Dynamic Agent Compression

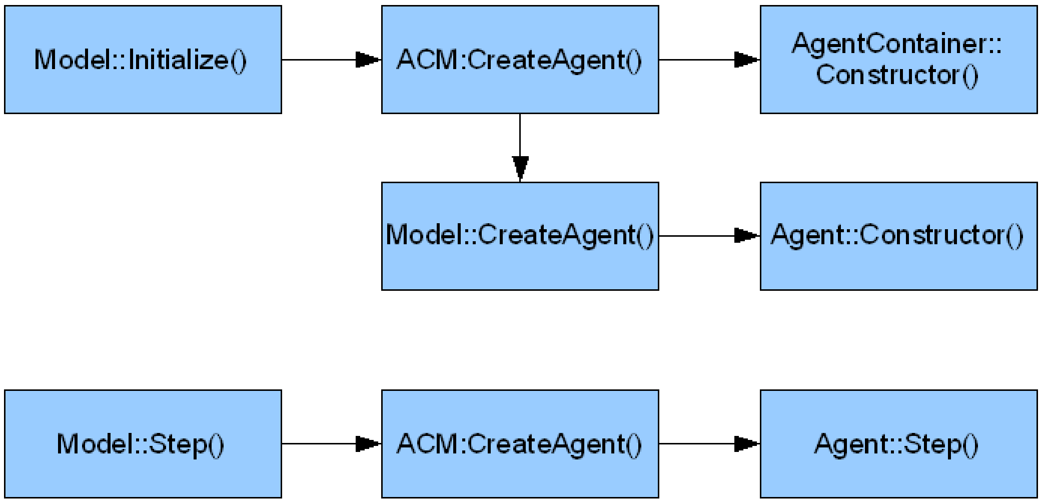


**Figure 2:** Program Execution After Implementing Dynamic Agent Compression
("ACM" = "Agent Compression Manager")

## ANALYSIS

### Lossless Dynamic Agent Compression and Hybrid Models

Lossless compression is related to and supports a general class of hybrid agent-based and systems-dynamics models. An agent-based model that uses lossless Dynamic Agent Compression is effectively a simple hybrid model, and the architecture of agent compression facilitates the development of more complex hybrids. In a systems dynamics model, stocks of some homogeneous entity are placed into distinct bins, and equations govern the flow of entities from one bin to another over time.[8] In a traditional agent-based model, agents are heterogeneous, keep their own internal state, and independently make decisions about their actions.

The Compression Manager creates the framework for both representations to coexist and feed each other in a single model.[9] Agent Containers function as the bins in a systems-dynamics model. The Compression Manager can either explicitly implement equations to handle the flow between the bins on each time tick, or it can use the self-update (i.e. step) functions within the contained Agents for the same effect. The Compression Manager also holds the collection of heterogeneous, individually instantiated agents, which are updated and queried normally by the model. As described above, the Compression Manager handles the conversion of “agents” between the systems-dynamics and agent-based settings. Finally, the model and individual agents are unaware of the internal division between the system dynamics and agent-based settings, and see only a single unified environment because of the intermediary Compression Manager.

Naturally, other implementations of the systems dynamics control logic and of the overall Dynamic Agent Compression structure are possible. This example is merely meant to serve as a proof of concept, and potentially as a starting point for further discussion.

### Lossy Compression

While we have not discussed the implementation of lossy compression in detail here, a number of observations can be made. In terms of architecture, lossy compression is not fundamentally different from lossless compression. Instead of requiring that compressed agents be identical, the Compression Manager would have to implement a clustering algorithm on the multi-dimensional attribute space. It could perform a thorough analysis at the initialization of the model, then use simpler cluster-boundary conditions to handle the compression and decompression of agents during the execution of the model (perhaps augmented by periodic re-

[8] One could consider the “agents” in a systems-dynamics model to be the stocks themselves, or the entities within the stocks. While in reality, we implement systems-dynamics stocks as Agent Containers, it helps to understand the relationship between the systems-dynamics and agent-based portions of the architecture if one thinks of the entities within the stocks as agents.

[9] The co-existence and intercommunication of an agent-based sub-model and system-dynamics model is relatively straightforward, and has been implemented in packages such as AnyLogic (XJ Technologies 2006). However, the transition of agents between the two worlds, and the transparent nature of this process to the modeler is what makes Dynamic Agent Compression unique.

analysis). The Compression Manager can utilize existing clustering algorithms on the agent-parameter space (e.g. Stage *et al. 1993*), or implement new techniques.

While the architecture is straightforward, the benefits of lossy compression are much less clear. On the one hand, clustering algorithms would allow for a more efficient representation of the agents. On the other hand, clustering would require more overhead and would obviously affect model behavior. Much of the burden for addressing this problem rests with the generic clustering algorithm. A good clustering algorithm will maximize variation between clusters and minimize variation within clusters, thus sacrificing fewer agent details. However, generic clustering algorithms cannot guarantee that the details they discard are unimportant to the model. The more information the modeler can supply to the clustering algorithm, the better. At a basic level, the modeler can limit model distortion by decreasing the compression level, by flagging selected variables as "never compressible," or by choosing lossless compression. At a more advanced level, the modeler can customize the compression algorithm using detailed knowledge of the situation.

While there may not be a way to completely eliminate the impact of lossy Dynamic Agent Compression on model behavior, there are methods to quantify and evaluate its effects. As with ordinary model parameters, those for Dynamic Agent Compression should be evaluated through a rigorous sensitivity analysis.[10] For example, in a Computational Laboratory setting (Dibble 2006), researchers could evaluate the effects of changing the level of compression or of excluding particular agent attributes from the compression process. This supports informed choices for the use and calibration of Dynamic Agent Compression.

Nonetheless, it is important to put the consequences of lossy Dynamic Agent Compression in context. When lossy Agent Compression is not used (and lossless compression is not feasible), modelers of large-scale agent based systems are often forced to manage their resource requirements by decreasing the total number of simulations used in their study, decreasing the number of agents in their model, or defaulting to common "super agent" approaches (Rose 1993). When researchers compensate for slower speeds by running fewer simulations in order to present timely results, they sacrifice thorough and rigorous exploration of the model's behavior. Similarly, limiting the number of agents purely in response to limitations of time or computational resources risks naively running the model with too few agents to elicit key effects. Alternatively, the use of super agent abstractions implicitly imposes static agent compression without providing opportunities to evaluate associated tradeoffs. Dynamic Agent Compression and Computational Laboratories support fully informed evaluation of tradeoffs.

## Overuse of Scaling Techniques

A less obvious pitfall of Dynamic Agent Compression is that it can provide a tempting solution where none is needed and may actually be detrimental. Many modelers naively presume that models require a strict one-to-one relationship between the number of agents in their model and the number of entities in the domain of study. In reality, many models do not

[10] Following similar robustness analyses that modelers already apply to input parameters, a la Steven Bankes's use of a Latin Hyber-cube to sample and analyze parameters (see www.evolvinglogic.com/el_news.html for examples).

require a one-to-one relationship because their behavior stabilizes after a certain number of agents are added to the population, beyond which the fundamental characteristics of system's behavior remain fundamentally unchanged (or scale in an easily predictable manner) as the number of agents increases further. Using Dynamic Agent Compression to unnecessarily scale a model up to a one-to-one relationship wastes programming effort, computational resources, and analysis time.

But how is a modeler to know that the system stabilizes at a certain threshold of agents such that one-to-one scaling is unnecessary? In some cases, this can be determined analytically. Yet in other situations the only way to evaluate model sensitivity with respect to the number of agents is to take the model up to the full "ideal" number of agents in order to determine the models' threshold of stability (if any). The modeler can use a simple implementation of Dynamic Agent Compression in order to test the model's behavior with larger numbers of agents. If it turns out the model requires the increased number of agents, then the modeler can customize the Dynamic Agent Compression tool to improve its efficiency. If large numbers of agents are not necessary, then the generic Dynamic Agent Compression tool has answered a valuable question, and saved resources for all future simulations.

## CONCLUSION

Dynamic Agent Compression can help agent-based modelers decrease the memory and computational resources required for certain models. While this approach is ideally suited for improving the efficiency and scalability of models in which a large number of homogeneous agents co-exist with a smaller number of unique agents, several complementary advantages arise. For example, Dynamic Agent Compression facilitates the creation of hybrid models with systems-dynamics and traditional agent-based components, and thus provides modelers with improved flexibility for model designs. We have also noted that a solid understanding of the operation of a model is advisable before exhaustively working to scale up to very large numbers of agents. Dynamic Agent Compression can help to evaluate the importance of such scaling.

Nonetheless, the discussion provided here is simply a beginning. Extensions could include a comprehensive analysis of the relative efficiency and tradeoffs of Dynamic Agent Compression versus alternative methods. We have briefly discussed the implications of lossy compression, but significant further research is needed before pursuing this perilous yet potentially rewarding path. Finally, the use of Dynamic Agent Compression in emergent multi-scale modeling seems especially promising.

## ACKNOWLEDGMENTS

Steve Wendel would like to thank Charles Macal for his presentation and discussion at NAACSOS 2006 (Macal 2006), which helped to inspire this paper. For financial support, we thank the Office of Naval Research, Grant N000140310062.

## REFERENCES

Dibble, C., 2006, “Computational Laboratories for Spatial Agent-Based Models,” in *Handbook of Computational Economics, Volume 2: Agent-Based Computational Economics*, edited by Leigh Tesfatsion and Kenneth L. Judd, Handbooks in Economics Series, Elsevier/North-Holland.

Hellweger F.L. and Kianirad E., 2006, “Spatially Explicit Individual-Based Modeling: Global vs. Local Fixed Agent Number Methods,” Presentation at SwarmFest 2006, Notre Dame, Indiana.

Macal C. M., 2006, “Methodological Comparison of Agent-Based and System Dynamics Simulation,” Presentation at the North American Association for Computational Social and Organization Sciences conference (NAACSOS), Notre Dame, Indiana.

Rose, K.A., S.W. Christensen, and D.L. DeAngelis, 1993, “Individual-based modeling of populations with high mortality: A new method based on following a fixed number of model individuals,” *Ecological Modelling* **68**(3-4):273-292.

Scheffer, M., J.M. Baveco, D.L. DeAngelis, K.A. Rose, and E.H. Nes, 1995, “Super-individuals a simple solution for modelling large populations on an individual basis,” *Ecological Modelling* **80**(2):161-170.

Servat D., E. Perrier, J.P. Treuil, and A. Drogoul, 1998, “When Agents Emerge from Agents: Introducing Multi-scale Viewpoints in Multi-agent Simulations,” *Lecture Notes in Computer Science* **1534**:183-198.

Stage, A.R., N.L. Crookston, and R.A. Monserud, 1993, “An aggregation algorithm for increasing the efficiency of population models,” *Ecological Modelling* **68**(3-4):257-271.

XJ Technologies, 2006, “AnyLogic Overview,” available at http://www.xjtek.com/anylogic/

Wikipedia, 2006, “Lossy Data Compression,” available at http://en.wikipedia.org/wiki/Lossy_data_compression

# POLYAGENTS MODEL MULTIPLE FUTURES CONCURRENTLY

H.V. PARUNAK,[*] NewVectors LLC, Ann Arbor, MI
S. BRUECKNER, NewVectors LLC, Ann Arbor, MI

## ABSTRACT

Modeling domain entities with agents reveals many aspects of system dynamics and interactions that other modeling techniques do not. However, an agent executes only one trajectory per run, and so does not capture the alternative trajectories accessible in the evolution of any realistic system. Averaging over multiple runs does not show the range of individual interactions. We address these problems with a new modeling construct, the *polyagent*, which represents each entity with a single persistent *avatar* supported by a swarm of transient *ghosts*. Each ghost interacts with the ghosts of other avatars through digital pheromone fields, capturing a wide range of alternative trajectories in a single run of the system that can proceed faster than real time.

**Keywords:** Multiple futures, uncertainty, BDI, swarm intelligence, digital pheromones, dynamics.

## INTRODUCTION

The fundamental entity in an agent-based model is the agent, corresponding to a discrete entity in the domain being modeled, and the fundamental operator is the interaction among agent behaviors. The fundamental entity in an equation-based model (Sterman 2000) is some system observable, and the fundamental operator is the evolution of that observable (e.g., by a differential equation).

Agent-based models often map more naturally to the structure of a problem than equation-based models, have more natural representations, and provide more realistic results (Parunak, Savit et al. 1998), but suffer an important shortcoming. The observables in an equation-based model are often averages across agents, so the model captures agent variation (although at an aggregate level). Each agent in an agent-based model executes only one trajectory per run of the system, and so does not capture the alternative trajectories accessible to its entity in the evolution of any realistic system. Good modeling practice requires averaging over multiple runs, but this approach still does not capture the range of individual interactions.

The *polyagent* modeling construct represents each entity with a single persistent *avatar* supported by a swarm of transient *ghosts*. Each ghost interacts with those of other avatars through digital pheromone fields, capturing many alternative trajectories in a single run. We have used this approach in several systems. This paper articulates the polyagent as a modeling construct and provides some guidance concerning its use.

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

## THE CHALLENGE OF MODELING MULTI-AGENT INTERACTIONS

Consider $n + 1$ entities. At each step, each entity interacts with one of the others. Thus at time $t$ its interaction history $h(t)$ is a string in $n^t$. Its behavior is a function of $h(t)$. This toy model generalizes many domains, including predator-prey systems, combat, innovation, diffusion of ideas, and disease propagation.

One simulation of such a system gives limited information.

1. We may have imperfect knowledge of the agents' internal states or the environment. If we change our assumptions about these unknown details, the agents' behaviors are likely to change.

2. The agents may behave non-deterministically: their perceptions may contain noise, and their decisions may be stochastic.

3. Even if the agents' reasoning and interactions were deterministic and we had accurate knowledge of all state variables, nonlinearities can result in chaotic dynamics: tiny differences in state variables can lead to arbitrarily large behavioral divergences.

An equation-based model typically tracks aggregate or average observables across the population (e.g., for predators and prey, predator population, prey population, average predator energy level, or average prey energy level) as functions of time, and does not model the trajectory of a single entity.

An agent-based model explicitly generates each agent's trajectory. In our model, over $\tau$ steps, each entity experiences one of $n^\tau$ possible histories. The population of $n + 1$ entities samples only $n + 1$ of these histories.

Such models must be run many times to explore the possible outcomes. Each run samples only one set of possible interactions. For large populations and scenarios that support alternative interactions, the number of runs needed to sample the possible interactions thoroughly is prohibitive, and the fraction of possible histories actually sampled by a single run is vanishingly small. Polyagents can capture the outcome of multiple possible interactions in a single run.

## TOWARD A THEORY OF POLYAGENTS

What is a polyagent, and how is it related to previous work?

### Understanding a Polyagent

A polyagent represents each entity with multiple agents: a single avatar that links it to the entity, and a swarm of ghosts that explore its alternative behaviors.

The avatar persists as long as its entity is active, and maintains its entity's state. It may use sophisticated reasoning. Each avatar generates a stream of ghosts. Ghosts die after a fixed period of time or after some defined event. Each avatar controls the rate it generates ghosts, and typically has several concurrent ghosts.

Ghosts explore alternative behaviors for their avatar. They are computationally simple, and interact through a digital pheromone field, a vector of scalars that depends on both location and time. Each ghost chooses its actions stochastically based on a weighted function of nearby pheromones, and optionally deposits its own pheromone. A ghost's "program" is the vector of weights.

Ghosts multiply the number of interactions that a single run of the system can explore. Instead of one trajectory for each avatar, we now have one trajectory for each ghost. If each avatar has $k$ concurrent ghosts, we explore $k$ trajectories concurrently. Pheromone dynamics in the environment make the multiplication even greater (Brueckner 2000):

1. The environment *aggregates* deposits from individual agents, fusing information across multiple agents, enabling a single ghost to interact with multiple other ghosts at the same time.

2. It *evaporates* pheromones, a form of truth maintenance. Traditional knowledge bases remember everything they learn unless they have a reason to forget. Detecting resulting inconsistencies is NP-complete. Pheromones forget what they learn, unless it is reinforced, so inconsistencies purge themselves.

3. It *propagates* pheromones, disseminating information to nearby agents.

If $n$ avatars deposit pheromones, each ghost's actions are influenced by up to $n$ other agents (depending on the propagation radius), so we are exploring in effect $nk$ interactions for each entity, or $n^2k$ interactions overall. If individual ghosts deposit pheromones, the number of interactions explored is even greater, on the order of $k^n$. The detail of interactions is not as great as in a conventional multi-agent model. But experience shows that the fidelity is adequate for the problems we have addressed, and pheromone-based interaction is computationally efficient.

The avatar can

- modulate the number of its ghosts, their rate of generation, and the distribution of their parameters to control the exploration of alternative futures;

- evolve them to learn the best parameters for a given situation;

- review their behavior to estimate its own future experience.

## Comparison with Previous Work (Table 1)

Traditionally, distinct agents model different functions of a single domain entity. In a polyagent, all ghosts have the same function: to explore one possible behavior of the domain entity. Multiple ghosts provide, not functional decomposition, but a range of estimates of alternative behaviors.

**TABLE 1** Comparing the polyagent with other technologies

| | Multiple agents per domain entity | Avatar/Ghost dualism | Parallel search of alternative behaviors | Parallel search of multiple possible interactions |
|---|---|---|---|---|
| Polyagent | X | X | X | X |
| Functionally distinct agents | X | | | |
| Evolutionary computation | X | | X | |
| Fictitious play | X | | X | |
| Ant colony optimization | X | | X | X |
| Kijima's polyagents | | | | |
| Polyagent therapies | | | | |

Many forms of evolutionary computation (Jacob 2001) execute multiple representatives of an entity concurrently. Each agent samples only one series of interactions with other entities in the domain. The polyagents' pheromone field permits each ghost to adjust its behavior based on multiple alternative behaviors of other entities.

The multiple behaviors contemplated in fictitious play (Lambert, Epelman et al. 2005) take place against a static model of the rest of the world.

Ant-colony optimization (Dorigo and Stuetzle 2004) uses pheromones to integrate the experiences of parallel searchers. The polyagent's innovation is an avatar that manages the searchers representing a single domain entity.

The term "polyagent" expresses this novel construct of several software agents collectively representing a domain entity and its alternative behaviors. The term is used in two other contexts. In medicine, "polyagent therapy" uses multiple treatment agents (notably, multiple drugs in chemotherapy). Closer to our domain, but still distinct, is the use of the term (Kijima 2001) to describe a game-theoretic approach to analyzing the social and organizational interactions of multiple decision-makers. Kijima's "poly-agent" makes sense only as a description of a system, and does not describe a single modeling construct, as does our term.

## EXAMPLES OF POLYAGENTS

Polyagents are a rationalization of techniques we have used in several applications.

### Factory Scheduling

Our earliest application of polyagents did real-time job-shop scheduling (Brueckner 2000) with three species of agents: processing resources, parts, and policy agents. Avatars of processing resources with different capabilities and capacities and avatars of parts with changing processing needs (due to rework) coordinate to optimize material flow through a complex, high-volume manufacturing transport system. Only part avatars deploy ghosts. Policy agents and resource (machine) avatars are traditional single agents.

In a job shop, parts interact by occupying resources, blocking access by other parts. Depending on the schedule, different parts may interact, and in different orders. Polyagents explore the space of alternative part routings and alternative interactions concurrently in a single model.

Part avatars continuously deploy ghosts that move through successive decision points in the manufacturing process. Each decision is stochastic, based on attractive pheromones around the next decision point. Policy agents deposit these pheromones to balance the material flow across the transport network, then ghosts modulate them.

The ghosts that avatars emit travel into the future without the delay imposed by physical part transport and processing. These ghosts may remain within the part's current context, or they may emulate the probabilistic outcome of a processing step and assume a new state for their part. In either case, the ghosts contribute to a pheromone field that reports the currently predicted load of parts across material handling stations. When ghosts for alternative parts explore the same resource, they interact through their pheromones.

Each ghost's stochastic decisions generate an alternative routing for its avatar. The pheromone field to which it responds has been modulated by the ghosts of other parts, and represents multiple routings of those parts. Thus each part's ghosts explore both alternative futures for that part, and alternative interactions with other parts.

Policy agents are informed either by humans or by other agents of the desired load of parts of specific states at a particular location. They deposit attractive or repulsive pheromones. Thus, through a local process, multiple policy agents supported by the flow of ghosts adapt the levels of attractive or repulsive pheromone deposits to shape the future flow of material.

By the time the avatar, delayed by physical movement constraints, makes its next routing choice, the ghosts and policy agents have adjusted the pheromone concentrations so that the avatar makes the "right" decision. The policy agents and the ghosts control the behavior of the avatar by converging on a low-entropy pheromone concentration that the avatar can sample.

## Path Planning for Robotic Vehicles

Robotic vehicles must continuously replan their paths, as their knowledge of the environment changes due to limited sensor range and environmental change. In military applications, vehicles must navigate dynamically changing sets of targets and threats.

Ants solve a similar problem in forming paths between nests and food sources (Parunak 1997). Ants searching for food deposit nest pheromone while climbing the food pheromone gradient left by successful foragers. Ants who find food deposit food pheromone while climbing the nest pheromone gradient left by outbound ants. The pheromone fields collapse into a path as the ants interact.

This algorithm depends on many ants exploring different paths, while a vehicle only traverses its path once. So we use a polyagent to represent the vehicle (Sauter, Matthews et al. 2005). As the avatar moves, it continuously emits ghosts, whose interactions continuously (re)form the path in front of the avatar. These ghosts seek targets and then return to the avatar, responding to several digital pheromones:

- *RTarget* is emitted by a target.
- *GNest* is emitted by a ghost that has left the avatar and is seeking a target.
- *GTarget* is emitted by a ghost that has encountered a target and is returning to the avatar.
- *RThreat* is emitted by a threat.

In general, the ghosts are attracted to *RTarget* pheromone and repelled from *RThreat* pheromone. While they have not found a target, they are attracted to *GTarget* pheromone. Once they have found a target, they are attracted to *GNest* pheromone. A ghost senses the relative strengths of these quantities in its current cell and each neighboring cell in a hexagonal lattice, weights them to compute a value for each cell, then selects its next cell with probability proportional to its value.

Each ghost explores one possible route. The avatar performs two functions.

1. It *integrates* the information from ghosts into a single route for the robot. *GTarget* pheromone is deposited only by ghosts that have found the target, and its strength shows how many ghosts that traversed that cell on their way home from the target. So the aggregate pheromone strength estimates the likelihood that a cell is on a reasonable path to the target.

2. It *modulates* ghost behavior by adjusting the weights that ghosts use to combine pheromones. In our initial implementation, all ghosts used the same weights, and their paths differed only because they chose successive steps stochastically. When the avatar randomly varied the weights around the hand-tuned values, performance improved by over 50%, because the ghosts could explore more routes. Then the avatar evolved the vector of weights during system operation, and performance improved nearly an order of magnitude over hand-tuned ghosts (Sauter, Matthews et al. 2002).

We tested this system's ability to route an aircraft in simulated combat (Parunak, Brueckner et al. 2004).

In one example, it found a path to a target through a threat gauntlet (Figure 1). A centralized planner that integrated a loss function and climbed the gradient could not solve this problem without introducing an intermediate goal at the gauntlet's entrance. The polyagent succeeded because some ghosts wandered into the gauntlet, laying pheromones to guide other ghosts.

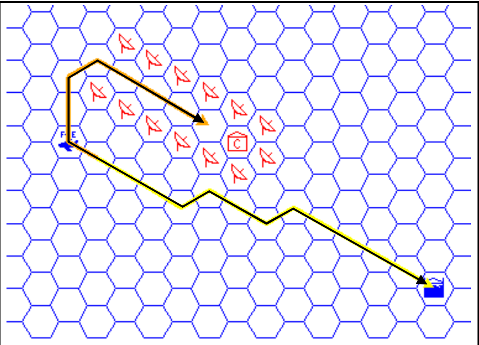


**FIGURE 1** Gauntlet routing problem

In another experiment, we compared the total surviving strength of the Red and Blue forces in missions over a changing landscape of threats and targets, in two different configurations. The polyagent's ability to deal with partial but up-to-date knowledge both inflicted more damage on the adversary and offered higher survivability than an avatar guided by preplanned scripts based on complete initial information.

This application shows how a polyagent can explore alternative behaviors concurrently, and integrate that experience into a single course of action. Since only one polyagent was active at a time, this work does not use polyagents' ability to manage the exploding space of possible interactions.

## Characterizing and Predicting Agent Behavior

We use polyagents to evolve a model of each real-world entity (a group of soldiers known as a fire team) in urban combat (Kott 2004) and predict its future behavior. Figure 2 shows the process. Ghosts live on a timeline of discrete pages indexed by $\tau$ (distinct from real time $t$) that begins in the past and runs into the future. The avatar inserts ghosts at the insertion horizon (say $\tau - t = -30$, the state of the world 30 minutes ago), sampling each ghost's parameters to explore alternative personalities of its entity. The avatar also estimates its entity's goals (using a belief network) and instantiates them as pheromone sources that attract the ghosts.

The avatars record pheromones representing the observed state of the world on each page between the insertion horizon and $\tau = t$. The inserted ghosts interact with this past state. Their fitness depends not just on their own actions, but also on the behaviors of the rest of the population, which is also evolving. $\tau$ advances faster than real time, so eventually $\tau = t$, when the avatar compares each ghost with its entity's actual state.

The fittest ghosts have three functions.

1. Their personality estimates the personality of the corresponding entity.

2. They breed, and their offspring reenter at the insertion horizon.

3. They run into the future, exploring possible futures of the battle that the avatar analyzes to predict enemy behavior and recommend friendly behavior. In the future, the pheromone field is generated by other ghosts rather than avatars. Thus it integrates the various futures that the system is considering, and each ghost interacts with this composite view of other entities.

The first and third functions are analogous to the integrating function of the avatars in the route planning problem, while the second is analogous to the modulation function.

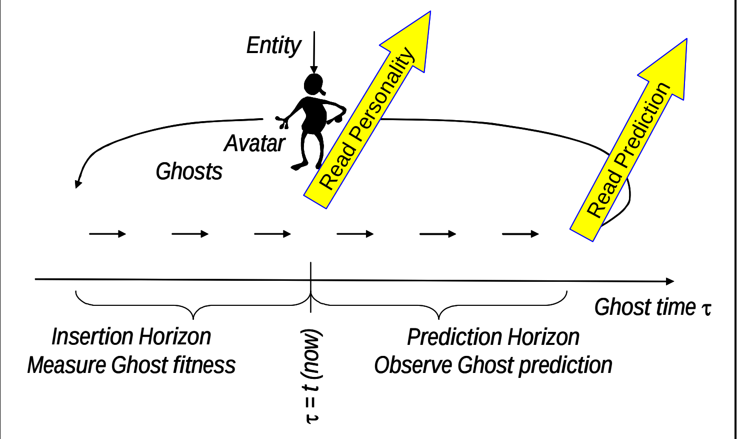


**FIGURE 2** Behavioral emulation and extrapolation

This system has successfully characterized the internal state of entities that we can only observe externally, and predicted their future behavior, in experimental wargames with human participants (Parunak, Brueckner et al. 2006). We can detect entities' emotions as well as a human observer, but faster. Our prediction of the future is also superior to a human's.

## DISCUSSION

Several features of the polyagent modeling construct deserve recognition.

- Multiple ghosts concurrently *explore* alternative behaviors of the domain entity.
- The ghosts *interact* through a digital pheromone field that combines multiple possible interactions among the entities.
- A single, possibly more complex, avatar *modulates* the swarm of ghosts.
- The avatar also *integrates* the behaviors of its ghosts (either directly or by observing the pheromones they deposit) to estimate the domain entity's likely behavior.

The strength of a pheromone field depends (*inter alia*) on the frequency with which agents of a given type visit various locations. If those agents are ghosts representing an entity's alternative futures, the field may be interpreted in terms of the likelihood of different future states. An analogous situation arises in quantum mechanics (Feynman and Hibbs 1965). Table 2 suggests several parallels between polyagents and quantum physics. In the spirit of our earlier work applying metaphors from theoretical physics to understanding multi-agent systems (Parunak, Brueckner et al. 2004), we are exploring how quantum mechanics may provide useful metaphors for engineering polyagent systems and interpreting their behavior.

## ACKNOWLEDGMENTS

This material is based in part upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract Nos. F3062-99-C-0202 and NBCHC040153. 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 DARPA or the Department of

**TABLE 2** Parallels between quantum physics and polyagents

| Quantum Physics | Polyagents |
|---|---|
| Duality between (single, localized) particle and (distributed) wave function | Duality between (single, localized) avatar and (distributed) swarm of ghosts |
| Interactions among wave functions' amplitude fields model interactions among particles | Ghosts' pheromone fields model interactions of agents |
| Wave function captures a range of possible behaviors | Swarm of ghosts captures a range of possible behaviors |
| Observation collapses the wave function to a single behavior | Avatar interprets the aggregate behavior of the ghosts and yields a single prediction of behavior |

Interior-National Business Center (DOI-NBC). Distribution Statement “A” (Approved for Public Release, Distribution Unlimited).

## REFERENCES

Brueckner, S., 2000, Return from the Ant: Synthetic Ecosystems for Manufacturing Control. Thesis, Department of Computer Science, Humboldt University Berlin, Berlin, Germany, http://dochost.rz.hu-berlin.de/dissertationen/brueckner-sven-2000-06-21/PDF/Brueckner.pdf.

Dorigo, M. and T. Stuetzle, 2004, *Ant Colony Optimization*. Cambridge, MA, MIT Press.

Feynman, R. and A. R. Hibbs, 1965, *Quantum Mechanics and Path Integrals*, McGraw-Hill.

Jacob, C., 2001, *Illustrating Evolutionary Computation With Mathematica*. San Francisco, Morgan Kaufmann.

Kijima, K., 2001, "Why Stratification of Networks Emerges in Innovative Society: Intelligent Poly-Agent Systems Approach." *Computational and Mathematical Organization Theory* **7**(1 (June)): 45-62.

Kott, A., 2004, "Real-Time Adversarial Intelligence & Decision Making (RAID)." Retrieved 20 January, 2005, from http://dtsn.darpa.mil/ixo/programdetail.asp?progid=57.

Lambert, T. J., III, M. A. Epelman, et al., 2005, "A Fictitious Play Approach to Large-Scale Optimization." *Operations Research* **53**(3 (May-June)).

Parunak, H. V. D., 1997, "’Go to the Ant’: Engineering Principles from Natural Agent Systems." *Annals of Operations Research* **75**: 69-101. http://www.newvectors.net/staff/parunakv/gotoant.pdf.

Parunak, H. V. D., S. Brueckner, et al., 2006, *Real-Time Evolutionary Agent Characterization and Prediction*. Social Agents: Results and Prospects (Agent 2006), Chicago, IL, Argonne National Laboratory.

Parunak, H. V. D., S. Brueckner, et al., 2004, *Digital Pheromones for Coordination of Unmanned Vehicles*. Workshop on Environments for Multi-Agent Systems (E4MAS 2004), New York, NY, Springer. 246-263. http://www.newvectors.net/staff/parunakv/E4MAS04_UAVCoordination.pdf.

Parunak, H. V. D., S. Brueckner, et al., 2004, *Universality in Multi-Agent Systems*. Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2004), New York, NY, ACM. 930-937. http://www.newvectors.net/staff/parunakv/AAMAS04Universality.pdf.

Parunak, H. V. D., R. Savit, et al., 1998, *Agent-Based Modeling vs. Equation-Based Modeling: A Case Study and Users' Guide*. Multi-agent systems and Agent-based Simulation

(MABS'98), Paris, FR, Springer. 10-25. http://www.newvectors.net/staff/parunakv/mabs98.pdf.

Sauter, J. A., R. Matthews, et al., 2002, *Evolving Adaptive Pheromone Path Planning Mechanisms*. Autonomous Agents and Multi-Agent Systems (AAMAS02), Bologna, Italy, ACM. 434-440. www.newvectors.net/staff/parunakv/AAMAS02Evolution.pdf.

Sauter, J. A., R. Matthews, et al., 2005, *Performance of Digital Pheromones for Swarming Vehicle Control*. Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems, Utrecht, Netherlands, ACM. 903-910. http://www.newvectors.net/staff/parunakv/AAMAS05SwarmingDemo.pdf.

Sterman, J., 2000, *Business Dynamics*. New York, NY, McGraw-Hill.

# WHO'S YOUR NEIGHBOR? NEIGHBOR IDENTIFICATION FOR AGENT-BASED MODELING

C.M. MACAL,* Argonne National Laboratory, Argonne, IL,
and The University of Chicago, Chicago, IL
T.R. HOWE, Argonne National Laboratory, Argonne, IL

## ABSTRACT

Agent-based modeling and simulation, based on the cellular automata paradigm, is an approach to modeling complex systems comprised of interacting autonomous agents. Open questions in agent-based simulation focus on scale-up issues encountered in simulating large numbers of agents. Specifically, how many agents can be included in a workable agent-based simulation? One of the basic tenets of agent-based modeling and simulation is that agents only interact and exchange locally available information with other agents located in their immediate proximity or neighborhood of the space in which the agents are situated. Generally, an agent's set of neighbors changes rapidly as a simulation proceeds through time and as the agents move through space. Depending on the topology defined for agent interactions, proximity may be defined by spatial distance for continuous space, adjacency for grid cells (as in cellular automata), or by connectivity in social networks. Identifying an agent's neighbors is a particularly time-consuming computational task and can dominate the computational effort in a simulation. Two challenges in agent simulation are (1) efficiently representing an agent's neighborhood and the neighbors in it and (2) efficiently identifying an agent's neighbors at any time in the simulation. These problems are addressed differently for different agent interaction topologies. While efficient approaches have been identified for agent neighborhood representation and neighbor identification for agents on a lattice with general neighborhood configurations, other techniques must be used when agents are able to move freely in space. Techniques for the analysis and representation of spatial data are applicable to the agent neighbor identification problem. This paper extends agent neighborhood simulation techniques from the lattice topology to continuous space, specifically $R^2$. Algorithms based on hierarchical (quad trees) or non-hierarchical data structures (grid cells) are theoretically efficient. We explore implementing hierarchical and non-hierarchical data structures by using efficient implementations that are designed to address spatial data specifically in the context of agent-based simulation. The algorithms are evaluated and compared according to computation times for neighborhood creation, neighbor identification, and agent updating.

**Keywords:** Agent-based model, neighbor identification, proximity detection, spatial data structure, quad tree, computational complexity

---

* *Corresponding author address*: Charles M. Macal, Decision and Information Sciences Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439-4832; email: macal@anl.gov.

## INTRODUCTION

Agent-based modeling and simulation (ABMS), based on the cellular automata paradigm, is an approach to modeling complex systems comprised of interacting autonomous agents (Macal and North 2005). Agent behaviors are modeled explicitly, using a range of behavioral models and representation schemes at appropriate levels of detail. ABMS promises to have far-reaching effects on the way that researchers use electronic laboratories to do their research and businesses use computers to support decision-making. Computational advances make possible a growing number of agent-based applications across many fields. Applications range from modeling agent behavior in the stock market and supply chains to predicting the spread of epidemics and the threat of bio-warfare; from modeling the growth and decline of ancient civilizations to modeling the complexities of the human immune system, and many more.

Open questions for agent-based simulation focus on scale-up issues encountered in simulating large numbers of agents. Specifically, how many agents can be included in a workable agent-based simulation? One of the basic tenets of agent-based modeling and simulation is that agents interact and exchange locally available information with other agents located only in their immediate proximity or neighborhood of the space in which the agents are situated. Generally, an agent's set of neighbors changes rapidly as a simulation proceeds through time and as agents move through space. Depending on the topology defined for agent interactions, proximity may be defined by spatial distance for continuous space, adjacency for grid cells (as in cellular automata), or by connectivity in social networks. Identifying an agent's neighbors is a particularly time-consuming computational task and can dominate the computational effort in a simulation. Two challenges in agent simulation are (1) efficiently representing an agent's neighborhood and the neighbors in it and (2) efficiently identifying an agent's neighbors at any time in the simulation. These problems are addressed differently for different agent interaction topologies. Efficient approaches have been developed for agent neighborhood representation and neighbor identification for agents on a lattice with general neighborhood configurations. Other techniques must be used when agents are able to move freely in space.

This paper extends agent neighborhood simulation techniques from the lattice topology to continuous space, specifically $R^2$. We explore implementing hierarchical and non-hierarchical data structures using efficient implementations that are designed to address spatial data specifically in the context of agent-based simulation. We also evaluate and compare the algorithms.

This paper is organized as follows. Section 2 frames the agent neighbor identification problem and reviews related work. Section 3 describes the experimental design for comparing the algorithms. Section 4 presents the study results and compares performance of the algorithms. Section 5 summarizes the findings and draws conclusions on the implications for agent-based modeling.

## STATEMENT OF THE PROBLEM

The agent neighbor problem is one of determining for each agent $a$, the set of agent neighbors $N_a$ of $a$ such that for any neighbor $n \varepsilon N_a$, $\| \text{loc}(a) - \text{loc}(n) \| < d_a$, where loc(x) is the location of x and $d_a$ is the maximum distance between $a$ and $n$ defining a neighborhood under some selected norm. In general, the neighborhood size defined by the radius measure $d_a$ varies by agent. Agent $a$ may or may not be an element of $N_a$, but this depends on the use of the neighborhood in the agent simulation. For example, in a social influence model, agents are influenced by their neighbors, and therefore an agent's neighborhood does not include itself. Alternatively, in a model in which each agent considers the mass of agents in its neighborhood and its influence on the larger system, an agent's neighborhood would include itself. Generally, neighbor proximity algorithms return neighborhoods that include the agent itself, and it adds computational effort to exclude the agent if this is necessary. In what follows, we will assume that an agent's neighborhood includes itself. Some observations about neighborhoods that hold true throughout the course of a simulation are that:

$1 \leq \sum_a | N_a | \leq \mathrm{N}^2$, where $| S |$ is the cardinality of set $S$

$\sum_a | N_a | \rightarrow \mathrm{N}^2$ as $d_a \rightarrow \infty$

$\sum_a | N_a | \rightarrow 1$ as $d_a \rightarrow \varepsilon > 0$

The general strategy for identifying an agent's neighbors is to first select a subset of agents as candidate neighbors for evaluation in such a way that the subset includes the agent's actual neighbors and then to evaluate the candidates on the proximity criteria.

The agent neighborhood problem is part of a larger problem of updating an agent's attributes based on the attributes of its neighbors. For example, in the Boids model (Reynolds 1987, 2006), an agent's velocity is based on the velocities of the agent's neighbors, which excludes the velocity of the agent itself. The general steps of updating agents' based on their neighbors at any point in a simulation can be broken down into neighborhood creation, neighbor identification, and agent updating:

1. *Neighborhood Creation:* Create a neighborhood data structure for all agents.

2. *Neighbor Identification:*

   2.1 For each agent, select candidate neighbors.

   2.2 For each agent, evaluate whether the candidates are in the agent's neighborhood.

3. *Agent Updating:* For each agent, update the agent according to who are its neighbors.

The computational complexity (Aho et al. 1976; Knuth 1998) for neighborhood creation and neighbor identification can vary significantly depending on the algorithm employed. In addition, these two steps may be overshadowed by the time spent in the agent updating step.

In this study, we use an objective measure to compare the neighbor identification algorithms in terms of computational complexity. In general, computation times vary by the implementation method (agent toolkit or other). Our objective measure is the same for whatever implementation is used. The measure is the number of agents that are identified as candidate neighbors as compared to the number agents identified that are actual neighbors.

## Related Work

The problem of agent neighbor identification is similar to problems in other domains, including:

- Collision detection in computer simulation and visualization (Basch et al. 1997),
- Finding the nearest object or k-nearest neighbors to a query object in a spatial database (Knuth, 1998; Hjaltason and Samet 1995),
- Find the closest pair of objects in a database (Cormen et al. 2001),
- Finding a clear path through an obstacle field in robotic planning (Lewis et al. 1997),
- Mobile communications (Amir et al. 2004; Küpper and Treu 2006), and
- Aircraft proximity detection in a controlled terminal airspace (Wieland et al. 2001).

Techniques for the analysis and representation of spatial data are applicable to these problems (Samet 1990) and the algorithms implemented here are derived from spatial data representation.

Requirements for operating on spatial data objects vary across applications. For example, in general database applications, standard operations include data structure creation, object insertion, object deletion, and spatial relationship querying. The latter includes finding the objects closest to a specified query object, such as finding the city closest to a point; finding the nearest object or specified number of objects to a query object; and compound queries that include combinations of attributes, such as finding the closest city with a specified population closest to another city.

The agent neighbor identification problem has some unique characteristics, including:

1. Agents are mobile and dynamic in their locations. All agents need to determine all their neighborhoods and neighbors each time the information is required. This means the complete data structure must be created each time it is used.
2. All agents are stored in memory, so there is not a high penalty for accessing a database like there would be in a spatial database application.

In agent neighborhood determination, the requirements are somewhat different from other applications. Although the various data structure algorithms are applicable to the agent neighbor problem, the criteria against which they are evaluated are different.

## EXPERIMENTAL DESIGN

A naïve implementation of a neighbor identification algorithm consists of enumerating all agents as neighbor candidates for each agent resulting in $N^2$ comparisons, an $O(N^2)$ algorithm. Such an $O(N^2)$ algorithm can be computationally expensive when N becomes large. We implement two algorithms for neighbor identification: the grid cell method and the quad tree approach. These algorithms are based on hierarchical data structures (quad tree) or non-hierarchical data structures (grid cells, or buckets). They are theoretically more efficient, performing much better than an $O(N^2)$ algorithm. We also implement a variation of the all-neighbors approach as a basis for comparison to the two algorithms tested. All of the algorithms examined here are inclusive in the sense that they all identify sets of candidate neighbors and these include the actual neighbors. No actual neighbors are missed.

### Algorithm 1: All Neighbors Algorithm

Due to the symmetry feature of the neighbor identification problem, i.e., agent *a* is a neighbor of agent *b* if and only if agent *b* is a neighbor of agent *a*, the number of agent comparisons can be reduced to (N-1) + (N-2) + … + 1 = N (N+1)/2. This still leaves an $O(N^2)$ algorithm. In what follows, we call this procedure the all-neighbors algorithm and use it as a baseline for comparison.

### Algorithm 2: Grid Cell Algorithm

The grid cell method proceeds by assigning a grid cell of a specified size to each agent, collecting the agents in each grid cell, and then identifying candidate neighbors for each agent based on the agent's grid cell and neighboring grid cells. The entire space represented consists only of the space spanned by the agents. Cells that do not contain at least one agent are not represented. These latter two features of the algorithm suggest that it may be an efficient way to identify neighbors. Figure 1 shows the grid cell indexing and candidate neighbor determination for an agent located in the southeast quadrant of a grid cell.

In the grid cell scheme tested here, grid cells do not overlap. Therefore, the algorithm does not have to check that a candidate agent appears multiple times in a list of candidate neighbors. Other specialized grid cell schemes do allow overlap of grid cells (Samet 1990). The grid cell method is easily extended to higher dimensions with the extension of grid cell indexing. In this grid cell implementation, the neighborhood size is the same for all agents for a single simulated time step. Neighborhood size could be varied by time step. Theoretically, the algorithm could be extended for the case in which neighborhood size varied by agent type, but with greater complication. For example, in a predator-prey model, the predators may detect prey in a different range than prey are able to detect predators. In this case, modelers would need to create two complete grid cell data structures, one whose grid cells reflect the predator neighborhood size and one whose grid cells reflect the prey neighborhood size.

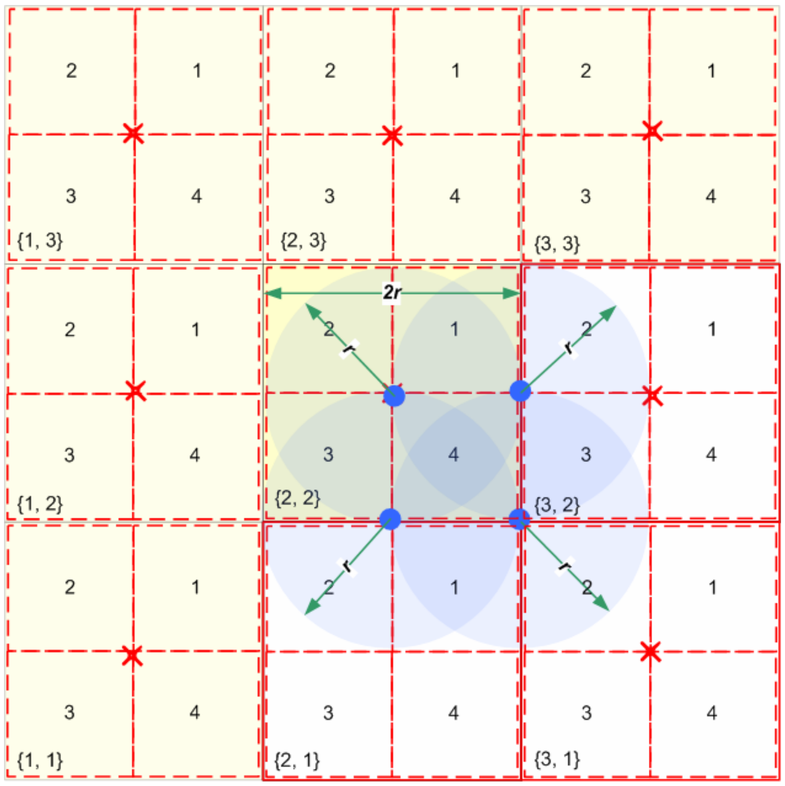


**FIGURE 1** Grid cell indexing and candidate neighbor determination for agent located in southeast quadrant (r = agent neighborhood radius)

## Algorithm 3: Quad Tree Algorithm

Quad trees came to prominence as a way to deal with range queries in database search (Finkel and Bentley 1974). Each node of a quad tree represents a rectangular area in two-dimensional space and contains one of the objects located in that area. There are four subtrees corresponding to the four quadrants of the original rectangle relative to the coordinates of the object. The subtrees are ordered, for example, NE (quadrant I in $R^2$), NW (quadrant II in $R^2$), SW (quadrant III in $R^2$), and SE (quadrant IV in $R^2$). In the quad tree representation, areas are overlapping, as the space is partitioned into four symmetric patches at each ensuing level of the hierarchy. Quad trees are also commonly used to represent spatial images as two-dimensional arrays in which the objects are pixilated patches of the same color. Traditionally, the quad tree representation is designed to minimize the search time to find an object in a given tree, and there is less regard for the time to create the tree since objects in spatial or image databases are static. In agent simulation, the quad tree must be created at each time step; therefore, there is a recurring penalty for data structure creation when used in agent neighbor identification.

Quad trees have numerous variants, including point quad trees, region quad trees, and *k-d* trees. The point quad tree is implemented here. An example of a quad tree (radial representation) for the agent neighbor problem is shown in Figure 2 and based on simulations conducted here.

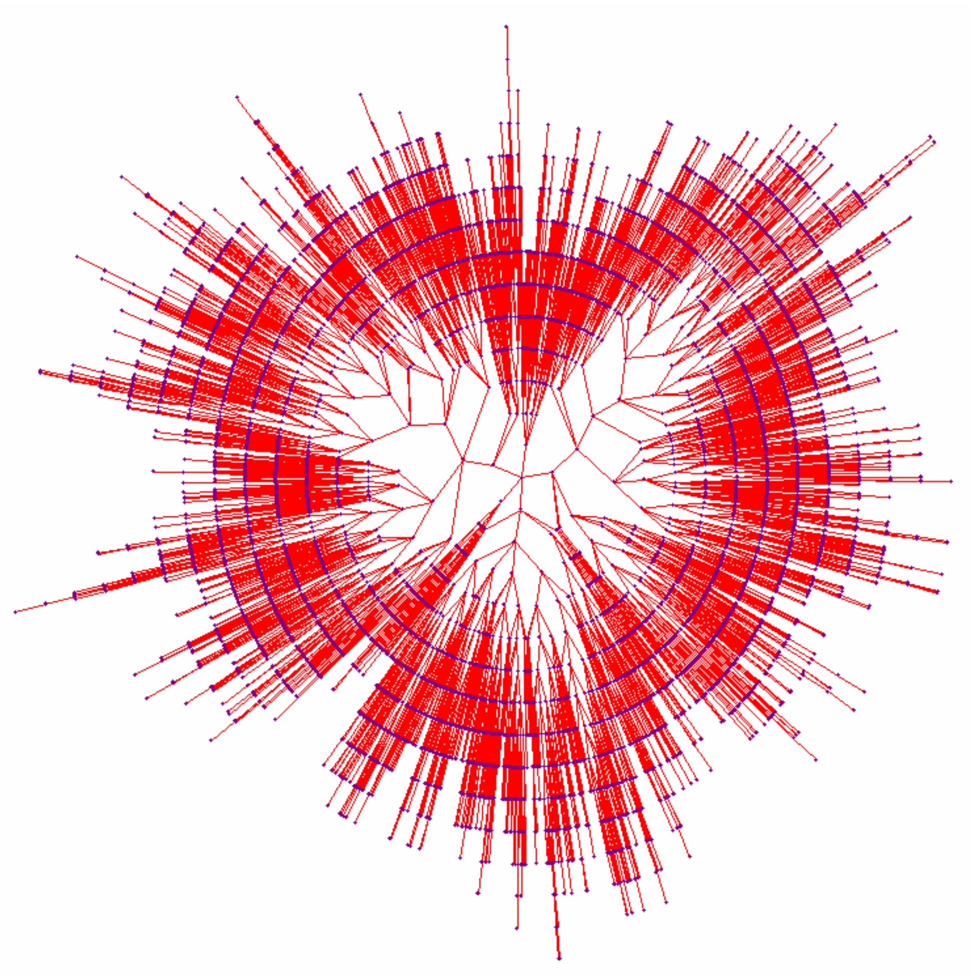


**FIGURE 2** Radial quad tree representation of agent neighborhoods for 10,240 agents Note: Maximum tree depth is 19 and maximum width is 1749

An important consideration is whether a quad tree is balanced, as the tree depth is the governing factor in the time required to use the quad tree to find an agent's neighbors. The tree structure is dependent on the order in which the agents populate the tree. The tree in Figure 2 is based on the random insertion of agents into the tree and is fairly well balanced.

In a quad tree implementation, the neighborhood size need not be the same for all agents in a single time step, unlike the grid cell method described above. The algorithm allows for neighborhood size to be an attribute of each agent.

## Implementation

All tests comparing the algorithms were done using *Mathematica* and the agent models in the Boids model. *Mathematica* is a viable platform for agent-based simulation (Gaylord and D'Andria, 1998) and is especially useful for prototype model development (Macal 2004; Macal and Howe 2005). *Mathematica* is a computational mathematics system with a large set of libraries for supporting data structures and algorithms (Maeder 2000). It is highly optimized for algorithmic and numerical processing performance.

## RESULTS

Figure 3 illustrates the dimensions of the agent neighbor problem and is presented for comparison purposes. In Figure 3, the number of operations for neighbor comparisons is illustrated and compared to Order $N^2$ and N Log N statistics. Figure 3 shows the neighbor comparisons for the All Neighbors algorithm is slightly better than $N^2$ but is still $O(N^2)$. Also, Figure 3 illustrates that the total number of actual agent neighbors grows at a rate between N Log N and $N^2$ for a problem of fixed neighborhood size, relative to the total area spanned by the agents at a given time in the simulation and agent density, as measured by the number of agent neighbors per agent.

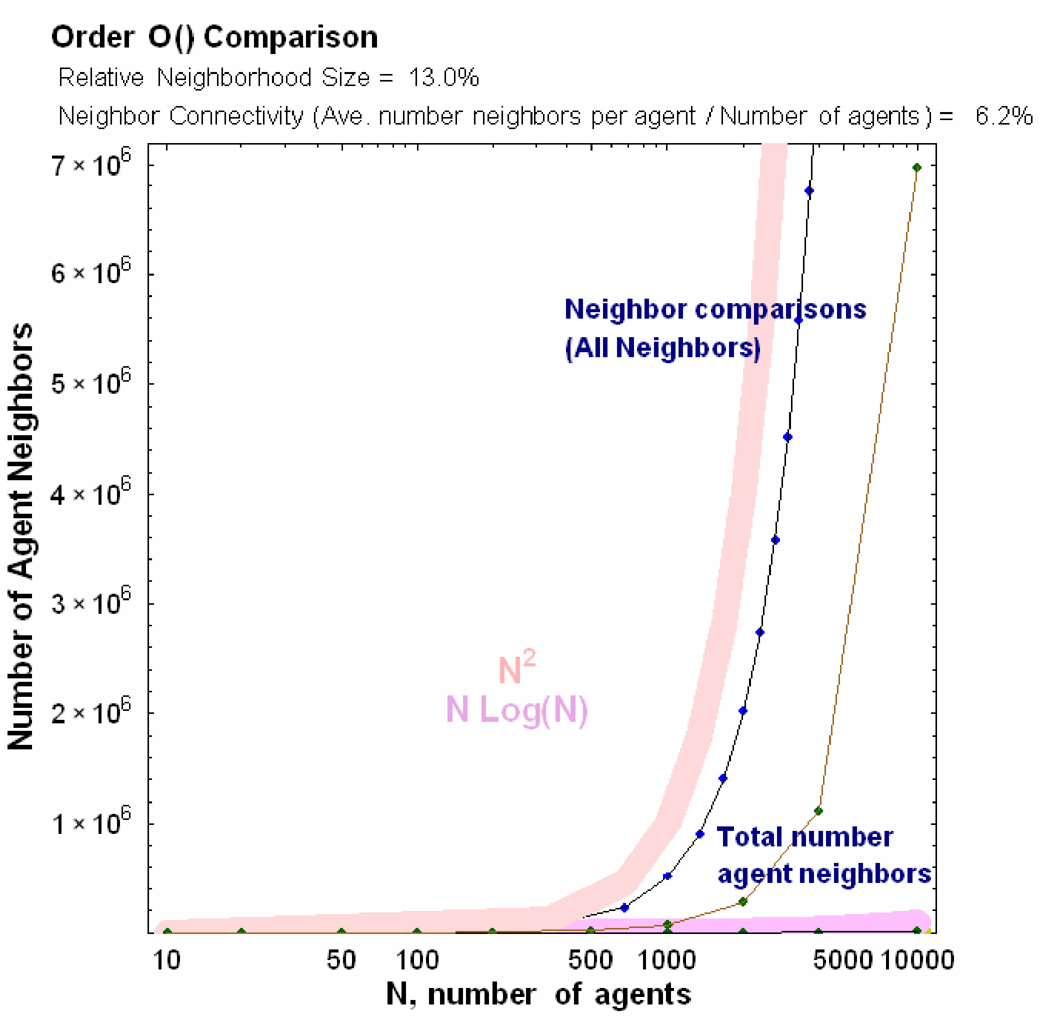


**FIGURE 3** Order of neighbor comparisons

Figure 4 illustrates the main results of the study and shows that over the full range of the number of agents simulated (10 to 10,000), the Grid Cell (GC) algorithm results in the least number of agent comparisons relative to the actual numbers of agent neighbors. Although the Quad Tree (QT) algorithm performance was close to the GC algorithm, especially as the number of agents increased, the GC algorithm resulted in superior performance for all cases. The All Neighbors (AN) algorithm performed the least well, as expected, for it identifies all agents as candidate neighbors, and there are many more candidate neighbors that do not turn out to be actual neighbors, as one would expect. The theoretical minimum ratio of candidate to actual neighbors is 1, meaning that no more candidates are identified than are actual neighbors, and no computational effort is wasted. Both the GC and QT algorithms appear to approach this theoretical limit at the upper range for the number of agents investigated.

## CONCLUSIONS

The problem of identifying agent neighbors is a key challenge in agent simulation. This study investigates two algorithms for identifying agent neighbors in $R^2$ using grid cells and quad trees, and compares the results to the all-neighbors algorithm. The methods trade off increasing implementation complexity against improved theoretical efficiency at finding agent neighbors. Our results indicate that, over the limited range of examples tested, the grid cell and quad tree algorithms perform well at identifying agent neighbors without identifying an excessive number of non-neighbor candidates. The grid cell algorithm performs somewhat better in all cases than

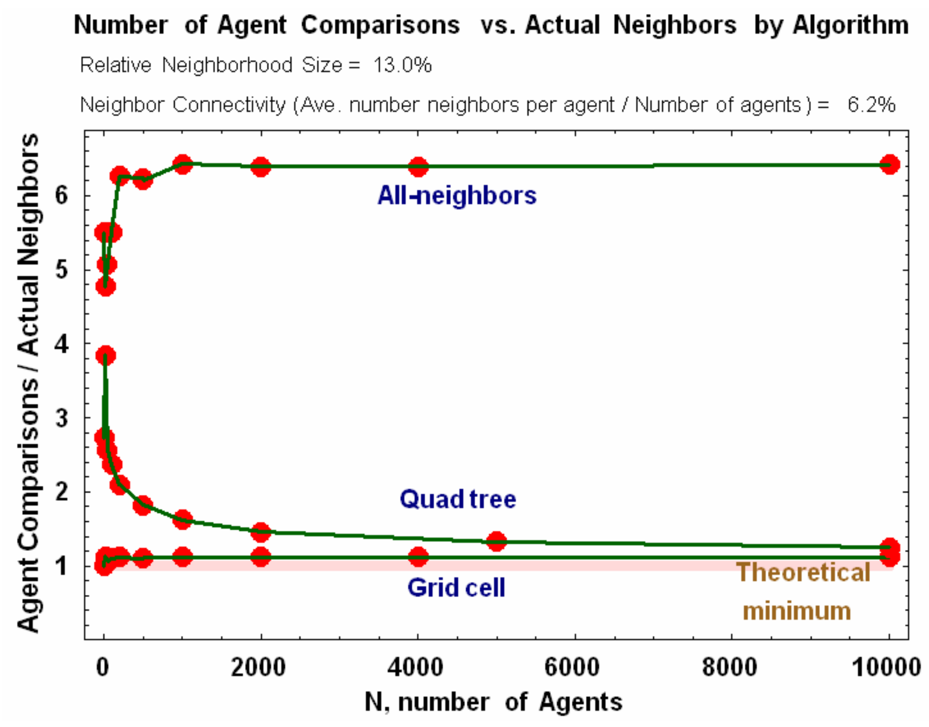


**FIGURE 4** Comparisons of candidate neighbors identified versus actual neighbors, by algorithm

the quad tree algorithm. Additional study into such matters as the computational times for creating the neighbor data structures and identifying agent neighbors should be considered in further research. Further refinements in the algorithms tested here, especially variants of the quad tree algorithm, may yield additional performance improvements. Finally, identifying agent neighbors is especially problematic for distributed agent simulation in which agents must communicate across networks of processors to identify neighboring agents and update themselves based on their neighbors. Further research is warranted in these areas.

## ACKNOWLEDGMENT

This work is sponsored by the U.S. Department of Energy, Office of Science, under contract W-31-109-ENG-38.

## REFERENCES

Aho, A.V., J.E. Hopcroft, and J.D. Ullman, 1976, *The Design and Analysis of Computer Algorithms*, Addison-Wesley: Reading, MA.

Amir, A., A. Efrat, J. Myllymaki, L. Palaniappan, and K. Wampler, 2004, “Buddy Tracking — Efficient Proximity Detection among Mobile Friends,” *Proceedings IEEE INFOCOM 2004*, pp. 298-309.

Basch, J., L. Guibas, and L. Zhang, 1997, “Proximity Problems on Moving Points,” *Proceedings 13th Annual ACM Symposium on Computational Geometry*, pp. 344-351; available at http://citeseer.ist.psu.edu/basch97proximity.html.

Cormen, T.B., C.E. Leiserson, R.L. Rivest, and C. Stein, 2001, *Introduction to Algorithms*, MIT Press: Cambridge, MA, 2nd edition.

Finkel, R.A., and J.L. Bentley, 1974, “Quad Trees for Retrieval on Composite Keys,” *Acta Informatica*, 4(1):1-9.

Gaylord, R., and L. D’Andria, 1998, *Simulating Society: A Mathematica Toolkit for Modeling Socioeconomic Behavior*, TELOS/Springer Verlag: New York, NY.

Hjaltason, G.R. and H. Samet, 1995, “Ranking in Spatial Databases,” in *Proc. 4th Symposium on Spatial Databases* (Egenhofer and Herring, eds.), *Lecture Notes in Computer Science 951*, Springer-Verlag, Berlin, 1995, pp. 83-95.: Portland, ME.

Knuth, D.E., 1998, *The Art of Computer Programming, Volume 3, Sorting and Searching*, Addision-Wesley: Reading, MA, 2nd edition.

Küpper, A., and G. Treu, 2006, “Efficient Proximity and Separation Detection among Mobile Targets for Supporting Location-Based Community Services,” *ACM SIGMOBILE Mobile Computing and Communications Review* 10(3):1-12.

Lewis, F.L., M. Fitzgerald, and K. Liu, 1997, "Robotics," *The Computer Science and Engineering Handbook*, Tucker, A.B., ed. CRC Press: Boca Raton, FL.

Macal, C., 2004, "Agent-Based Modeling and Social Simulation with Mathematica and MATLAB," in C. Macal, D. Sallach, and M. North (eds.) *Proceedings of the Agent 2004 Conference on Social Dynamics: Interaction, Reflexivity and Emergence*, ANL/DIS-05-6, co-sponsored by The University of Chicago and Argonne National Laboratory, Oct. 7-9, pp. 185-204; available at www.agent2006.anl.gov.

Macal, C., and T. Howe, 2005, "Linking Mathematica and the Repast Agent-Based Modeling Toolkit," *Wolfram Technology Conference 2005*, Champaign, IL, Oct. 6-8; available at http://library.wolfram.com/infocenter/Conferences/5767/.

Macal, C., and M. North, 2005, "Tutorial on Agent-based Modeling and Simulation," in M. Kuhl, N. Steiger, F. Armstrong, and J. Joines (eds.), *Proceedings 2005 Winter Simulation Conference*, Orlando, FL, Dec. 4-7.

Maeder, R., 2000, *Computer Science with Mathematica*, Cambridge University Press: Cambridge.

Reynolds, C., 1987, "Flocks, Herds, and Schools, a Distributed Behavioral Model," *Computer Graphics*, 21(4):25-34.

Reynolds, C., 2006, "Boids"; available at http://www.red3d.com/cwr/boidss/.

Samet, H., 1990, *The Design and Analysis of Spatial Data Structures*, Addison-Wesley: Reading, MA.

Wieland, F., D. Carnes, and G. Schultz, 2001, "Using Quad Trees for Parallelizing Conflict Detection in a Sequential Simulation," *Proceedings 15th Workshop on Parallel and Distributed Simulation*, Lake Arrowhead, CA, USA, May 15-18, pp. 117-123.

# Advanced Modeling Methods

# MACHINE LEARNING MEETS AGENT-BASED MODELING: WHEN NOT TO GO TO A BAR

W. RAND,* Northwestern University, Evanston, IL

## ABSTRACT

One of the promises of ABM is the ability to have adaptive agents make decisions in changing environments. Though great work has been done using adaptive agents in ABM, more research into the theoretical understanding of these systems would be useful. Adaptive agents have already been studied within machine learning (ML) — an area of artificial intelligence specifically concerned with adaptation and building internal models. The first part of this paper presents a framework for understanding ML as a component of ABM, and describes how different ML techniques can be incorporated into some ABMs. At the high level this framework consists of two cycles that involve evaluating input, making decisions and then generating output. Within this generalized framework, the ML algorithm is using the ABM as an environment and a reward generator, while the ABM is using the ML algorithm to refine the internal models of the agents. There are many details that must be answered before any ML technique can be incorporated into an ABM. In this paper I start to explore some guidelines for how to more closely integrate ABM and ML and will discuss complications that arise when combining ABM and ML techniques. To illustrate some of these issues, I will describe an integration of a ML technique within the El Farol Bar Problem. I will conclude with some discussion of this integration and a look toward future research.

**Keywords:** Machine learning, agent-based modeling, framework El Farol Bar Problem, genetic algorithms

## INTRODUCTION

As we pause to reflect on how agent-based modeling (ABM) has changed in the ten years since SugarScape (Epstein and Axtell 1996), one aspect of ABM that could use more analysis is adaptation. Though there are notable exceptions like the El Farol Bar Problem (Arthur 1994) among others, few models make use of an adaptive mechanism within the ABM framework. By an adaptive mechanism, I refer not to the ability of agents to take different actions, but rather the ability for agents to come up with a new strategy of how to take action. This is particularly surprising since the ability to allow agents to adapt to their surrounding is often listed as a reason to use ABM instead of other modeling techniques. When Holland discussed complex adaptive systems (CAS) and their relationship to ABM in *Hidden Order* (Holland 1995), he devoted an entire chapter to adaptive agents, and specifically mentioned internal models as one of the mechanisms that define a CAS. Despite this more effort needs to be placed into understanding adaptive agents. However, as we examine the last ten years of ABM it is important to not only notice its deficiencies but also to see how these areas can be improved. In a fortuitous

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

coincidence, at the same time ABM research has been gathering momentum, so has machine learning — an area of artificial intelligence specifically concerned with adaptation and building internal models. If the ABM community can make use of the knowledge and research developed by the machine learning community, it would greatly facilitate the study of adaptation within ABM.

ABM and ML can be combined in a variety of ways and has been examined in the past (Wolpert, Wheeler et al. 1999). However, in this paper I choose to examine the use of ML to refine the internal models of agents in an ABM. The first part of this paper presents a framework for understanding machine learning as a component of ABM, and describes how different machine learning techniques like genetic algorithms (GAs), neural nets (NNs), and Bayesian Classifiers can easily be incorporated into many agent-based models. At the high level this framework consists of two interlocked cycles that examine input, make decisions and generate output. In this generalized framework, the machine learning algorithm is using the ABM as an environment and a reward generator, while the ABM is using the machine learning algorithm to maintain the internal models of the agents.

There are many details that must be answered before any machine learning technique can be incorporated into even the simplest agent-based model. In this paper, some of these details of how to build this general framework are discussed. Of course even after a general framework has been decided upon there are many more questions that still need to be answered, like what particular technique to use, and how to set the parameters of that technique. This paper will discuss these complications.

The final section of this paper will illustrate some of these issues with an example. This practical example will consist of a genetic algorithm implemented within the context of the El Farol Bar Problem. The design of such an implementation and the consideration of the various issues involved will be discussed.

## THE FRAMEWORK

At a high level ABM and ML both 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. To give this high level description more context I will first discuss the *ABM cycle*, then the *ML cycle* and finally an *integrated cycle*.

The first cycle is the standard agent-based model cycle and can be broken down into three steps: (1) initialize the world and a population of agents, (2) each agent observes its world, and (3) each agent takes an action based on the current observations, and the model repeats by going back to (2). This cycle becomes an *adaptive* agent-based model if we incorporate a fourth step between (2) and (3) where each agent updates their internal model of the world, and decides what action today based on that internal model. The adaptive ABM cycle is illustrated in Figure 1.

The second distinct cycle, as seen in Figure 2, is the machine learning cycle and can be broken down into four steps as well: (1) create an initial internal model, (2) observe the world

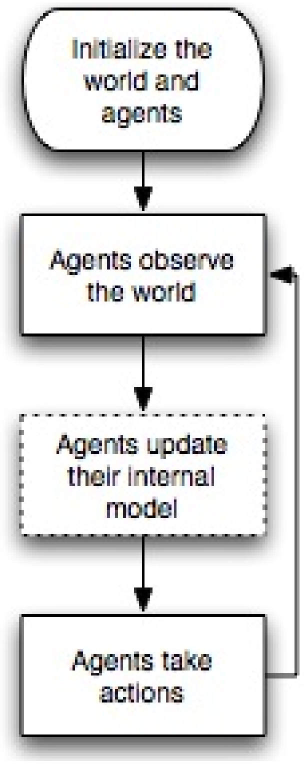

Initialize the world and agents
Agents observe the world
Agents update their internal model
Agents take actions

**Figure 1** ABM cycle

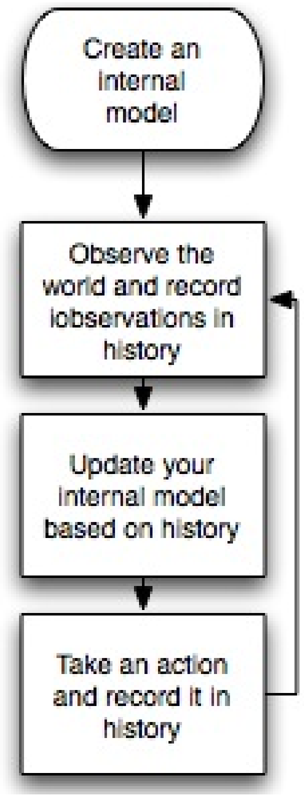

Create an internal model
Observe the world and record iobservations in history
Update your internal model based on history
Take an action and record it in history

**FIGURE 2** ML Cycle

and take note of rewards received, (3) update the internal model, (4) take an action based on the internal model and the current observations, go back to (2) and repeat.

As is obvious the two cycles are quite similar to each other and thus integrating the frameworks is not very difficult at all. However how this integration is practically accomplished could be done in many different ways. In this paper I have chosen to explore the use of the ML cycle as a model refinement engine for the ABM. Thus the integrated cycle focuses on the ABM and interrupts its standard flow in step three, by sending data to the ML cycle to handle the model refinement. This is illustrated in Figure 3.

## PRACTICAL DECISIONS

Of course utilizing machine learning techniques within agent-based modeling is not as simple as describing the framework. There are many practical details that must be addressed when deciding how to integrate ABM and ML. One question that must be answered is whether the machine learning technique should be a supervised learning technique (an external teacher determines whether any action taken was correct or incorrect) or an unsupervised learning technique (agents take actions and occasionally gain rewards but there is not necessarily a chain of causation from any action to any reward). Supervised learning requires explicit knowledge of

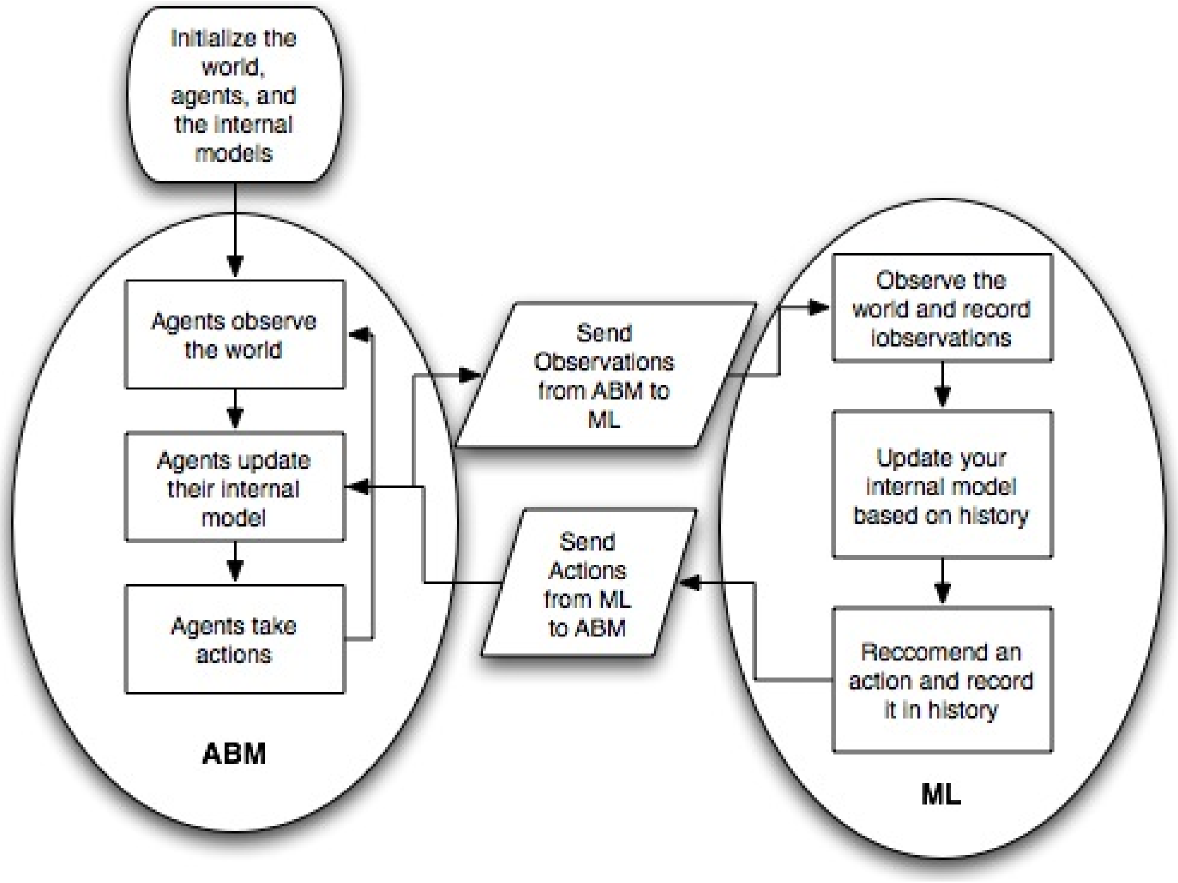

Initialize the world, agents, and the internal models
Agents observe the world
Agents update their internal model
Agents take actions
ABM
Send Observations from ABM to ML
Send Actions from ML to ABM
Observe the world and record lobservations
Update your internal model based on history
Reccomend an action and record it in history
ML

**FIGURE 3** Integrated cycle

what actions provoked what rewards, i.e., a mapping of inputs and outputs. Unsupervised learning does not require this but instead simply builds a model of how the world behaves.

Another question that must be addressed is whether or not the agent's own action needs to be taken into account while building the internal model of the world. In many cases within ABM, agents assume that they can make decisions about the world as if they were not a part of the world. This is sometimes called the *Wonderful Life Assumption*[1] from the Frank Capra movie *It's A Wonderful Life* where James Stewart's character George Bailey wonders if anything he has done has made the world a better place. However in many cases like the El Farol Bar Problem, agents' actions do influence the world. In fact in the case of the El Farol Bar Problem, since the only variable of interest is the attendance at the bar, agents' actions basically define the world. On the other hand since agent's actions may have a minimal effect on the world it may be possible to make the Wonderful Life Assumption and safely ignore the agent's own action when building up an internal model. This may actually be the case in the El Farol Bar Problem since each agent only contributes 1/100 to the total attendance of the bar.

Even after the family of techniques has been decided upon, there are still many specific algorithms that are more or less useful and must be carefully considered. Neural networks for instance, are good at classifying large amounts of data fairly quickly, but in the end they do not yield white box results; that is after they have run for awhile it is very difficult to determine how they are making their decisions. Decision trees, on the other had, do create very white box results, but are not very good at classifying continuous data. There are a variety of books that discuss ML algorithms, their implementations and their pros and cons (Mitchell 1997; Hastie, Tibshirani et al. 2001).

Even after the particular technique, there are still a large variety of parameters that need to be set, and tuned in order to work properly within the ABM environment. Much of this is a matter of art to get the results one desires, but some sets of ML algorithms have more literature than others regarding advice on how to tune the parameters.

All in all, there are many matters to consider when combining ABM with ML, but the advantages that one gains from having truly adaptive agents, which can modify not only the actions they are taking but also the strategies that they use to determine those actions is often worthwhile.

## A CASE STUDY: THE EL FAROL BAR PROBLEM

It is difficult to discuss many of these issues without a particular example to focus the discussion around. After all in the end a programmer or model builder must actually write some code to integrate ABM and ML. Thus, it is important to think of how these issues affect actual model development. In order to illustrate a few of these issues, I will consider one integration in specific details. I have chose to use the El Farol Bar Problem (Arthur 1994) as an example. This was an early ABM that included adaptive agents. In short, the model consists of a 100 agents trying to decide whether or not to attend a bar on a certain night. If they attend and the bar is

---

1 This is related to Wolpert et al's *wonderful life subworld utility* but here I am concerned with a particular subworld—the one where the agent does not account for their own action—and hence the assumption I am discussing is less general than Wolpert's utility function (Wolpert, Wheeler et al. 1999).

crowded they receive no reward; if they stay home they receive no reward. However, if they attend and the bar is not crowded (less than 60 attendees in Arthur's model) they receive a reward. The only information they have to make this decision is the attendance of the bar the last week which is printed in the paper and can be remembered over time. So the question was: how do agents decide which strategy to use to determine whether to attend the bar or not?

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 and can be described within the framework of Figure 2. Initialize your group of strategies by generating some random strategies, like take last week's attendance double it, or subtract the third to last week's attendance from last weeks, or take a running average of the last three weeks attendances (Step 1). Then at each time step observe how the strategies have done on the current set of training data, i.e. the previous bar attendances (Step 2). After that, refine your internal model by selecting the best one given the new data (Step 3). Finally act on the strategy that reflects your refined model (Step 4) and repeat (at Step 2). This ML technique could be used for other problems than the El Farol Bar Problem; for instance, Arthur's technique could be used to predict stock market price or estimate the rainfall in a certain geographic location. Thus Arthur's technique is not particular to the El Farol Bar Problem and could be replaced by any number of standard ML techniques.

I wanted to make use of a different ML technique than the one Arthur described. So first I had to decided whether to use a supervised or unsupervised learning technique. It might appear at first that it is necessary to use an unsupervised technique since an agent's action is not directly responsible for their reward. However since in Arthur's version of the problem the agents are not necessarily trying to maximize their utility but rather just trying to minimize their error of prediction and then take an action based upon that, we can assume that previous time data series is in fact a supervised training set. Given this assumption we can safely choose a supervised machine learning technique. Since supervised ML techniques tend to be faster than unsupervised techniques, in general when there is enough information available to classify the problem as a supervised problem it is helpful to utilize a supervised ML technique. However, it would also be possible to use an unsupervised learning method if someone simply wanted to build up a model of how attendances influenced each agent's rewards.

Of course it is also necessary to consider whether or not to model the agent's own action. However the way this problem has been framed for the agents, they are automatically not making the Wonderful Life Assumption. This is because agents are predicting the attendance at the bar and making a decision about whether or not they will attend the bar based on that prediction. Thus, it can be assumed that their prediction automatically takes into account their own decision. The agent is asking the ML technique to predict next week's attendance and is not putting any restrictions on their request; therefore the prediction should take into account whatever action the agent will take.

Second, it was necessary to choose a particular machine learning technique. There is no obvious decision here, but partially since it was originally suggested in Arthur's paper, I decided to investigate the use of the genetic algorithm (GA) as originally devised by Holland (Holland 1975). Fogel had previously explored such a technique within the El Farol Bar Problem (Fogel, Chellapilla et al. 1999). 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, decides which ones to keep using, changes them slightly and re-evaluates them. In addition the GA is often described as manipulating *schemata* and thus may be similar to the human process of induction (Holland, Holyoak et al. 1986) which is what Arthur's original model was intended to emulate. Clearly, then by examining the benefits of various ML algorithms and choosing the one that seemed to satisfy the task at hand I was able to choose a particular algorithm.

In order to integrate a GA within the El Farol Bar Problem I had to first place the original El Farol Bar Problem within the context of the Integrated cycle described above. Thus I filled out the left hand bubble with the details of El Farol Bar Problem. Then I filled out the right hand bubble with the details of the GA. The result is illustrated in Figure 4.

After I had visualized the integration I had to actually accomplish the task, which involved not only setting the parameters of the El Farol Bar Problem but also that of the GA of each agent. My over-riding goal in this task was to see if I could generate results similar to Arthur's original results. Thus I used a set of parameters similar to what Arthur had described for the El Farol Bar Problem. For the GA, I could have chose to use Fogel's parameters, but I decided that those were farther away from Arthur's original model than I wanted to deviate since the Fogel's parameters seemed to require a larger amount of computational resources than

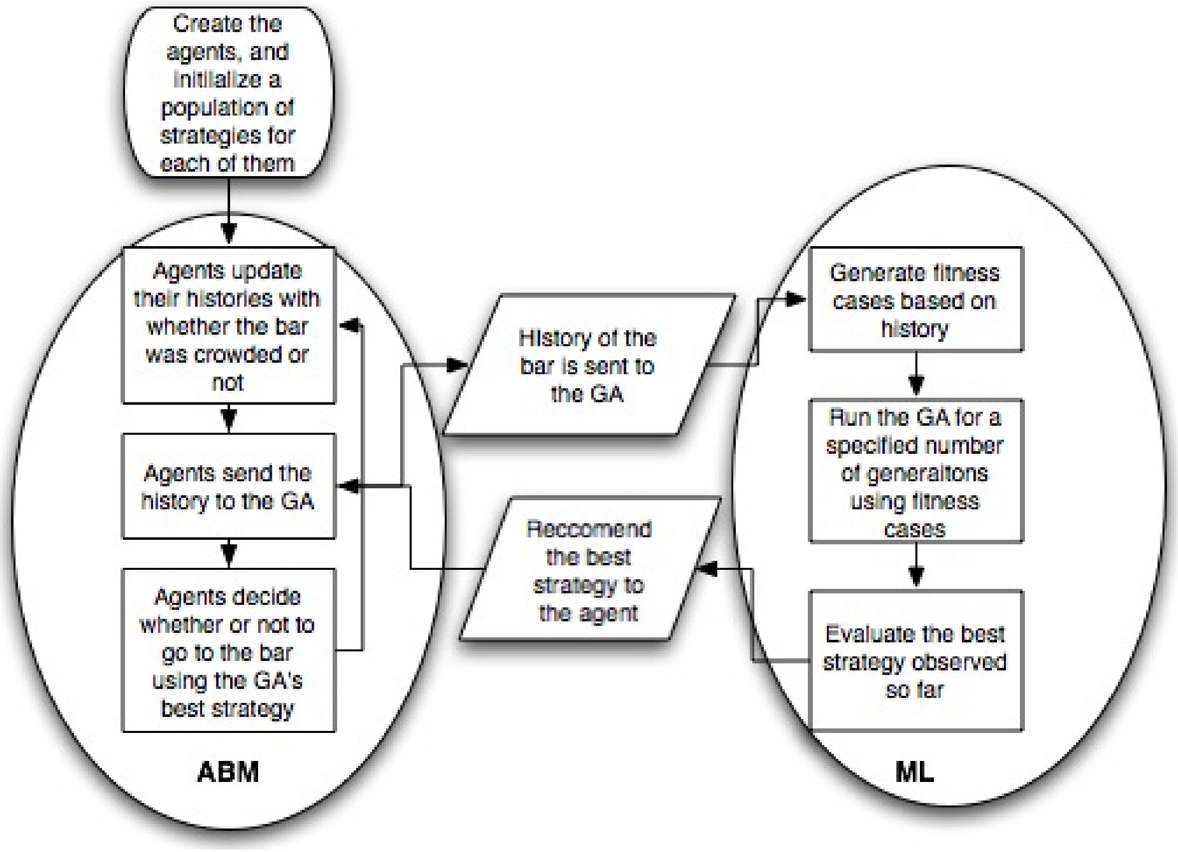

Create the agents, and initialize a population of strategies for each of them
Agents update their histories with whether the bar was crowded or not
Agents send the history to the GA
Agents decide whether or not to go to the bar using the GA's best strategy
ABM
History of the bar is sent to the GA
Reccomend the best strategy to the agent
Generate fitness cases based on history
Run the GA for a specified number of generaitons using fitness cases
Evaluate the best strategy observed so far
ML

**FIGURE 4 Example using El Farol**

Arthur’s original technique. Thus instead I decided to choose parameters for the GA that would more closely imitate Arthur’s original model. In general, it is good to consider two factors when setting ML parameters: (1) review relevant literature on good parameter settings, and (2) keep your modeling goal in mind when setting parameters.

## CONCLUSION

Over the last ten years, there have been a lot of exciting and interesting developments in ABM, and the use of truly adaptive agents within ABM is one stimulating promise that is still being explored. The integration of ML techniques within ABM will hopefully allow for the development of novel and original models. This paper has discussed a general framework for these combined algorithms, and has begun to discuss some of the issues that must be addressed. Finally one integration has been described and used as a case study to further discuss the issues.

In the future, this line of research will continue. The integration of the El Farol Bar Problem with a variety of ML techniques and a thorough analysis of the results is warranted. As is consideration of other ABMs and how to integrate them with ML algorithms. As more and more of these integrations are attempted it is hoped that a suite of best practices will develop that can serve as advice and eventually form the basis for a more concrete set of guidelines for future model developers. The scope and use of ABM will be greatly expanded by the increased use of strategically adaptive agents.

## REFERENCES

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

Epstein, J. and R. Axtell (1996). *Growing Artificial Societies: Social Science from the Bottom Up*. Cambridge, MA, MIT Press.

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.

Wolpert, D. H., K. R. Wheeler, et al. (1999). General Principles of Learning-Based Multi-Agent Systems. *Autonomous Agents*. Seattle, WA, ACM.

# REAL-TIME EVOLUTIONARY AGENT CHARACTERIZATION AND PREDICTION

H.V. PARUNAK,[*] NewVectors LLC, Ann Arbor, MI
S. BRUECKNER, NewVectors LLC, Ann Arbor, MI
R. MATTHEWS, NewVectors LLC, Ann Arbor, MI
J. SAUTER, NewVectors LLC, Ann Arbor, MI
S. BROPHY, NewVectors LLC, Ann Arbor, MI

## ABSTRACT

Available information about real-world agents is often limited to external observations. BEE (Behavior Evolution and Extrapolation) uses a faster-than-real-time agent-based model of the environment to characterize agents' internal state by evolution against observed behavior, and then predict their behavior, taking into account the dynamics of their interaction with the environment.

**Keywords:** Plan recognition, plan inference, evolution, prediction, emotion, BDI, swarm intelligence, digital pheromones, dynamics.

## INTRODUCTION

Many application domains require reasoning from external agent behavior to an estimate of internal state, often motivated by a desire to predict the agent's behavior (the "plan recognition" or "plan inference" problem). Work to date focuses almost entirely on recognizing the rational (not emotional) state of a single agent (not an interacting community), and frequently requires explicit communications between agents. Many problems deviate from these conditions.

- Adding agents leads to a combinatorial explosion that can swamp conventional analysis.
- The dynamics of the environment can frustrate an agent's intentions.
- Agents often hide their intentions (and even their presence), rather than intentionally sharing information.
- An agent's emotional state may be at least as important as its rational state in determining its behavior.

Adversarial domains, including military combat, competitive business tactics, and multi-player computer games, exhibit these constraints.

BEE (Behavioral Evolution and Extrapolation) is a novel approach to recognizing the rational and emotional state of multiple interacting agents based solely on their behavior, without recourse to intentional communications. It is inspired by techniques used to predict nonlinear

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

dynamical systems by continually fitting a representation of the system to its recent past behavior. For nonlinear dynamical systems, the representation is a closed-form mathematical equation. In BEE, it is a set of parameters governing the behavior of software agents representing the individuals being analyzed. BEE currently characterizes and predicts the behavior of agents representing soldiers in urban combat (Kott 2004).

We survey relevant previous work (including both plan recognition and prediction in nonlinear dynamical systems), describe BEE's architecture, and report results from experiments with the system. Further details that cannot be included here for the sake of space are available in an on-line technical report (Parunak, Brueckner et al. 2005).

## PREVIOUS WORK

### Plan Recognition in AI

One often describes an agent's cognitive state in terms of its beliefs, desires, and intentions (Rao and Georgeff 1991). An agent's beliefs are propositions about the state of the world that it considers true, based on its perceptions. Its (possibly mutually inconsistent) desires are propositions that it would like to be true. Its intentions, or goals, are a subset of its desires, believed to be consistent, that it selects, based on its beliefs, to guide its actions.

Goals guide actions. An agent's past actions should reflect its goals, and knowledge of its goals enables predictions of future actions.

Reasoning from an agent's actions to its goals is called "plan recognition" or "plan inference." This body of work (Carberry 2001) is rich and varied. It covers both single-agent and multi-agent plans, intentional vs. non-intentional actions, speech vs. non-speech behavior, adversarial vs. cooperative intent, complete vs. incomplete world knowledge, and correct vs. faulty plans, among other dimensions. Plan recognition usually supports a higher-level function, such as predicting the agent's future actions.

An agent's plan is useful input to predicting its future behavior, but hardly sufficient. There are at least two other influences, one internal and one external.

The external influence is the dynamics of the environment, which may include other agents. Rational analysis of an agent's goals may help us predict what it will attempt, but any nontrivial plan with several steps will depend on the environment's reactions, for two reasons.

- The environment may do things on its own that interfere with the desires of the agent (Michel 2004).
- Most interactions among agents, and between agents and the world, are nonlinear. When iterated, these can cause trajectories to diverge rapidly.

Actual simulation of futures is the only one we know to address this problem.

Human agents are subject to an internal influence, emotional state, which can modulate their decision process and focus of attention (and thus their perception). Sometimes emotion can lead an agent to act in a way that logically appears irrational.

Most work on plan recognition for prediction focuses on the rational plan, and does not take into account either external environmental influences or internal emotional biases. BEE integrates all three elements.

## Real-Time Fitting in Nonlinear Systems Analysis

Many systems can be described by a time-varying state vector, often analyzed as vector differential equations, $\frac{d\vec{x}}{dt} = f(\vec{x})$.

When *f* is nonlinear, the system can be formally chaotic, making long-range prediction impossible. However, one can anticipate the system's near-term behavior, by fitting a convenient functional form for *f* to the system's trajectory in the recent past, and then extrapolating this fit (Figure 1, (Kantz and Schreiber 1997)). Iterating this process provides a limited look-ahead.

This approach requires systems that can efficiently be described by mathematical equations that can be fit using methods such as least squares. BEE extends this approach to agent behaviors, which it fits to observed behavior using evolution.

## ARCHITECTURE

BEE predicts the future by observing the behavior of agents in a fine-grained simulation. Key elements of the architecture include the pheromone infrastructure through which agents interact, the model of an individual agent, the information sources that guide agents, and the evolutionary cycle that they execute.

### Pheromone Infrastructure

BEE must keep pace with the battle. Thus we use simple agents coordinated using pheromone mechanisms (Brueckner 2000). This infrastructure runs on the nodes of a rectangular lattice. Each node maintains a scalar value for each flavor of pheromone, and provides three functions: aggregation of deposits from individual agents to fuse information across multiple agents and through time, evaporation over time to provide truth maintenance by discarding obsolete information, and diffusion to nearby places to share information with other agents.

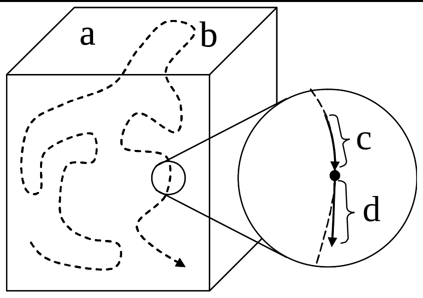


**FIGURE 1** Tracking a nonlinear dynamical system. a) State space; b) trajectory; c) recent past observations; d) extrapolated fit.

Each page of the timeline is a complete pheromone field for the world at the BEE time τ represented by that page. Movement based on pheromone gradients is a simple process, so this system can support realistic agent populations without excessive computer load. In our current system, about 24,000 agents execute concurrently on a stock desktop computer. Two features of the environment permit it to be scaled using multiple processors. First, agents interact only with their local neighborhood. Second pheromones below a certain threshold are deleted, so they propagate only to a certain radius.

## Agent Model

BEE's agents are inspired by our previous work on fine-grained agents that coordinate their actions through digital pheromones in a shared environment (Parunak, Brueckner et al. 2004), and the success of previous agent-based combat modeling.

Agents follow local gradients of functions over digital pheromones. Their movements change the deposit patterns. This feedback loop, together with evaporation and propagation in the environment, supports complex patterns of interaction and coordination among the agents (Parunak, Brueckner et al. 2003). Table 1 in (Parunak, Brueckner et al. 2005) shows the BEE's current pheromone flavors. For example, a living member of the adversary emits a RED-ALIVE pheromone, while roads emit a MOBILITY pheromone.

Our soldier agents are inspired by EINSTein and MAUI. EINSTein (Ilachinski 2004) represents an agent as a set of six weights, each in [-1, 1], describing the agent's response to six kinds of information (e.g., number of alive friendly; distance to objective). A positive weight indicates that the agent is attracted to the entity described by the weight, while a negative weight indicates that it is repelled. MANA (Lauren and Stephen 2002) extends EINSTein. Among other changes, it defines a set of triggers (e.g., reaching a waypoint, being shot at) that shift the agent from one personality vector to another.

The personality vectors in MANA and EINSTein reflect both rational and emotive aspects of decision-making. Attraction or repulsion to friendly or adversarial forces in various states of health is a component of emotion (e.g., fear, compassion, aggression). The notion of waypoints reflects goal-oriented rationality.

BEE embodies an integrated rational-emotive personality model.

A BEE agent's rationality is a vector of seven desires, which are values in [-1, +1], including ProtectRed (the adversary), ProtectBlue (friendly forces), and AvoidDetection. For example, a negative value of ProtectRed indicates a desire to harm Red. Table 2 in (Parunak, Brueckner et al. 2005) shows which pheromones attract or repel an agent with a given desire, and how that tendency translates into action. For example, an agent with a high positive desire to ProtectRed will be attracted to RED-ALIVE, RED-CASUALTY, and MOBILITY pheromone, and will move at maximum speed.

The BEE's personality model (Parunak, Bisson et al. 2006) is based on the Ortony-Clore-Collins (OCC) framework (Ortony, Clore et al. 1988). OCC define emotions as "valanced reactions to agents, states, or events in the environment." MANA's trigger states capture this notion of reaction. BEE goes further by recognizing that agents may differ in their sensitivity to triggers. For example, threatening situations stimulate fear, but a given level of threat produces more fear in a new recruit than in a seasoned veteran. Thus we model not only Emotions, but Dispositions (one per Emotion). Dispositions are relatively stable, and stay constant over the time horizon of a run of the BEE, while Emotions vary based on the agent's Disposition and the stimuli to which it is exposed.

Interviews with military domain experts identified the two most crucial emotions for combat behavior as Anger (with the corresponding disposition Irritability) and Fear (whose disposition is Cowardice). Table 3 in (Parunak, Brueckner et al. 2005) shows which pheromones trigger which emotions. For example, RED-CASUALTY pheromone stimulates both Anger and Fear in a Red agent, but not in a Blue agent.

A non-zero emotion modifies the agent's actions. Elevated Anger increases the likelihood of movement, weapon firing, and an exposed posture. Elevated Fear decreases these likelihoods.

## Information Sources

The flexibility of the BEE's pheromone infrastructure permits the integration of numerous information sources as input to our characterizations of entity personalities and predictions of their future behavior. Our current system draws on three sources of information, but others can readily be added.

**Real-world observations**.—Observations from the real world are encoded into the pheromone field each increment of BEE time, as a new "current page" is generated.

**Statistical estimates of threat regions**.—Statistical techniques[1] estimate the level of threat to each force (Red or Blue), based on the topology of the battlefield and the known disposition of forces. For example, a broad open area with no cover is threatening, especially if the opposite force occupies its margins. The results of this process are posted to the pheromone pages as "RedThreat" pheromone (representing a threat to red) and "BlueThreat" pheromone (representing a threat to Blue).

**AI-based plan recognition**.—While plan recognition is not sufficient for effective prediction, it is a valuable input. We dynamically configure a Bayes net based on heuristics to identify the likely goals that each entity may hold.[2] The destinations of these goals function as "virtual pheromones." Ghosts include their distance to such points in their action decisions, achieving the result of gradient following without the computational expense of maintaining a distinct pheromone flavor.

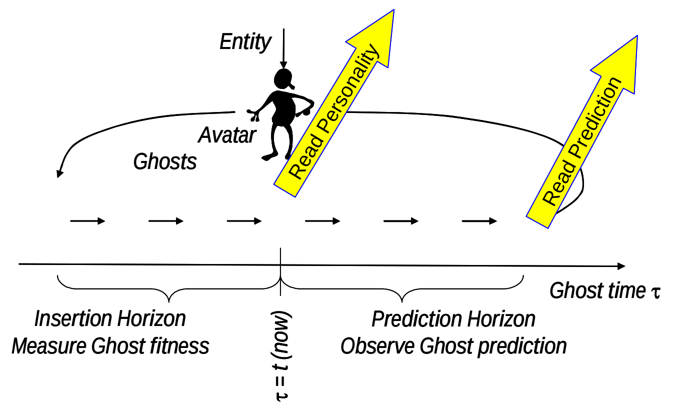


**FIGURE 2** Behavioral emulation and extrapolation

## The BEE Cycle

The major innovation in BEE

[1] This process is developed by Rafael Alonso, Hua Li, and John Asmuth at Sarnoff Corporation.

[2] This process is developed by Paul Nielsen, Jacob Crossman, and Rich Frederiksen at Soar Technology.

extends the nonlinear systems technique of continuous fitted prediction to agent behaviors.

Figure 2 summarizes the BEE process. Each active entity has an persistent avatar that continuously generates a stream of ghost agents representing itself. The modeling entity combining an avatar and its ghosts is a *polyagent* (Parunak and Brueckner 2006). The avatar inserts its ghosts into the recent past in a faster-than-real-time simulation. It evolves their personalities against the behavior it observes in the real world, then lets the fittest ghosts run into the future to explore alternative future behaviors, and fuses these into a single prediction.

## EXPERIMENTAL RESULTS

We test BEE in experiments in which human wargamers make decisions that are played out in a battlefield simulator. The commander for each side (Red and Blue) has at his disposal a team of human operators who set waypoints for individual units in the simulator. Each puckster is responsible for four to six units. The simulator moves the units, determines firing actions, and resolves the outcome of conflicts.

### Fitting Dispositions

To test our ability to recognize emotions from behavior, one Red operator secretly selects two of his units to be cowardly (“chickens”) and two to be irritable (“Rambos”). He moves each unit according to the commander’s orders until the unit encounters the emotion’s triggers. Then he manipulates chickens as though they are fearful (avoiding combat, avoiding Blue), and moves Rambos into combat as quickly as possible.

The difference between the two disposition values (Irritability - Cowardice) of the fittest ghosts proves an excellent indicator of their entity’s emotional state. We maintain a 800-second exponentially weighted moving average of this Delta Disposition, and declare the unit to be a chicken or Rambo if this value passes a negative or positive threshold, respectively (currently, ± 0.25).

Figure 3 compares our emotion detector with the judgments of human officers. We played two chickens in each of 11 experiments. The plot shows how many had been detected at a given point in the runs. For example, the square at (60%, 10) means that the total number of chickens detected in all runs at the 60% point is 10. BEE was able to detect chickens earlier than humans, while missing only one chicken that the humans detected.

In addition to units intentionally played as emotional, BEE sometimes detects

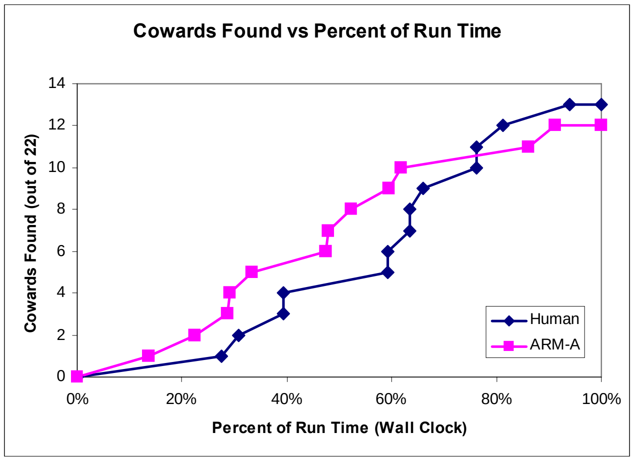


**FIGURE 3** BEE vs. human

other units as cowardly or brave. Analysis shows that these characterizations are appropriate: units that flee in the face of enemy forces or weapons fire are detected as chickens, while those that stand their ground or rush the adversary are denominated as Rambos.

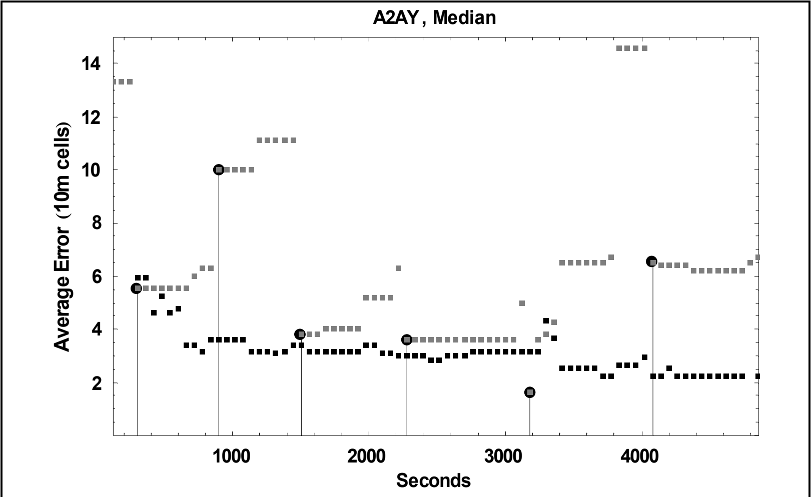


**Figure 4** Predictive accuracy

## Integrated Predictions

Each future ghost generates a possible future path that its unit might follow. The set of such paths for all ghosts embodies a number of distinct predictions, including the most or least likely future, the future that poses the greatest or least risk to the opposite side, the future that poses the greatest or least risk to one's own side, and so forth. Here, we select the future whose ghost receives the most guidance from pheromones in the environment at each step along the way. In this sense, it is the most likely future.

We evaluate predictions spatially, comparing an entity's location with the prediction made 15 minutes earlier. Figure 4 compares BEE's prediction errors (median over 20 Red units) with those from a game-theoretic predictor based on linguistic geometry (LG) (Stilman 2000) on a typical run. The LG predictor can produce only one prediction (a 15 minute trajectory for each unit) about every 15 minutes (the stemmed points in the Figure), so the plotted errors for times in between these computations are based on a world state that becomes increasingly obsolete as time passes until a new prediction is issued. The BEE prediction is updated at each time step, by reading off from the any-time extrapolation process. BEE outperforms the LG predictor not only between successive LG predictions, but also when the LG prediction is current (stemmed LG points).

## CONCLUSIONS

BEE reasons from observed external behavior to internal state using a faster-than-real-time simulation of swarming agents, coordinated through digital pheromones. This simulation integrates threat regions, a cognitive projection of the agent's beliefs, desires, and intentions, a model of the agent's emotional disposition and state, and the dynamics of environmental interactions. By evolving agents in this rich environment, we can fit their internal state to their observed behavior. In realistic wargames, the system successfully detects deliberately played emotions and makes reasonable predictions about the entities' future behaviors.

BEE only models internal state variables that impact the agent's external behavior. It cannot fit variables that the agent does not manifest externally, since its evolutionary cycle compares the agent's outward behavior with that of the real entity. This limitation is serious if our purpose is to understand the entity's internal state for its own sake. If we are fitting agents to predict their behavior, the limitation is much less serious. State variables that do not impact behavior, invisible to a behavior-based analysis, are irrelevant to a behavioral prediction.

BEE lends itself to extension in several directions.

- A dynamical simulation can integrate many other inputs besides the current ones, requiring less engineering than more traditional ways of reasoning about multiple knowledge sources.
- Our repertoire of emotions is a small subset of those that might be useful for understanding and projecting behavior.
- The mapping between an agent's psychological state and its behavior is not one-to-one. Multiple internal states might be consistent with a given observed behavior under one set of conditions, but yield distinct behaviors under others. If the recent environment confounds such distinct states, we will be unable to distinguish them. As long as the environment stays in this state, our predictions will be accurate, whichever internal state we assign. If the environment then shifts to one under which the different states lead to different behaviors, using the previously chosen internal state will yield inaccurate predictions. One approach is to probe the real world, perturbing it to stimulate distinct behaviors from entities whose psychological state is otherwise indistinguishable. Such probing is an important intelligence technique. BEE's faster-than-real-time simulation may enable us to identify appropriate probing actions.
- BEE has been developed for adversarial reasoning in urban warfare. Other potential applications include computer games, business strategy, and sensor fusion.

## ACKNOWLEDGMENTS

This material is based in part upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. NBCHC040153. 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 DARPA or the Department of Interior-National Business Center (DOI-NBC). Distribution Statement "A" (Approved for Public Release, Distribution Unlimited).

## REFERENCES

Brueckner, S., 2000, Return from the Ant: Synthetic Ecosystems for Manufacturing Control. Thesis, Department of Computer Science, Humboldt University Berlin, Berlin, Germany, http://dochost.rz.hu-berlin.de/dissertationen/brueckner-sven-2000-06-21/PDF/Brueckner.pdf.

Carberry, S., 2001, "Techniques for Plan Recognition." *User Modeling and User-Adapted Interaction* **11**(1-2): 31-48. http://www.cis.udel.edu/~carberry/Papers/UMUAI-PlanRec.ps.

Ilachinski, A., 2004, *Artificial War: Multiagent-based Simulation of Combat*. Singapore, World Scientific.

Kantz, H. and T. Schreiber, 1997, *Nonlinear Time Series Analysis*. Cambridge, UK, Cambridge University Press.

Kott, A., 2004, "Real-Time Adversarial Intelligence & Decision Making (RAID)." Retrieved 20 January, 2005, from http://dtsn.darpa.mil/ixo/programdetail.asp?progid=57.

Lauren, M. K. and R. T. Stephen, 2002, "Map-Aware Non-uniform Automata (MANA)—A New Zealand Approach to Scenario Modelling." *Journal of Battlefield Technology* **5**(1 (March)): 27ff. http://www.argospress.com/jbt/Volume5/5-1-4.htm.

Michel, F., 2004, Formalisme, méthodologie et outils pour la modélisation et la simulation de systèmes multi-agents. Thesis, Department of Informatique, Université des Sciences et Techniques du Languedoc, Montpellier, France, http://www.lirmm.fr/~fmichel/these/index.html.

Ortony, A., G. L. Clore, et al., 1988, *The cognitive structure of emotions*. Cambridge, UK, Cambridge University Press.

Parunak, H. V. D., R. Bisson, et al., 2006, *Modeling Agent Emotions*. Social Agents: Results and Prospects (Agent 2006), Chicago, IL, Argonne National Laboratory.

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.

Parunak, H. V. D., S. Brueckner, et al., 2003, *A Design Taxonomy of Multi-Agent Interactions*. Agent-Oriented Software Engineering IV, Melbourne, AU, Springer. 123-137. www.newvectors.net/staff/parunakv/cox.pdf.

Parunak, H. V. D., S. Brueckner, et al., 2005, Characterizing and Predicting Agents via Multi-Agent Evolution. Ann Arbor, MI, Altarum Institute. http://www.newvectors.net/staff/parunakv/BEE.pdf.

Parunak, H. V. D., S. Brueckner, et al., 2004, *Digital Pheromones for Coordination of Unmanned Vehicles*. Workshop on Environments for Multi-Agent Systems (E4MAS 2004), New York, NY, Springer. 246-263. http://www.newvectors.net/staff/parunakv/E4MAS04_UAVCoordination.pdf.

Rao, A. S. and M. P. Georgeff, 1991, *Modeling Rational Agents within a BDI Architecture*. International Conference on Principles of Knowledge Representation and Reasoning (KR-91), Morgan Kaufman. 473-484.

Stilman, B., 2000, *Linguistic Geometry: From Search to Construction*. Boston, Kluwer.

# AGENT-BASED META-MODELS

M.T. PARKER,[*] Argonne National Laboratory, Argonne, IL,
and Independent Consultant, East Hampton, NY
M.J. NORTH, Argonne National Laboratory, Argonne, IL,
and The University of Chicago, Chicago, IL
N.T. COLLIER, Argonne National Laboratory, Argonne, IL,
and PantaRei Corp., Cambridge, MA
T.R. HOWE, Argonne National Laboratory, Argonne, IL
J.R. VOS, Argonne National Laboratory, Argonne, IL,
and the University of Illinois at Urbana-Champaign, Urbana, IL

## ABSTRACT

The agent-based modeling and simulation (ABMS) community has long recognized a need for concise, complete, and implementation-neutral representations of agent-models, and for modeling tools that do not require significant computer programming experience. We discuss earlier efforts to address these needs, arguing that proposed representations were typically too high-level and did not cover behavior. It may be that these weaknesses were insurmountable at the time—and that it is only now, with the availability of relatively mature Domain Specific Languages (DSLs) and Model Driven Software Development (MDSD) tools that these needs may finally be met. We justify this claim by identifying significant issues modelers face in using General Purpose Languages (GPLs) for agent-based models and how these issues might be overcome by using DSLs. We describe the specific tools we intend to employ in that effort and how we plan to use those tools, and we propose a general meta-model for ABMS.

**Keywords:** Agent-based modeling and simulation, domain specific languages, model driven software development

## INTRODUCTION

Agent-based modeling often demands sophisticated computer programming skills, limiting adoption of this revolutionary technique. While high-level mathematics is a core training requirement for physical, life, and social sciences, programming rarely is—effectively denying the power of agent-based modeling to a large swath of researchers and arguably biasing explanations toward traditional equation-based approaches. In addition, while significant efforts have been made to develop frameworks that leverage programming resources and provide model exploration tools to non-programmers (Parker 2000; Inchiosa and Parker 2002), the models themselves typically have been expressed in general purpose languages such as Java, which as we argue below, are neither transparent nor particularly expressive under many usages. These issues, coupled with a lack of consistent representations, have made it quite difficult to share models and to expose them to outside review. The authors and others have long sought solutions to these issues.

---

[*] *Corresponding author address*: Miles T. Parker, 24 Gann Road, East Hampton, NY 11937; email: milesparker@gmail.com.

For example, Gulyás et al. discussed the potential of XML (e.g., Multi-Agent Modelling Language or MaML) as an agent-representation language as early as 1999. In the same period, Parker argued that strict compositions of hierarchical collections (“scapes”) with high-level spatial and execution model abstractions would facilitate the use of a declarative approach to model design in general (2000). More recently, the Repast team has demonstrated tools that allow extensive portions of model definitions to be made within XML (North et al. 2005). These approaches provide more transparent model representations, but there are typically still large gaps in what can be represented at such a high level. In particular, the proposed ABMS schemas typically have focused on structural issues and have left behavioral issues aside. As such, these schemas have not yet provided an effective and generic way to completely specify agent-based models. While one could argue that part of this weakness stems from the declarative nature of most of the proposals and that the difficulties encountered are largely representational, there may be other issues at hand.

## DOMAIN-SPECIFIC LANGUAGES AND META-MODELS

For many purposes, agents are naturally implemented as objects. Fortunately, object-oriented (OO) imperative languages are readily available to implement these objects. However, there are two important issues with this approach.

First, OO languages are general-purpose by design and so, to accommodate the universe of possible uses, carry complex syntax and semantics, very extensive Application Programming Interfaces (APIs), and idiomatic (but very general) usage patterns. Requiring such general and extensive knowledge is like requiring a person who simply wants to travel from Philadelphia to Boston to memorize the commercial route-map for the entire country and to become conversant with the maintenance procedures of the Boeing 737 to do so. Conversely, GPLs lack articulate idioms that fit a specific usage or context.[1] These kinds of idioms—e.g., shared agreement on the context of a particular communication and a “way of speaking” that uses this agreement to shape powerful abstractions and analogies—are what turn the symbols and structures of human language into artifacts of expressivity and beauty.[2] Doing without these idioms is like requiring text and instant messengers to use complete grammatical sentences with no abbreviations; or it is like requiring that the works of William Shakespeare be edited to replace any archaic language with contemporary “equivalents.”

Second, mainstream OO languages are compiled and static. We can distinguish these from “scripting” languages such as Javascript, Python, and Perl, which are interpreted and dynamic.[3] At the risk of making broad generalizations about controversial and complex issues, it can be argued that there are good reasons for these language design choices: they can encourage

1 For an excellent example of how modeling and DSLs can address this issue by using a simple state machine language, see Voelter’s (2006) article.

2 We leave discussion of what idioms may be most appropriate and powerful for ABMSs to a later paper.

3 Compiled languages convert language-level code to machine code in one step (i.e., at “compile-time”). Interpreted languages do so while a program is actually executing (i.e., at “runtime”). Java is thought of as a compiled language because its instructions are written in Java “byte-code,” a platform-neutral machine language for the Java runtime environment. Under certain circumstances, Java can also be compiled at runtime. We are ignoring many subtleties and caveats here.

code quality; have a dramatic effect on performance (e.g., orders of magnitude); prevent significant security issues; and aid modern development environment techniques such as code completion and refactoring, and so on. But compiled languages also have a strong net effect of forcing design decisions into build-time (i.e., they prevent users from making significant changes in structure, as opposed to state and behavior).

Note that it is frequently possible to allow design decisions to be made at runtime, at least when we know at compile-time what the set of potential design decisions is. But that almost always involves sophisticated architecture and significant, sometimes dramatic, runtime slowdowns. As a simple example, suppose we want to allow model developers to provide arbitrary properties to agents at runtime. We could give a pre-defined agent type a collection of properties, but any time we wanted to access those properties, we would have to do so indirectly, e.g., through a map lookup.[4] We would introduce significant complexity and performance costs, and would lose language support for types, behaviors, and so on. In effect, we would almost be designing our own language, but a clumsy, inefficient, and opaque one.

Until now, there have been three general approaches to these issues. One approach is to simply employ a dynamic, interpreted scripting language and accept the costs of doing so, as discussed above. For example, this strategy was employed successfully in NetLogo (Wilensky 1998). The second is to employ techniques such as Java reflection that allow users to discover and update object state arbitrarily at runtime. While this approach allows a great deal of flexibility, it again imposes a severe performance penalty without solving fundamental issues. The third approach is to use code-generation techniques to compile and load custom agent designs. This approach is used successfully in the core Repast Simphony (Repast S) environment to support the use of Java annotations in specifying agent state “watchers.” The Repast for Python Scripting (North et al. 2006a) environment had advantages of the first and third approaches, but was of necessity a focused solution designed to support beginning developers. While code generation can provide good performance, it is typically inflexible and very difficult to implement and maintain. As such, it is best left to well-defined key subsystems. But the basic code-generation approach is sound, as the next discussion demonstrates.

Many see an elegant and general solution to these issues in the employment of Domain Specific Languages (DSLs) that capture idiomatic expressivity and implied context. In the past, the development of special purpose languages has been limited since the path to their development requires very specialized knowledge and substantial effort. But recently, tools have emerged offering support for such languages at a deep level and allowing them to be defined and implemented in a relatively straightforward and consistent manner. These tools, which partially fulfill the generative programming vision and are variously referred to as intentional programming tools, software factories, meta-programming tools, or what Martin Fowler aptly terms “Language Workbenches” (LWs), may be exactly the tools we need to build our agent-based modeling language.

Toward this end, one of us explored purpose-built LWs but theoretical and practical evaluation of these tools was halted when it was realized that it could not be predicted when these tools would be mature or even generally available. Later, the Eclipse project’s Eclipse Modeling Framework (EMF) was recognized as a candidate for this effort. EMF seemed to

4 In fact, this was considered in the earliest designs for Ascape, circa 1997.

support at least the sub-problem of modeling such a language's structure[5] and provides robust code-generation facilities to support it. For those unfamiliar with code-generation and other techniques mentioned below, the next section provides a brief explanation. Initially, experimentation focused on the Java Emitter Template (JET) toolset for the definition of code-generation mechanisms. JET is essentially a version of Java Server Pages modified for Java template use, and while this approach seemed workable, it might not provide adequately transparent, maintainable, or modular support for template definitions. Fortunately, a different toolset, Open Architecture Ware (oAW), has more recently joined Eclipse's official project. oAW provides a rich, problem-specific "template" language for code-generation called "Xpand2," a powerful constraints language called "Check," a model transformation language called "Xtend," and other supporting tools, including validation rules through the "Recipe Framework," and a workflow engine to tie the entire process together. More serendipitously, oAW can be extended with a language for defining DSLs using high-level Backus-Naur form (BNF) notation and can use this definition easily to build the kinds of supporting features that contemporary developers are beginning to demand, such as live syntax checking, syntax highlighting, and code completion. It is this toolset that we are now actively employing as we develop our new MDSD/DSL system, which we describe in outline in the next section.

## THE REPAST SIMPHONY "SCORE" ENVIRONMENT

Our pilot effort[6], dubbed "Score,"[7] utilizes EMF and oAW to provide the support we need for a complete ABMS environment and language. We now describe the current design in detail.

Figure 1 depicts the tool-chain we currently envision. Please note that we are describing how we will use the meta-model before we describe the meta-model itself, since it may inform readers about why the meta-model looks the way it does.

First, note that the model makes use of the Repast S code-base (ROAD [Repast Organization for Architecture and Design] 2006); that is, the set of APIs that a model developer can use to create and run a complete, Java-coded Repast S model. This is important because it ensures that there are no dependencies of the essential parts of the Repast S code base on the EMF or Eclipse systems. Next, we have the Score meta-model, an instantiation of EMF's Ecore meta-model that encapsulates everything we need to know about a model. For example, a model developer may specify a type of agent that has a happiness attribute and moves upon a 2-dimensional grid. Next, agent modelers design models for their particular problem space using EMF tools, such as the basic tree editor depicted in Figure 4; purpose-built graphical tools designed in the Eclipse Graphical Modeling Framework; plain XML; or a DSL implemented by using xText.

5 The question of whether we are and should be developing first and foremost a "model of" or a "language for" agent-based modeling is deep and complex. We will not address these issues here, but interested readers could start with Fowler's companion posting on "Language Workbenches and Model Driven Architecture."

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.

7 Credit or blame to Howe, "Ensemble" (Parker) has also been proposed in keeping with an analogizing context while "milieu" and "gestalt" remain candidates.

Here we take special note of the upper half of the diagram. The core modeling component is targeted for our pilot. We will describe the initial release and the runtime component more fully in a later paper. Briefly, we plan on using Xtend, a model transformation language, to

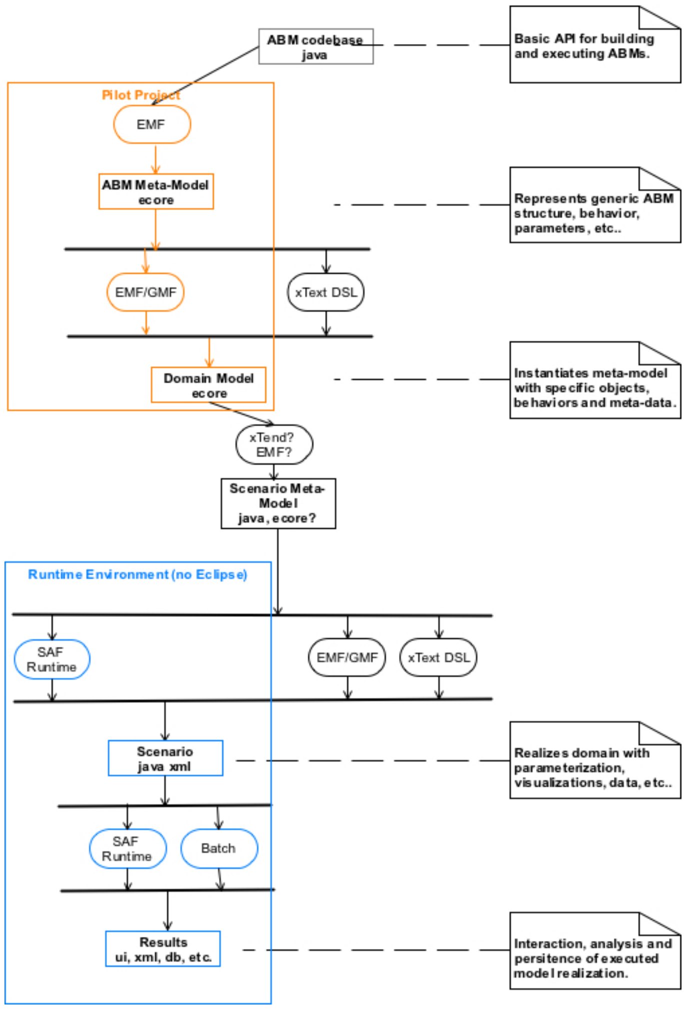

ABM codebase
java
Basic API for building and executing ABMs.
Pilot Project
EMF
ABM Meta-Model
ecore
Represents generic ABM structure, behavior, parameters, etc..
EMF/GMF
xText DSL
Domain Model
ecore
Instantiates meta-model with specific objects, behaviors and meta-data.
xTend?
EMF?
Scenario Meta-Model
java, ecore?
Runtime Environment (no Eclipse)
SAF
Runtime
EMF/GMF
xText DSL
Scenario
java xml
Realizes domain with parameterization, visualizations, data, etc..
SAF
Runtime
Batch
Results
ui, xml, db, etc.
Interaction, analysis and persitence of executed model realization.

**FIGURE 1** Score activity diagram

infer a scenario model from the domain model. For example, suppose that our modeler includes a 2-dimensional grid in his or her model. The domain model will then include a slot in the scenario meta-model[8] for defining for each scenario what size the scenario's grid should be or the allowed range of sizes for the scenario's grids. Next the modeler or researcher can define specific scenarios to run interactively (e.g., using real-time visualization) or in batch mode (e.g., a parameter sweep). The results of these models can then be externalized and the models used to guide interpretation of these results.

8 It is certainly possible that we will find that a straightforward object model could suffice for the definition of arbitrary scenarios, but the services provided by EMF (e.g., built-in editors) and persistence alone seem to justify its use here.

Figure 2 provides greater detail about how different kinds of users will interact with the Repast S system as a whole and the dependencies between different tools. The exposition that follows focuses mainly on the block of the system that represents the modeling process itself. First, the Repast team or outside contributors develop the core Repast S API, the Score meta-model which makes use of the Repast S API, and the templates that will be used for model transformation.

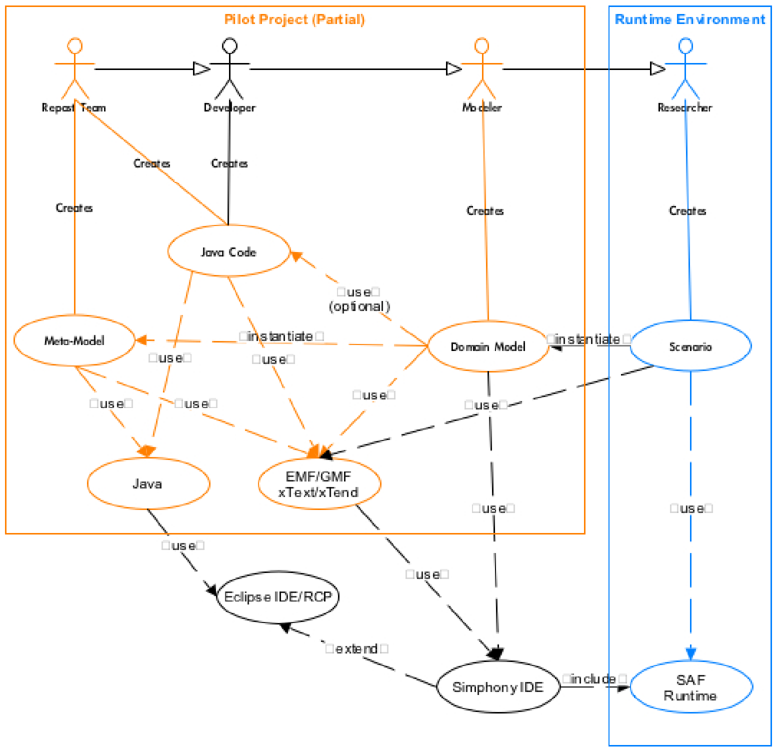


**FIGURE 2** Score use-case diagram

There is an intermediate role for a developer who develops models and Repast S extensions, continuing to use the Java API directly. The modeler is a new entity who, up to now, has been imagined mostly as a renaissance person who is both a master of his or her domain and a software developer. The approach discussed in this paper allows modelers to focus on their domain rather than on software development by letting them create meta-models directly by using the graphical and textual tools described above. Finally, we have the researcher. With the existing toolsets, this scholar must find a graduate student somewhere to "code up" his or her model. Guided by the meta-information provided by the Score and Scenario meta-models, the Repast S system can now provide much more sophisticated tools to aid in model exploration and offers the researcher a natural path to the role of modeler.

Fundamentally, the Score meta-model captures the specification needed to generate a complete agent model. The meta-model is depicted in a UML object diagram in Figure 3. The key to understanding such meta-models is to recognize that they are not describing agent simulations or even the object structure for agent simulations, but a model sufficient to inform the creation of such simulations. For example, SField does not represent an actual agent field, but the meta-data we would need to describe some agent variable (e.g., the name of the field, its data type, its allowed ranges, and so on). Consider further that these values are themselves represented in Ecore, this meta-model's meta-model, as EAttributes carrying similar meta-data. This field then would be used along with other meta-data to create an object model for the actual agent with an actual "happiness" member variable. Similar descriptions can be given for other model meta-objects. In some cases, however, there is a direct one-to-one correspondence. For example, for every SValueLayer defined in the model, there will be an actual ValueLayer in our domain agent model.

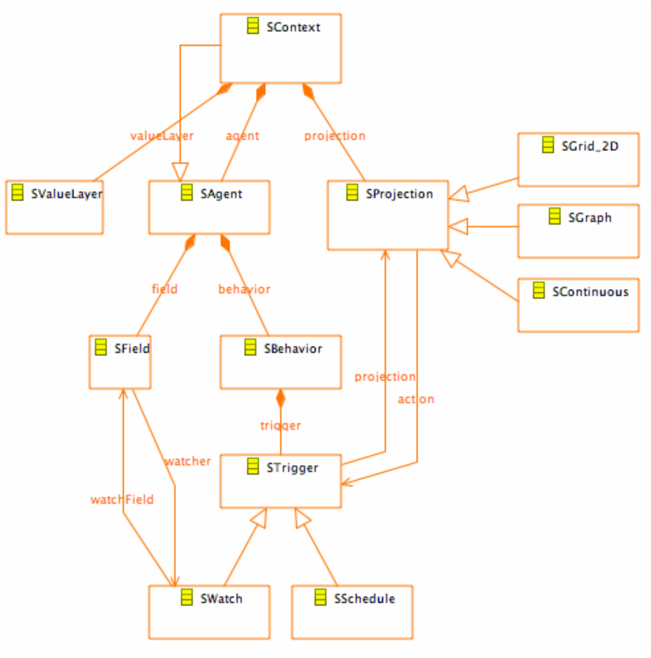
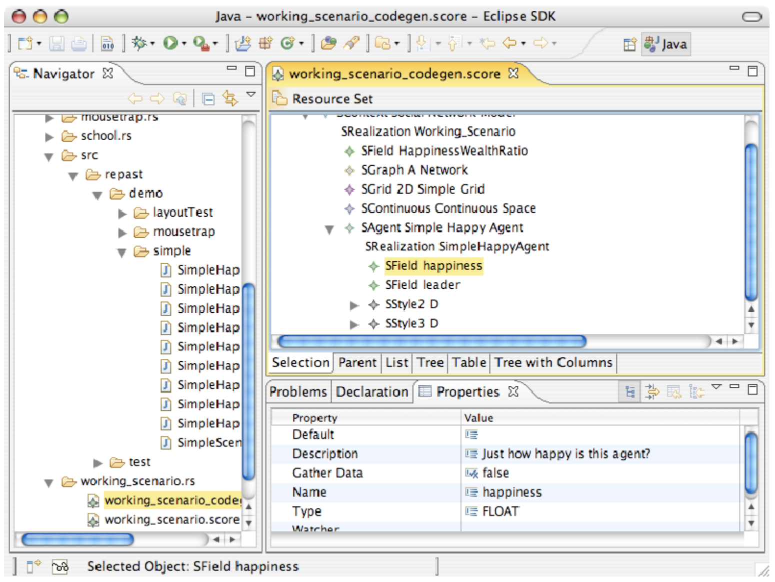


**FIGURE 3** Score meta-model



```
mousetrap.rs
school.rs
src
  repast
    demo
      layoutTest
      mousetrap
      simple
        SimpleHap
        SimpleHap
        SimpleHap
        SimpleHap
        SimpleHap
        SimpleHap
        SimpleHap
        SimpleHap
        SimpleHap
        SimpleScen
  test
working_scenario.rs
  working_scenario_codeg
  working_scenario.score
```

```
SRealization Working_Scenario
  SField HappinessWealthRatio
  SGraph A Network
  SGrid 2D Simple Grid
  SContinuous Continuous Space
  SAgent Simple Happy Agent
    SRealization SimpleHappyAgent
    SField happiness
    SField leader
    SStyle2 D
    SStyle3 D
```

| Property | Value |
|---|---|
| Default | |
| Description | Just how happy is this agent? |
| Gather Data | false |
| Name | happiness |
| Type | FLOAT |
| Watcher | |

Selected Object: SField happiness

**FIGURE 4** Simple Score tree editor

## "SCORE" IN ACTION

We provide a very preliminary walk-through for the initial incarnation of Score, excluding from this discussion many of the tools envisioned above, such as a DSL textual language. First, an agent modeler defines a simple model using either a basic tree-based modeler or an XML editor. In fact, it is instructive to compare the two.

In the tree editor, the modeler can add various components to a hierarchy (see Figure 4). As discussed elsewhere, the Repast S context approach is naturally hierarchical, but more complex graph relationships certainly can exist beneath the hierarchy (North et al. 2006b). Future GUI modeling tools may provide more complex editors for these structures, but for reasons outlined in Howe et al. (2006), we believe that hierarchies defined through contexts will remain the most natural and effective representation. Ultimately, the user's model is stored in an XML/XMI file, and users can also edit this file directly and employ a referenced schema to enforce basic constraints, which assure that the XML file can be parsed into an instantiation of the meta-model.

After we have developed a complete model, we can use EMF's built-in XMI persistence scheme to easily create a Java object instantiation. This step is so much simpler than typical approaches to XML-Java and mappings that it is essentially transparent. At this point, we have objects for all of the meta-data (e.g., types, agent names, built-in components, etc.) that we need for creating a complete model specification as defined in the initial Repast S roadmap.[9] That, coupled with the flexibility we will gain in evolving the model specification into the future, may alone be worth the initial effort in integrating EMF into our tool-chain. However, we should note that we have only implemented a prototype of the template code that will drive our code-generation; we will leave detailed discussion and examples for a future paper.

By using the oAW workflow engine, we should be able to integrate the model development process into a seamless development experience within the Eclipse environment. The Check language and Recipe framework provide a way to give model developers real-time feedback, immediately notifying them of incorrect or ill-advised edits and providing meaningful hints on how to correct them.

The meta-model will next be run through our Xpand2 templates, producing Java source code for our actual agents. For example, from the above SimpleHappy model we could create source code for a SimpleHappyAgent, including lines for variable and getters and setters for the agent based on text patterns we have defined. But we could do much more than that; our agent could have direct references to its parent context(s) and their state and could very efficiently support listeners (watchers) for agent state, and we can generate parent context-level variables (e.g., minHappiness, maxHappiness, happinessDistribution) to randomly initialize each SimpleHappyAgent's state. We note, however, that our agents can be plain old Java objects and thus be transparent and full-fledged members of the Java world; they can support JavaDoc; contain inspection, reflection, and other persistence mechanisms; and integrate into production

9 There are two modes that can be used in realizing the actual code used in a running model. The more sophisticated usage we are focusing on in this paper involves generating the code for model components and then loading them into the runtime environment. But we can also simply use "pre-built" Java classes directly by setting the realization mode attribute to "LOAD." This usage is the one supported by the walk-through to this point.

and enterprise environments and toolsets. Finally, we can use the model to generate efficient and appropriate implementations for runtime tools such as visualizers and data collection, etc.[10]

Most importantly, we can realize arbitrarily complex behaviors for our agents that can be transformed into native Java syntax. In our first implementation, we plan to provide a simple but quite focused Java-like syntax for specifying many of these behaviors. We anticipate that sophisticated models for rich and expressive behavior will be a very active area of future research and development.

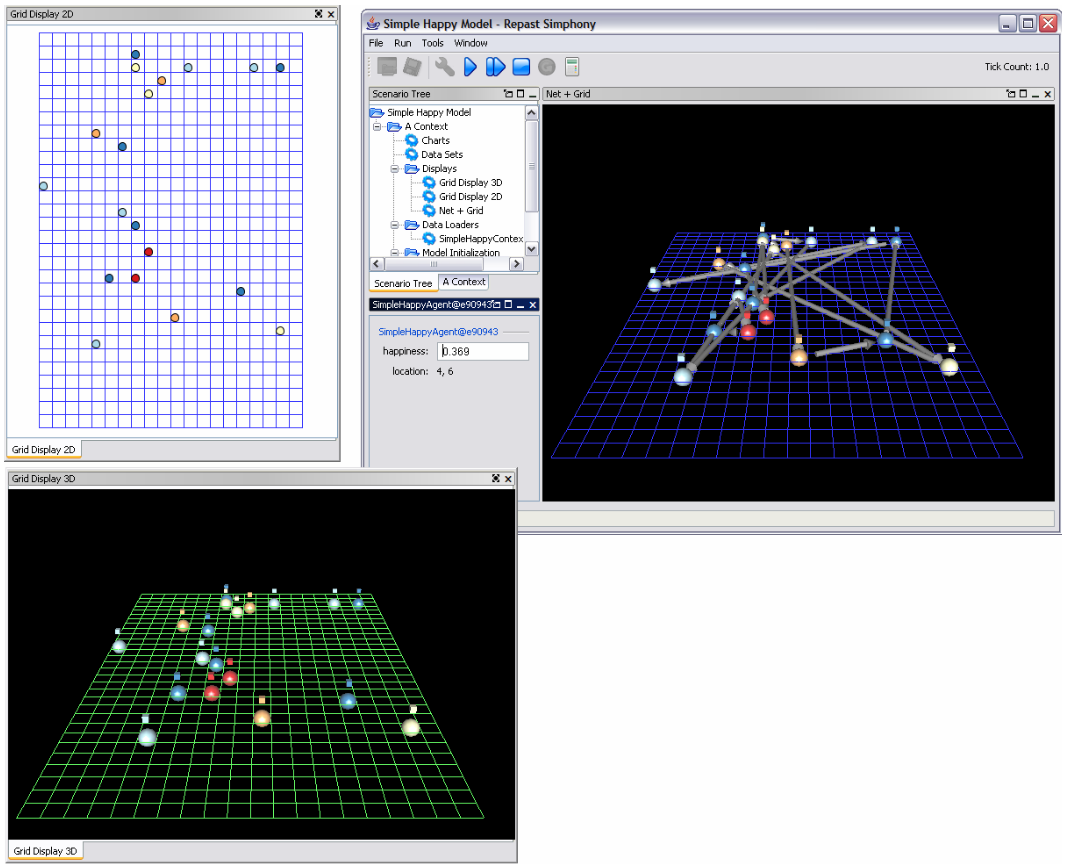
At this stage, the meta-model will also be transformed through Xtend into a scenario Ecore meta-model that can then be used to provide the same kinds of tools as outlined above but at the level of model instantiation and execution. Researchers will be able to define realizations within their own models (e.g., by adding specific agents to a particular context, specifying batch constraints, and so forth). As mentioned above, some of this work can be (and has been) done using a generic object model, but employing a meta-level specification for each model may provide much greater leverage and cleaner abstractions for developing such supporting tools. For example, we could eliminate Java reflection and easily be able to generate XML Schemas for individual model batch runs.

```
<?xml version=“1.0” encoding=“UTF-8”?>
<repast.score:SContext  xmi:version=“2.0”  xmlns:xmi=“http://www.omg.org/XMI“
xmlns:xsi=“http://www.w3.org/2001/XMLSchema-instance“
xmlns:repast.score=“http://repast.score“ name=“Social Network Model”>
 <realization    package=“repast.demo.simple”    className=“Working_Scenario”
path=“..\plugins\repast.test.models” mode=“GENERATE”/>
 <field name=“HappinessWealthRatio” type=“FLOAT” default=“.45”/>
 <projection xsi:type=“repast.score:SGraph” name=“A Network”/>
 <projection  xsi:type=“repast.score:SGrid_2D”  name=“Simple  Grid”  xSize=“10”
ySize=“10”/>
 <projection xsi:type=“repast.score:SContinuous” name=“Continuous Space”/>
 <agent name=“Simple Happy Agent”>
 <realization className=“SimpleHappyAgent” path=““ mode=“GENERATE”/>
 <field  name=“happiness”  description=“Just  how  happy  is  this  agent?”
type=“FLOAT” default=““/>
 <field name=“leader” default=““/>
 <style xsi:type=“repast.score:SStyle2D”>
 <realization className=“SimpleHappyNodeStyle2D” path=““ mode=“LOAD”/>
 </style>
 <style xsi:type=“repast.score:SStyle3D”>
 <realization className=“SimpleHappyNodeStyle” path=““ mode=“LOAD”/>
 </style>
 </agent>
</repast.score:SContext>
```

**FIGURE 5** An example Score XML file

10 Beyond the structural and behavioral aspects described above, there are many more intriguing code-generation dimensions that are far outside the scope of this paper.

Finally, the researcher will actually execute the model by using either a stand-alone runtime as shown in Figure 6, a batch runner, or a even a possible model runtime environment fully integrated with the Eclipse environment. Data and other artifacts collected during the model run can then be interpreted and manipulated using scenario and model meta-data.



Tick Count: 1.0

SimpleHappyAgent@e90943

happiness: 0.369

location: 4, 6

**FIGURE 6** Running model

## CONCLUSION

We advocate an approach to software development that may strike some as arcane and overly complex. The possibilities for “meta-confusion” may seem endless, and the specter of proliferating “little languages” may cause panic for those who believe that ubiquitous standards and generic UML are the answer to every problem. However, the approach discussed in this paper is not an “all-or-nothing” proposition. Developers interested in Java-level programming can continue to use the Repast Java APIs directly, mixing in generated code as appropriate and making use of a powerful new agent-specific language as time and interest permit. And modelers without traditional programming skills will finally have a complete set of tools for generating agent-based models and sharing the insights these models provide into our complex world.

## ACKNOWLEDGMENT

This work is supported by the U.S. Department of Energy, Office of Science, under contract W-31-109-Eng-38.

## REFERENCES

Gulyás, L., T. Kozsik, and J.B. Corliss, 1999, "The Multi-Agent Modeling Language and the Model Design Interface," *The Journal of Artificial Societies and Social Simulation*, Vol. 2, No. 3.

Howe, T.R., N.T. Collier, M.J. North, M.T. Parker, and J.R. Vos, 2006, "Containing Agents: Contexts, Projections, and Agents," *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects*, co-sponsored by Argonne National Laboratory and The University of Chicago, Argonne, Sept. 21–23 (in press).

Inchiosa, M.E. and M.T. Parker, 2002, "Overcoming Design and Development Challenges in Agent-based Modeling Using Ascape," *Proceedings of the National Academy of Sciences*, **99**: Suppl. 3, 7304-7308.

North, M.J., T.R. Howe, N.T. Collier, and J.R. Vos, 2005, "The Repast Simphony Runtime System," *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 (in press).

North, M.J., N.T. Collier, and J.R. Vos, 2006a, "Experiences Creating Three Implementations of the Repast Agent Modeling Toolkit." *ACM Transactions on Modeling and Computer Simulation*, vol. 16, issue 1, pp. 1–25.

North, M.J., T.R. Howe, N.T. Collier, J.R. Vos, and M.T. Parker, 2006b, "Spaces, Places, and the Lernaean Hydra of Agent-Based Modeling," *North American Association for Computational Social and Organizational Science 2006 Conference*, Argonne National Laboratory, Argonne, IL.

Parker, M., "Ascape: Abstracting Complexity," *Brookings Institution Report*, March 2000, and *SwarmFest Proceedings 2000*, Utah State University, Logan Utah; available at http://www.brookings.edu/es/dynamics/models/ascape/20000301_ascape.htm.

ROAD (Repast Organization for Architecture and Design), 2006, Repast Home Page, Chicago, IL,; available at http://repast.sourceforge.net/.

Voelter, M., B. Kolb, S. Efftinge, and A. Haase, 2006, "From Front End to Code—MDSD in Practice," available at http://www.eclipse.org/articles/Article-FromFrontendToCode-MDSDInPractice/article.html.

Wilensky, U., 1998, *NetLogo*, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

## Repast Toolkit Developments

# AN INTRODUCTION TO REPAST SIMPHONY MODELING USING A SIMPLE PREDATOR-PREY EXAMPLE

ERIC TATARA, Argonne National Laboratory, Argonne, IL
M.J. NORTH,[*] Argonne National Laboratory, Argonne, IL,
and The University of Chicago, Chicago, IL
T.R. HOWE, Argonne National Laboratory, Argonne, IL
N.T. COLLIER, Argonne National Laboratory, Argonne, IL,
and PantaRei Corp., Cambridge, MA
J.R. VOS, Argonne National Laboratory, Argonne, IL,
and the University of Illinois at Urbana-Champaign, Urbana, 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 wolf-sheep predation as an introductory tutorial and illustration of the modeling capabilities of Repast S.

**Keywords**: Agent-based modeling and simulation, Repast, toolkits, and development environments

## INTRODUCTION

Repast (ROAD 2005) 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 (North et al. 2006). 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, Howe, Collier, and Vos (2005a and 2005b) provide an overview of the Repast S runtime and development environments.

We use a model of wolf-sheep predation to demonstrate the capabilities of the Repast S toolkit and as an introductory tutorial. While the example is not intended to model real phenomenon, the model's complexity is high enough to illustrate how the user may develop multi-agent models. Spatial and temporal patterns emerge in the model consisting of potentially hundreds of instances of three agent types.

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.

- 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 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

- An integrated two-dimensional or three-dimensional geographical information system (GIS), and other model views

- Automated connections to enterprise data sources

- Automated connections to powerful external programs for conducting statistical analysis and visualizing model results.

## WOLF-SHEEP PREDATION MODEL

We implement a model of wolf-sheep predation (Wilenski 1998) in Repast S as a demonstration of the toolkit's capabilities. This model represents a simple variation of predator prey behavior using three agents: wolves, sheep, and grass. Both the wolves and sheep move randomly on a grid, and the movement has a cost in the form of lost energy. The wolves and sheep need to eat food in order to replenish their energy, and they will die once their energy level reaches zero.

Wolves prey on sheep and may eat them if the two are located in the same spatial position, thereby increasing the wolf's energy level. Sheep may similarly eat grass if the sheep is located on a patch which contains living grass. Once a sheep eats the grass in its location, the grass needs to regrow before the sheep can eat it again. Repast S models a re-grow rate for grass by counting down after the grass has been eaten in a specific location. Reproduction is modeled by a random process that creates a child from the parent, divides the energy of the parent agent in half, and assigns the energy equally to the parent and child.

## REPAST S IMPLEMENTATION

North et al. (2005a and 2005b) provide details on the Repast architecture and general modeling concepts. The Repast S implementation of the wolf-sheep predation model consists of several classes of interest, which include, specifically, the data loader and agent classes.

The context creator class, which implements DataLoader, constructs the main context and returns it to the Repast run environment as shown in Figure 1. Within the user-designed context creator, modelers create a root context and add the desired agents and projections to the context. In this example, we add a Grid projection to the context to model the world using a discrete Cartesian grid.

Next, we add the agents (i.e., the wolf, sheep, and grass) to the context. We obtain the simulation parameters, which may be set in the graphical user interface (GUI), from the run environment to set the initial number of wolves and sheep. Finally, we add a grass agent to each location on the grid and return the context to the run environment.

## SIMPLE AGENTS

The three agent classes in the model extend the SimpleAgent class, which contains common methods related to movement and death, for example. The SimpleAgent class also contains Repast-specific `@ScheduleMethod` annotation, which precedes methods to be scheduled. The `@ScheduleMethod` annotation has several options, including the start time and the updated interval. The SimpleAgent step method has an annotation that specifies the method to be scheduled starting at tick 1 and to recur every one tick thereafter (see Figure 2). In this implementation of the model, the SimpleAgent class has an empty step method that we can override, so we can specify the individual step behavior in subclasses.

Movement on the grid is accomplished by the move method, which simply obtains the Grid object from the agent's context and then obtains the agent's coordinates on the grid. The agent moves by randomly selecting one of the nearest eight neighboring grid positions (i.e., the agent's Moore neighborhood). The "die" method occurs when either the agent's energy level reaches zero or, in the case of sheep, when the agent is eaten. We model the death process by simply removing the agent from its context.

The Wolf class shown in Figure 3 has two constructors. We use the first when the wolf is created from a reproduction process. The energy from the parent wolf is passed into the

```
public class PPContextCreator implements DataLoader {
  public Context create(Object creatorID) {
    int xdim = 50;
    int ydim = 50;

    Context<SimpleAgent> context = Contexts.createContext(SimpleAgent.class, creatorID);
    Projections.createGrid("Simple Grid", context,new DefaultGridParameters<SimpleAgent>(
                           new RandomGridAdder<SimpleAgent>(), true, xdim, ydim));

    Parameters p = RunEnvironment.getInstance().getParameters();
    int numWolves = (Integer) p.getValue("initial number of wolves");
    p = RunEnvironment.getInstance().getParameters();
    int numSheep = (Integer) p.getValue("initial number of sheep");

    for (int i = 0; i < numWolves; i++) {
      Wolf wolf = new Wolf();
      context.add(wolf);
    }
    for (int i = 0; i < numSheep; i++) {
      Sheep sheep = new Sheep();
      context.add(sheep);
    }

    AgentCounter counter = new AgentCounter();
    context.add(counter);

    Grid grid = (Grid) context.getProjection("Simple Grid");

    for (int i=0; I < xdim; i++){
      for (int j=0; j < ydim; j++){
        Grass grass = new Grass();
        context.add(grass);
        grid.move(grass, i, j);
      }
    }

    return context;
  }

  public void load(Context context) {
  }
}
```

**FIGURE 1** Repast S context creator for the predator-prey model

constructor and is assigned to the child. We use the second constructor for wolves that are created during model initialization, in which the initial energy is randomized. The step method of the Wolf class overrides the SimpleAgent method. Since the SimpleAgent method is already scheduled, the Wolf class step method does not require an additional annotation. The step method consists of the following processes: moving, reducing energy, catching sheep, reproducing, and dying. (The movement and death methods are described in the SimpleAgent class section.)

A wolf may catch a sheep to eat if a sheep exists on the same grid coordinate as the wolf. The wolf agent obtains the Grid object from its context and scans through the objects at its location. If there is a sheep at the wolf's location, the wolf eats the sheep and increases its energy

```
public class SimpleAgent {
  private int x;
  private int y;

  @ScheduledMethod(start = 1, interval = 1)
    public void step() {
    // override by subclasses
  }

  public void move() {
    Context context = ContextUtils.getContext(this);
    Grid grid = (Grid) context.getProjection(“Simple Grid”);
    GridDimensions dims = grid.getDimensions();
    GridPoint point = grid.getLocation(this);

    x = point.getX() + (int) Math.round(2*Math.random() - 1);
    y = point.getY() + (int) Math.round(2*Math.random() - 1);

    grid.move(this, x, y);
  }

  public void die(){
    Context context = ContextUtils.getContext(this);
    if (context.size() > 1)
      context.remove(this);
    else
      RunEnvironment.getInstance().endRun();
  }

  public int getX() {
    return x;
  }

  public int getY() {
    return y;
  }
}
```

**FIGURE 2** The SimpleAgent class

according to the gain set in the parameters list. Repast S models reproduction by a random process which generates a random number, and if the number is less than the wolf reproduction rate, a new wolf is created. The energy level of the parent is halved and given in equal parts to the parent and child.

Sheep behavior is very similar to wolf behavior and is specified in the Sheep class as shown in Figure 4. The only major difference between the Wolf and Sheep classes is the manner by which they acquire energy. The sheep scan their location for grass agents and, if the grass is alive, the sheep eat the grass and increase their energy levels. When the grass is eaten, the grass agent’s state is switched from alive to dead.

The Grass class shown in Figure 5 again overrides the step method of SimpleAgent. It has a single constructor which randomizes the countdown timer for re-growth and sets the initial state of the grass to alive or dead. The step method simply checks whether or not the grass is dead. If it is, the step method reduces the re-grow timer *or* switches the state of the grass to alive if the timer has expired. If the grass is alive, the step method does not execute any code.

```
public class Wolf extends SimpleAgent {
  private double energy;

  public Wolf (double energy){
    this.energy = energy;
  }

  public Wolf(){
    Parameters p = RunEnvironment.getInstance().getParameters();
    double gain = (Double) p.getValue(“wolf gain from food”);
    energy = Math.random() * 2 * gain;
  }

  @Override
  public void step() {
    Context context = ContextUtils.getContext(this);

    move();

    energy = energy - 1;  // Reduce energy

    // Catch sheep
    int x = getX();
    int y = getY();

    Parameters p = RunEnvironment.getInstance().getParameters();
    double gain = (Double) p.getValue(“wolf gain from food”);

    Grid<SimpleAgent> grid = (Grid<SimpleAgent>) context.getProjection(“Simple Grid”);

    Sheep sheep = null;
    for (SimpleAgent agent : grid.getObjectsAt(x,y)){
      if (agent instanceof Sheep)
        sheep = (Sheep) agent;
    }
    if (sheep != null){
      sheep.die();
      energy = energy + gain;
    }

    // Reproduce
    p = RunEnvironment.getInstance().getParameters();
    double rate = (Double) p.getValue(“wolf reproduce”);

    // Spawn
    if (100 * Math.random() < rate){
      energy = energy / 2;
      Wolf wolf = new Wolf(energy);
      context.add(wolf);
    }

    // Death
    if (energy < 0) die();

  }
}
```

**FIGURE 3** The Wolf class

```
public class Sheep extends SimpleAgent {
  double energy;

  public Sheep(double energy){
    this.energy = energy;
  }

  public Sheep(){
    Parameters p = RunEnvironment.getInstance().getParameters();
    Double gain = (Double) p.getValue(“sheep gain from food”);

    energy = Math.random() * 2 * gain;
  }

  @Override
  public void step() {
    Context context = ContextUtils.getContext(this);

    move();

    energy = energy - 1;  // Reduce energy

    // Eat Grass
    int x = getX();
    int y = getY();

    Parameters p = RunEnvironment.getInstance().getParameters();
    double gain = (Double) p.getValue(“sheep gain from food”);

    Grid<SimpleAgent> grid = (Grid<SimpleAgent>) context.getProjection(“Simple Grid”);

    Grass grass = null;
    for (SimpleAgent agent : grid.getObjectsAt(x,y)){
      if (agent instanceof Grass)
        grass = (Grass) agent;
    }
    if (grass != null && grass.isAlive()){
      grass.setAlive(false);
      energy = energy + gain;
    }

    // Reproduce
    p = RunEnvironment.getInstance().getParameters();
    double rate = (Double) p.getValue(“sheep reproduce”);

    // Spawn
    if (100 * Math.random() < rate){
      energy = energy / 2;
      Sheep sheep = new Sheep(energy);
      context.add(sheep);
    }

    // Death
    if (energy < 0) die();

  }
}
```

**FIGURE 4** The Sheep class

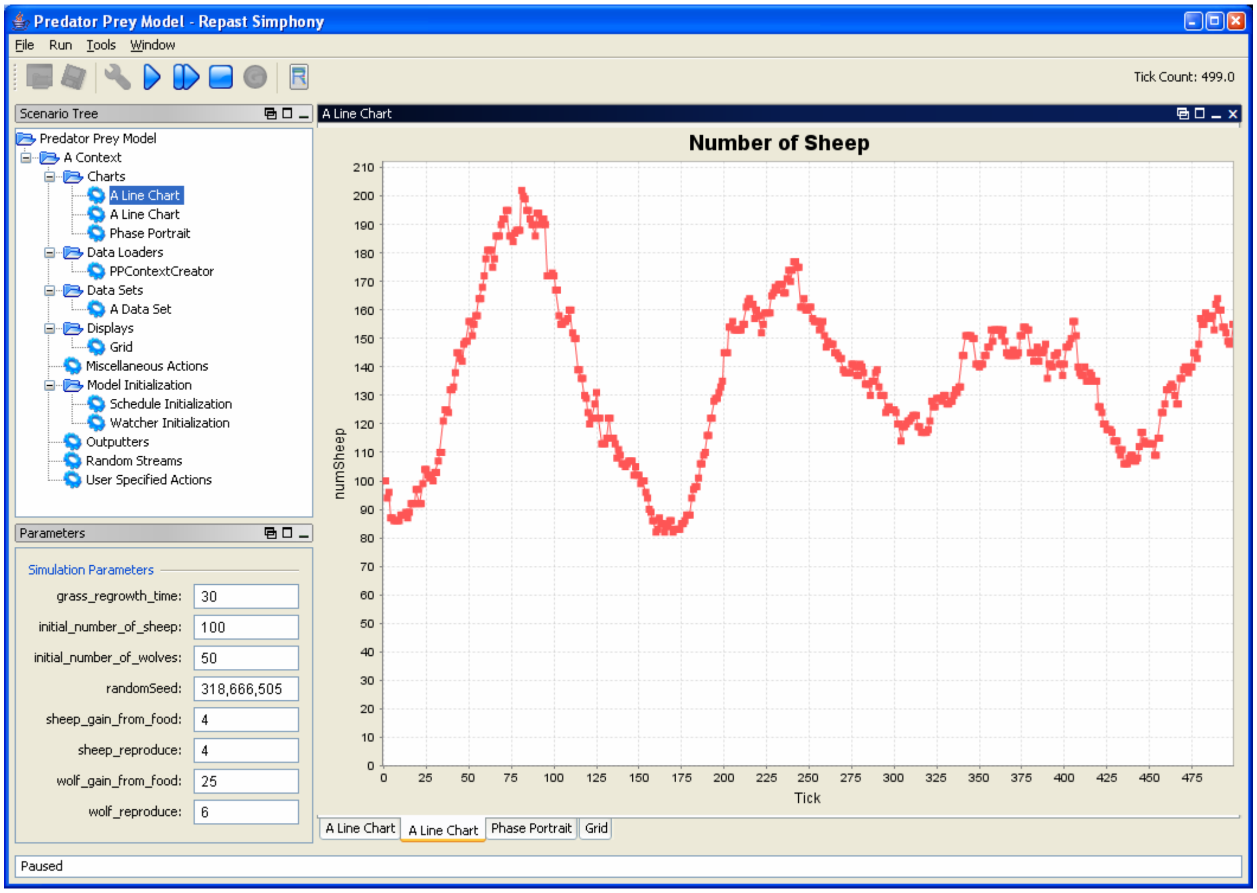
```
public class Grass extends SimpleAgent {
  private int countdown;
  private boolean alive;

  public Grass(){
    Parameters p = RunEnvironment.getInstance().getParameters();
    int regrowTime = (Integer) p.getValue(“grass regrowth time”);

    countdown = (int) (Math.random() * regrowTime);

    if (Math.random() <= 0.5) alive = true;
    else alive = false;
  }

  @Override
  public void step(){
    if (!alive){
      if (countdown <= 0){
        Parameters p = RunEnvironment.getInstance().getParameters();
        int regrowTime = (Integer) p.getValue(“grass regrowth time”);

        alive = true;
        countdown = regrowTime;
      } else {
        countdown--;
      }
    }
  }

  public boolean isAlive() {
    return alive;
  }

  public void setAlive(boolean alive) {
    this.alive = alive;
  }
}
```

**FIGURE 5** The Grass class

## MODEL EXECUTION AND OUTPUT

Once the essential model classes are created, the user may load them into the Repast S runtime shown in Figures 6 and 7. The Repast GUI has a series of buttons along the top menu bar for model loading; setup; playing and pausing; stepping; stopping; and resetting. The tree in the top left pane displays its model components graphically and is interactive, allowing users to create and add various components to the model. The lower left pane shows the table of simulation parameters that users can modify during model execution. Parameters would typically include global model data, such as the initial number of agents or properties that are common to classes of agents, rather than properties associated with an individual agent instance.

The right side of the GUI may contain one or more graphical representations of model data, including time-series charts; bar charts; and two- and three-dimensional grid and network projections. Figure 6 shows the population of sheep over time, starting at tick 0. The oscillations in the population are typical of the dynamic behavior observed in predator-prey systems.



| | |
|---|---|
| grass_regrowth_time: | 30 |
| initial_number_of_sheep: | 100 |
| initial_number_of_wolves: | 50 |
| randomSeed: | 318,666,505 |
| sheep_gain_from_food: | 4 |
| sheep_reproduce: | 4 |
| wolf_gain_from_food: | 25 |
| wolf_reproduce: | 6 |

**FIGURE 6** A series chart displaying the sheep population over time

Figure 7 shows a three-dimensional display of the grid projection with the wolves and sheep represented by three-dimensional objects. We organize the grass agents on a plane such that one grass agent is positioned at each discreet grid location. The model shown here uses a discrete 50 by 50 grid, with 100 initial sheep and 50 initial wolves, whose spatial positions are randomly determined during the model initiation.

Modelers can update displays at regular user-specified intervals or whenever a move event causes the spatial position of one of the agents to change. Modelers can also undock the two- and three-dimensional displays from the Repast GUI to provide multiple simultaneous model displays. Data may also be saved via one of several types of data logging "outputters." The user may create or modify existing data using these outputters in the GUI model tree. North et al. (2005a and 2005b) provide additional detail on these features.

## 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

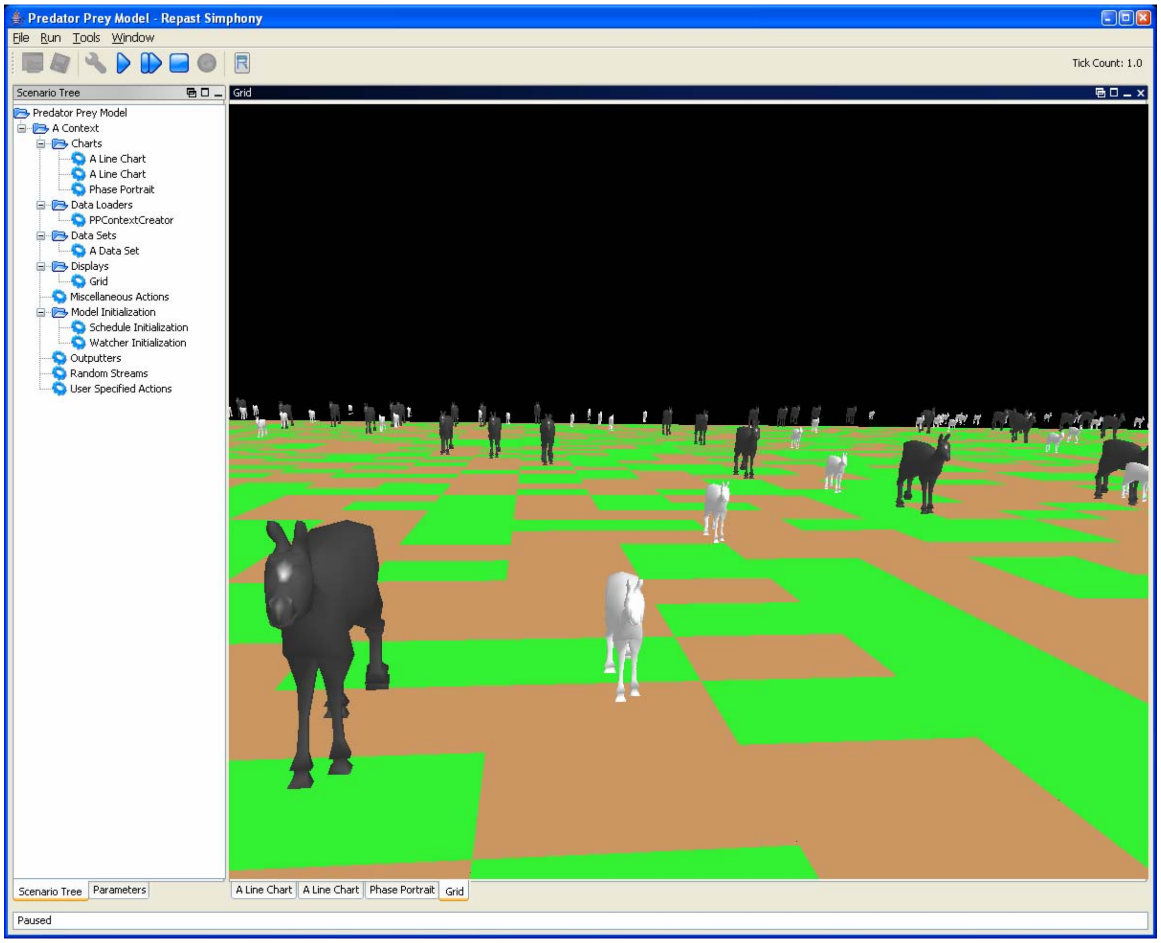


**FIGURE 7** 3D display of model grid. Sheep are light and wolves are dark. Light and dark squares represent living and dead grass, respectively.

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 modeling capabilities of Repast S using a simple model of wolf-sheep predation. We describe a specific implementation of the model in Repast S with three agent classes by using detailed model source code.

## 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. This work is supported by the U.S. Department of Energy, Office of Science, under contract W-31-109-Eng-38.

## REFERENCES

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.

Repast, ROAD (Repast Organization for Architecture and Design), 2005, Repast Home Page, Chicago, IL; available at http://repast.sourceforge.net.

Wilensky, U., 1998, "NetLogo Wolf Sheep Predation Model," Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL; available at http://ccl.northwestern.edu/netlogo/models/WolfSheepPredation.

# LEGACY MODEL INTEGRATION WITH REPAST SIMPHONY

M.J. NORTH,* Argonne National Laboratory, Argonne, IL,
and The University of Chicago, Chicago, IL
P. SYDELKO, Argonne National Laboratory, Argonne, IL,
J.R. VOS, Argonne National Laboratory, Argonne, IL,
and the University of Illinois at Urbana-Champaign, Urbana, IL
T.R. HOWE, Argonne National Laboratory, Argonne, IL
N.T. COLLIER, Argonne National Laboratory, Argonne, IL,
and PantaRei Corp., Cambridge, MA

## ABSTRACT

Repast is a widely used, free, and open-source, agent-based modeling and simulation toolkit. Three Repast version 3 platforms are currently available, each of which has the same core features but with differing environments for these features. Repast Simphony (Repast S) extends the Repast 3 portfolio by offering a new approach to simulation development and execution. Repast S's new simulation development capabilities include direct support for integrating existing (i.e., legacy) file-based models into agent-based simulations. This paper reviews related work on model integration and data exchange; introduces the Repast S's legacy model integration system; and discusses how the new system can be used to integrate existing file-based models, either agent-based or non-agent-based, into agent models.

**Keywords:** Agent-based modeling and simulation toolkits, model integration, Repast

## INTRODUCTION

Agent-based modeling systems provide a simulation platform in which agents can act and react to one another and to their environment. However, in many agent-based modeling systems, the objects that represent the agent's environment are static or exhibit simple time evolution. For instance, in the basic heatbugs demonstration model, the distribution of temperatures changes via diffusion during the course of the simulation; connecting a model that simulates day and night temperature fluctuations into the heatbug simulation could add greater interest. Another example may be an agent model that simulates commuter travel behavior, where interest may be added by using a dynamic weather model to change the situational context in which the commuter agents were operating. This paper introduces Repast S's legacy model integration system and discusses how modelers can use this new system to integrate existing file-based models, either agent-based or non-agent-based, into agent models.

Repast (ROAD [Repast Organization for Architecture and Design] 2005) is a widely used, free, and open source, agent-based modeling and simulation toolkit with three released version 3 platforms: Repast for Java, Repast for the Microsoft.NET framework, and Repast for Python Scripting. North et al. (2006) discuss the Repast 3 portfolio in depth. The Repast 3

---

* *Corresponding author address*: Michael J. North, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439-4832; e-mail: north@anl.gov.

platform is free, open source, and available directly from the web at http://repast.sourceforge.net (ROAD, 2005). 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) provide overviews of the Repast S runtime and development environments.

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.

Repast S's new simulation development capabilities include direct support for integrating existing (i.e., legacy) file-based models into agent-based simulations. It should be noted that the integration of models requires both a deep understanding of each model to be integrated and careful consideration of the scientific and engineering appropriateness of the interconnections. Obviously, due to scientific and engineering reasons, only certain models are good candidates to be connected to any other given model. However, even after modelers have addressed these important issues and identified good candidates for integration, there is often quite a bit of work to do manually to complete the integration. The Repast S legacy model integration system is designed to reduce the level of effort required to integrate certain classes of models; reduce the chance for error; and assist in the creation of useful documentation on the integration approach.

## RELATED WORK

The direct online integration of computational models can be seen in some ways as a type of collaborative scientific and engineering data exchange in which the data exchange occurs directly from program to program and is fully automated. There are a variety of projects that address various aspects of scientific and engineering data exchange and sharing, including the Extensible Markup Language (XML) Dataset Type and Mapping system (XDTM) (Moreau et al. 2005), the Binary Format Description (BFD) Language (Myers and Chappell 2005), the Data Format Description Language (DFDL) (DFDL-WG 2005), and the External Data Representation Standard (XDR) (Sun Microsystems 1987).

XDTM is a promising and exciting research project that provides powerful description languages for both the logical structure and the physical storage of data. Moreau et al. (2005) introduce XDTM as follows:

> We are concerned with the following problem: How do we allow a community of users to access and process diverse data stored in many different formats? Standard data formats and data access API's [application programming interfaces] can help but are not general solutions because of their assumption of homogeneity. We propose a new approach based on a separation of concerns between logical and physical structure. We use XML Schema as a type system for expressing the logical structure of datasets and define a separate notion of a mapping that combines declarative and procedural elements to describe physical representations. For example, a collection of environmental data might be mapped variously to a set of files, a relational database, or a spreadsheet but can look the same in all three cases to a user or program that accesses the data via its logical structure. This separation of concerns allows us to specify workflows that operate

> over complex datasets with, for example, selector constructs being used to select and initiate computations on sets of dataset elements regardless of whether the sets in question are files in a directory, tables in a database, or columns in a spreadsheet. We present the XDTM design and also the results of application experiments with an XDTM prototype.

As stated above, XDTM includes the ability to work with detailed file formats. However, the current focus with the XDTM effort is on specifying high level workflows and large-scale data structures. Modelers need a more detailed tool for many types of legacy model integration.

Myers and Chappell (2005) describe BFD as an effort "to address both the technical and domain-dynamics issues involved in sharing scientific data." Myers and Chappell (2005) state:

> The problems involved in sharing scientific data files are myriad. They range from low-level issues of programming language and operating system differences in number formats (e.g., "big-endian" versus "little-endian") and the ordering of elements for multi-dimensional arrays to higher level issues involving optional elements (format variations), assumed units (e.g., "feet" versus "meters"), multi-file data sets, and general lack of documentation. These types of problems are clearly not unique to science, but the dynamics of scientific research have limited the success of solutions such as standard data formats and self-documenting files; any up-front effort to make data accessible to potential future users incurs an opportunity cost measurable in terms of research time.
>
> The growing interest in and infrastructure for scientific data mining and informatics approaches that gather and analyze data from across a wide range of techniques and disciplines further complicates the issue, as data may be of interest in multiple communities with conflicting interests.

Myers and Chappell describe BFD as "an XML dialect based on the eXtensible Scientific Interchange Language (XSIL) that supports the executable documentation of 'arbitrary' binary and ASCII [American Standard Code for Information Interchange] data sets" (2005). They state that "applying a BFD template to a set of files produces an XML output containing the original data in an XML-tagged format that can be interpreted by other programs or subjected to further processing (i.e., using XSLT [Extensible Stylesheet Language Transformations])" (2005). This transformational ability makes BFD potentially quite useful for data mapping and conversion between programs. The challenge is that the integration of file-based legacy models into Repast programs requires not only mappings between files but also mappings between files and Java objects.

The DFDL Working Group states that "the aim of this working group is to define an XML-based language, the Data Format Description Language (DFDL), for describing the structure of binary and character-encoded (ASCII/Unicode) files and data streams so that their format, structure, and metadata can be exposed" (2005). DFDL is still in an early stage of development, but it would seem that the challenges with its use in model integration are the same as with BFD.

The XDR Request for Comments (RFC) introduces XDR as follows (Sun Microsystems Network Working Group 1987):

> XDR is a standard for the description and encoding of data. It is useful for transferring data between different computer architectures, and has been used to communicate data between such diverse machines as the Sun Workstation, VAX, IBM-PC, and Cray. XDR fits into the ISO presentation layer, and is roughly analogous in purpose to X.409, ISO Abstract Syntax Notation. The major difference between these two is that XDR uses implicit typing, while X.409 uses explicit typing.
>
> XDR uses a language to describe data formats. The language can only be used only to describe data; it is not a programming language. This language allows one to describe intricate data formats in a concise manner. The alternative of using graphical representations (itself an informal language) quickly becomes incomprehensible when faced with complexity. The XDR language itself is similar to the C language, just as Courier is similar to Mesa. Protocols such as ONC RPC (Remote Procedure Call) and the NFS (Network File System) use XDR to describe the format of their data.
>
> The XDR standard makes the following assumption: that bytes (or octets) are portable, where a byte is defined to be 8 bits of data. A given hardware device should encode the bytes onto the various media in such a way that other hardware devices may decode the bytes without loss of meaning. For example, the Ethernet standard suggests that bytes be encoded in “little-endian” style, or least significant bit first.

XDR clearly provides a detailed description language to define the format of data stored in files. However, it does not provide a mapping between files and Java objects.

The Repast S legacy model integration system provides straightforward, two-way mappings between files and Java objects, which allows data in Repast models, or more generally Java programs, to be stored automatically and then retrieved from either binary or text files. Following is a brief discussion of the overall Repast S design to provide a context for the description of the Repast S legacy model integration system.

## REPAST S

The Repast S design is based on a 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 by using automated tools.

- 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. Repast S includes a powerful way to create POJOs, namely, the Repast Simphony development environment. However, modelers can use any method—from hand coding to wrapping binary legacy models to connecting into enterprise information systems—to create 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) details the Repast S runtime system. In summary, the Repast S runtime design includes:

- Point-and-click model configuration and operation

- Integrated two-dimensional, three-dimensional, geographical information system (GIS), and other model views

- Automated connections to enterprise data sources

- Automated connections to powerful external programs to conduct statistical analysis and visualize model results

## THE LEGACY MODEL INTEGRATION TOOL

The Repast S legacy model integration tool uses XML to define the structure of the files to be written or read. The legacy model integration tool maps between a given file and given Java objects. The tool also provides a harness for creating multiple output files from Java objects; for running one or more legacy models; and then for reading the legacy model output files back into the same or different Java objects.

The Repast S legacy model integration tool can work with Java objects in several ways. The most convenient method is to move file data into and out of JavaBeans. JavaBeans are Java objects that, for each property that is intended to be publicly accessible (e.g., “int x”), implement a “getter” (e.g., “int getX()”) and a “setter” (e.g., setX(int newValueForX)). Using the getter returns the value of the variable of interest, and using the setter assigns a new value. Read-only properties only have getters, while write-only properties only have setters. JavaBeans is a widely used convention in Java programming (Sun Microsystems 2005). See Sun Microsystems (2005) for more information on JavaBeans.

Before using this system, users need to look at the external legacy models (or systems) to which they wish to connect and gather some details about what data these legacy models require. In general, this process will involve going through the following steps:

1. Determining which model or models to which they wish to connect: Is there a pre-existing model that would aid the primary model?

2. Determining how they wish to use the legacy models: Should these models run just once at the start of the simulation or at some other point in the simulation? Or should the model(s) be run repeatedly throughout the simulation?

3. Determining what data those models require: Do they require a settings file and/or other data files? What do these input files look like?

4. Determining what data those models produce: Do these models produce individual output data files or multiple files? What do these output files look like?

5. Determining how the legacy model executes: Is there an executable that can be run? What command line arguments need to be used?

6. Determining where the legacy model's input data will come from: Is there a JavaBean that will hold the data, or is there some other set of data in the primary model that can be used?

7. Determining how the legacy model's output data will be used: Is there a JavaBean or other object in the primary model that needs the generated data?

Users would have to perform many of the above steps when connecting to any system or library in developing a model (or any other system). However, we are addressing the primary complication: How does data from inside a Java environment get converted to a usable form for a potentially external environment? This complication refers to the third and fourth steps above and the simultaneous movement of data (mentioned previously) into and out of JavaBeans.

Once users have identified the files and other data sources that the legacy models require, the primary task is then to determine the format of these files. In many cases, there will not be a formal document specifying the format of these files, nor will there be a straightforward, easy-to-understand set of code for looking at how these files are generated or read. In these cases (and those in which this data is available), the most straightforward manner of determining this format may be simply to look at examples of the files. By checking visually, users can deduce what needs to be in the files, and how it should be formatted. Users may consider:

- Is the data a single parameter or is the data a set of records?

- Is the data of a fixed width, or does it end with some delimiter?

- Is the data a string of text, an integer, a floating point number, or something else?

- How will this data be used best in the primary model? If it is a record, should it be stored as a nested JavaBean in another JavaBean, or would it be better as a matrix of data?

- What data is required and what is optional? If no optional data is there, what will the file look like?

As an example of what this analysis would produce, Figure 1 shows a simple example legacy model data file, and Figure 2 shows the corresponding Repast S legacy model integration XML description file. The example file in Figure 1 is broken up into lines. Each line contains one data item for the legacy model and a comment for clarity. The data item is on the left (e.g., “Person” for the first line) and the comment (e.g., “the agent name” for the first line) begins following the exclamation point. The corresponding XML shown in Figure 2 begins and ends with a file definition (“fileDef”) tag and then contains a list of data (“<data/>“) tags. The attributes of the data tags describe the data item expected at that point in the file as well as the delimiter, if any, that is expected to follow the given data item. Data items are described using a variety of attributes including the “target” (e.g., “itemClass”) which gives the name of the JavaBean property, the “type” (e.g., the item class is a string), and the delimiter that follows (e.g., a space character). Note that a “null” target can be defined to account for file contents (e.g., ends of lines indicated by “\n”) that provide file structure but that do not map to JavaBean data.

The XML description file provides quite a bit of detail on the contents of the corresponding data files. Note, however, what the XML description file does not state:

1. It does not state whether the data file is to be generated from or read into a Java object.

2. It does not state which Java objects are to be used.

3. It does not state whether the data file is in binary or text format.

Each of these questions is answered by the way the user invokes the legacy integration tool. An example in Figure 3 shows how to write data from the “myAgent” JavaBean to a text file named “dataOut.txt” in the current directory using the XML description given in Figure 2. Figure 4 shows code to reverse the process and read the data back into the JavaBean. Binary file access can be achieved simply by changing the specified file type.

```
Person       ! The item to model
10           ! The number of agents
13.5         ! The legacy model run length in hours
A            ! The legacy model interaction type (A, B, or C)
```

**FIGURE 1** A simple example data file

```
<fileDef>
      <data target="itemClass" type="String" delimiter=" "/><data target="null" delimiter="\n"/>
      <data target="numItems" type="int" delimiter=" "/><data target="null" delimiter="\n"/>
      <data target="runLength" type="double" delimiter=" "/><data target="null" delimiter="\n"/>
      <data target="interactType" type="double" delimiter=" "/><data target="null" delimiter="\n"/>
</fileDef>
```

**FIGURE 2** An example Repast S legacy model integration system XML description file that can read and write the file shown in Figure 1

```
// Define the file writer.
DataFileWriter writer = new DataFileWriter(myAgent);

// Set the main properties.
writer.setDescriptorFileName("figure2XMLDescription.xml");
writer.setDestFileName("dataOut.txt");

// Attempt to write out the file.
try {
      // Write the file.
      writer.write();

// Check for errors.
} catch (IOException error) {

      // Note errors.
      error.printStackTrace();

}
```

**FIGURE 3** Code to write JavaBean data to a file using the XML description in Figure 2

Figure 4 shows a file that contains multiple sections, each of which has a repeated block of data. The repeated block of data is marked using an "array" or "record" tag. The array tag is used on tabular data, where each element of the data is of the same type. The record tag's use is very similar to the array tag's, but it is more generic because each element of the record (each column of a row) can be of a different type.

Figures 5 and 6 provide a somewhat more detailed legacy model integration example. In the example of Figure 5, the first set of tabular data represents some set of agents in the model. The first two columns represent the agent's location in a space, and the last column represents the agent's name. Each row would correspond to an agent with X, Y, and Name properties; therefore, the entire table represents a one-dimensional collection of these agents. Accordingly, the integration tool will map each row of data to a unique agent bean. In Java, the entire table represents an agent array (e.g., Agent[]).

In the first table (Figure 1), the data are of different types whereas all the data in the second table (Figure 5) are integers; therefore, it is more appropriate to work with this data as a

```
// Define the file reader.
DataFileReader reader = new DataFileReader(myAgent);

// Set the main properties.
reader.setDescriptorFileName(“figure2XMLDescription.xml”);
reader.setFileToParseName(“dataOut.txt”);

// Attempt to read in the file.
try {
      // Read the file.
      reader.read();

// Check for errors.
} catch (IOException error) {

      // Note errors.
      error.printStackTrace();

}
```

**FIGURE 4** Code to read JavaBean data from a file using the XML description in Figure 2

```
0002
1, 2, Mary
11.9E6, -72.1, Joe
# Event data
9, 17, 19, -83
23, 18, -19, 78
2, 23, 19, 84
```

**FIGURE 5** A second example data file

```
<fileDef>
        <!-- Read in a specific number of agents, the data is of the form #### -->
        <data target=“agentCount” type=“int” length=“4” />
        <record target=“agents” count=“agentCount “>
                <data target=“x” type=“double” delimiter=“,” />
                <data target=“y” type=“double” delimiter=“,” />
                <data target=“name” type=“String” delimiter=“\n” />
        </record>
        <!-- Skip the comment line, using the default delimiter of “\n” -->
        <data target=“null” />
        <!-- Now read data on events that occurred, store them into an array of integers -->
        <array target=“eventData” type=“int” colDelimiter=“,” rowDelimiter=“\n” cols=“4”/>
</fileDef>
```

**FIGURE 6** An example Repast S legacy model integration system XML description file that can read and write the file shown in Figure 5

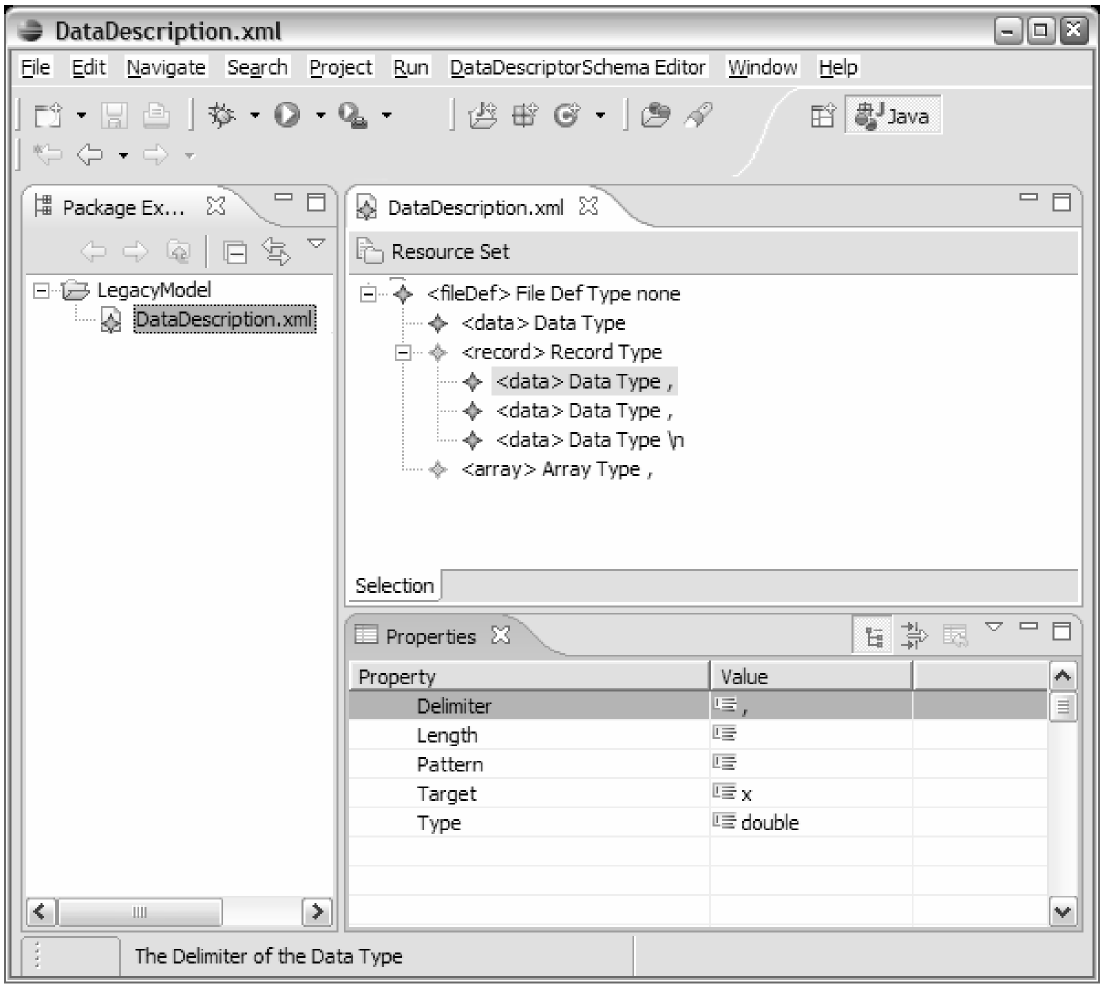
matrix (e.g., int[][]) rather than to work with the data as a collection of beans. For this reason, it makes more sense to use the array tag than the record tag used previously.

This example also illustrates the different ways in which the integration tool can determine the number of rows of data, and the ways in which previously used values can be reused later. The first table is preceded by an integer that represents the number of rows in the agent table. This count is being mapped to the “agentCount” property. Afterwards, this property is used in the agent table’s count property, stating that only two rows of data are in the table. The array table has a similar “rows” property that specifies how many rows of data are in the array; however, as illustrated in this example, that property is optional, and if it is not specified, the number of rows of data is deduced from the file being read in, or the data being written out.

As can be seen from the examples, the Repast S legacy model integration system not only simplifies file-based model integration by replacing detailed Java programming with XML specification, but the resulting XML description files are useful to document the exact contents and formats of legacy model data files.

## 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. As discussed above, Repast S’s new simulation development capabilities include direct support for integrating existing (i.e., legacy) file-based models into agent-based simulations. Obviously, because of scientific and engineering reasons, only certain models are good candidates to be connected to any given model. However, once users have addressed these important issues and identified good candidates for integration, there is often quite a bit of work to do manually to complete the integration. The Repast S legacy model integration system is designed to reduce the level of effort required to integrate certain classes of models; reduce the chance for error; and assist in the creation of useful documentation on the integration approach. Future work includes the development of an Eclipse (2005) extension that provides a point-and-click editor for constructing legacy model integration XML description files (an example of which is presented in Figure 7), as well as the possible development of XSLT style sheets for converting the XML description files into natural language documents. Finally, future work may address expanding the execution system to include simple execution of executables to more complex mechanisms like remote procedure calls.



```
LegacyModel
  DataDescription.xml
```

```
<fileDef> File Def Type none
  <data> Data Type
  <record> Record Type
    <data> Data Type ,
    <data> Data Type ,
    <data> Data Type \n
  <array> Array Type ,
```

| Property | Value |
|---|---|
| Delimiter | , |
| Length | |
| Pattern | |
| Target | x |
| Type | double |

The Delimiter of the Data Type

**FIGURE 7** An example legacy model integration system XML description file editor

## 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. This work is sponsored by the U.S. Department of Energy, Office of Science, under contract W-31-109-Eng-38.

## REFERENCES

Data Format Description Language Working Group, 2005, *Data Format Description Language Working Group (DFDL-WG) Home Page*, Data Format Description Language Working Group, available at http://forge.gridforum.org/projects/dfdl-wg/.

Eclipse Foundation Inc., 2005, *Eclipse Home Page*, Ottawa, ON, Canada; available at http://www.eclipse.org.

Moreau, L., Y. Zhao, I. Foster, J. Voeckler, and M. Wilde, 2005, “XDTM: The XML Data Type and Mapping for Specifying Datasets,” *Advances in Grid Computing Revised Selected Papers from the 2005 European Grid Conference (EGC)*, Springer, Amsterdam, The Netherlands, February 14–16.

Myers, J., and A. Chappell, 2005, *Binary Format Description Language Home Page*, Pacific Northwest National Laboratory, Richland, Washington; available at http://collaboratory.emsl.pnl.gov/sam/bfd/.

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):1–25, ACM, New York, NY, Jan.

ROAD (Repast Organization for Architecture and Design), 2005, *Repast Home Page*, Chicago, IL; available at http://repast.sourceforge.net.

Sun Microsystems Network Working Group, 1987, *RFC 1014 - XDR: External Data Representation Standard*, Sun Microsystems, Inc., Santa Clara, CA, June.

Sun Microsystems, 2005, *JavaBeans Home Page*, Sun Microsystems, Inc., Santa Clara, CA; available at http://java.sun.com/products/javabeans/.

# CONTAINING AGENTS: CONTEXTS, PROJECTIONS, AND AGENTS

T.R. HOWE,* Argonne National Laboratory, Argonne, IL
N.T. COLLIER, Argonne National Laboratory, Argonne, IL, and PantaRei Corp., Cambridge, MA
M.J. NORTH, Argonne National Laboratory, Argonne, IL, and The University of Chicago, Chicago, IL
M. T. PARKER, Argonne National Laboratory, Argonne, IL,
J.R. VOS, Argonne National Laboratory, Argonne, IL, and the University of Illinois at Urbana-Champaign, Urbana, IL

## ABSTRACT

Agent-based models have historically maintained a tight coupling between individuals, their behaviors, and the space in which they interact. As a result, many models have been designed in a way that limits their ability to express behaviors and interactions. In this paper, we propose a new approach toward designing simulations that builds upon the experiences of developing and working with several agent-based toolkits. This approach encourages flexibility and reusability of components and models. A preliminary implementation is the core structure of the upcoming Repast Simphony agent-based modeling and simulation toolkit. By creating a “proto-space” called a Context, we provide model designers with a container that can maintain a localized state for agents. A Context’s state can maintain multiple interaction spaces called Projections, as well as more typical state information. Projections are designed such that they can be used to represent a wide range of abstract spaces, from graphs to grids to realistic geographic spaces. Importantly, projections and agents or individuals are independent of one another. Agents can be agnostic toward the type of projection in which they are interacting, and projections can be agnostic toward the type of agents whose relationships they maintain. Finally, the context provides a logical location to maintain agent behaviors that are dependent on localized agent interactions and environment.

**Keywords:** Agent-based modeling, context, projection, simulation

## INTRODUCTION

Agent-based modeling and simulation (ABMS) is a field that in the past 10 years has seen not only rapid growth in adoption, but also development of several toolkits and frameworks for assisting model designers. Notable open-source implementations include Swarm, Repast (Recursive Porous Agent Simulation Toolkit), MASON (Multi-Agent Simulator of Neighborhoods), and NetLogo (Luke et. al. 2006; ROAD [Repast Organization for Architecture and Design] 2006; SDG 2006; Wilensky 1999). All of these toolkits, except for NetLogo, are geared for model developers who are working in Java or a similar language. They all provide

* *Corresponding author address*: Thomas Howe, 9700 South Cass Avenue, Argonne, IL 60439-4832; email: trhowe@anl.gov.

tools for designing proto-agents. Proto-agents, in this situation, are modeling entities that maintain a set of properties and behaviors but need not exhibit learning behavior, although they may (North and Macal In Press). If proto-agents gain learning behavior, they become agents. This paper uses the term “proto-agent” for generality. These frameworks also provide the tools to develop an environment in which the proto-agents interact. However, none of these toolkits has provided an infrastructure that supports highly modular sets of behaviors and relationships. By factoring proto-agents, relationships, and behaviors into separate components, the Repast team intends to provide tools to create more modular and expressive models. The results form the backbone of the upcoming Repast Simphony (Repast S) framework[1]. The Repast S framework provides many other tools for assisting the model developer in creating a model, but the standard model structure is based on these *Contexts* and *Projections*.

## CONTEXTS

The core data structure in Repast S is called a *Context*. The Context is a simple container based on set semantics. Any type of object can be put into a Context, with the simple caveat that only one instance of any given object can be contained by the Context. From a modeling perspective, the Context represents an abstract population. The objects in a Context are the population of a model. For simplicity, we refer to these objects as proto-agents. However, the Context does not inherently provide any mechanism for interaction between proto-agents. One could say that a Context represents a “soup” where the agents have no concept of space or relation, but the Context is actually more of a proto-space. The Context provides the basic infrastructure to define a population and the interactions of that population without actually providing the implementations. As a proto-space, the Context holds proto-agents that have idealized behaviors, but the behaviors themselves cannot actually be realized until a structure is imposed on them.

Contexts create an abstract environment in which the proto-agents exist at a given point in the simulation. In addition to maintaining the collection of proto-agents, the Context also holds its own internal state. This state can consist of multiple types of data. Simple values, such as time, can be maintained within the Context. These provide proto-agents with information about the world in which they interact. In addition, *Data Fields* can be maintained by the Context. A Data Field is essentially an *n*-dimensional field of values with which the proto-agents in a Context can interact. For example, the classic Heatbugs model provides such a field to represent the heat at a given location in space. These Data Fields can, but are not required to, be directly associated with a physical space. The field is sufficiently generic in that each value is derived simply from a set of coordinates. How that set of coordinates is translated into values is determined by the implementation. Common implementations might include a matrix for storing values or a function that converts coordinates into some value.

In addition to simply maintaining state, a Context can have behaviors associated with it. Generally, these behaviors would affect the internal state of the Context. For example, a context that represents a farming village might have a weather model and a crop model associated with it. These models would represent the behavior of the Context. The weather model would be driven by using the state of the Context, and the output would update the state of Context. One

[1] 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.

could apply more complex or adaptive behaviors to a Context, thus giving the Context an agent-like quality. This provides a rather powerful mode of composition.

Contexts themselves are arranged in a hierarchical structure. An analagous structure, treating model containers ("scapes") as agents, was first introduced by Parker (2000) as a feature of the Ascape modeling environment. This structure had similar composability benefits, including hierarchies of parent containers and associated behaviors. Because scapes had a one-to-one relationship with spaces (i.e., projections), however, the structure was often too rigid for more complex models (i.e., those with agents existing in multiple spaces).

Contexts can contain sub-Contexts. Most contexts can be broken into component parts if necessary. For example, a farming village might consist of multiple families. Each family could be a sub-Context of the larger village. Membership in a Context is inherited. If you are in a sub-Context, by definition, you are a member of the parent Context. Imagine that the parent Context is not a farming village but is instead the world. Each farming village is a sub-Context of the world. The population of the farming village is by definition also part of the population of the world. This hierarchical structure allows for the model designer to consciously define the granularity of the model. Each of the sub-Contexts can have its own state that is separate from the state of its parent.

Membership in a Context is designed to be fluid. As the circumstances surrounding a proto-agent change, the proto-agent can move between Contexts. This shift can happen at a highly granular level, where an action of one proto-agent can cause another proto-agent to shift into a separate Context. Once that shift occurs, the moving proto-agent now interacts with the state and the other proto-agents in its new Context. As a result, proto-agents that are designed to engage in a behavior on the basis of their environment can switch behaviors very easily when they migrate into another context.

While the preceding example described a Context based on location (a farming village vs. the world), Contexts are designed to be as abstract as the modeler desires. A Context could be a family or a business, neither of which is necessarily constrained by space. The key to the Context is that it describes a meta-population and provides the state regarding the circumstances of the proto-agents that exist in the Context.

## PROJECTIONS

Much of the last section emphasized how Contexts describe a proto-space with no specification of relationships between its members. Since the member proto-agents within a Context likely interact across multiple sets of relationships, we deliberately designed them to *not* impose a particular structure on how the proto-agents interact. Projections are data structures designed to define and enforce relationships between proto-agents within a given Context.

Projections take the meta-population as defined in a Context and impose a new structure on it. The structure defines and imposes relationships on the population by using the semantics defined in the Projection. In other words, a proto-agent population is realized once a Projection is applied to it. From a practical perspective, this means that Projections are added to a Context to allow the proto-agents to interact with one another. Projections have a many-to-one relationship with Contexts. Each Context can have an arbitrary number of Projections associated with it, which means that within each Context, the proto-agents can create an arbitrary number of types of relationships with each other. This ability is a significant step away from the methodology of writing proto-agents that are designed to work with grids or networks specifically. Figure 1 shows how Contexts and Projections interact.

The Projection framework is designed to support testing multiple Projections with the same model. As Flache and Hegselmann (2001) show, space does matter when it comes to the behavior of cellular automata (CA). They point out that while the results of CA are robust with regard to grid structure, "…dynamics may be sensitive to local variation in the number of neighbors within a grid." Parker (2000) observed the following:

> Space, at least as we relate to it in finite models, is essentially also an arbitrary idealized concept. To test our models effectively we should be willing to subject them to many different spaces. Hopefully, a tool that enforces the capability to explore these spaces transparently will encourage further exploration.

Similarly, Gulyás (2003) noted that, "most of today's models operate on a fixed, and often arbitrary topology." Gulyás (2003) later states that "experimenting with different topologies would, in practice, mean to re-implement the simulation a number of times, with all the hassles and risks of such an endeavor." For these reasons, when conceiving of a framework for building ABMS, one must at least consider the need to support multiple relationship sets concurrently as well as serially.

Each Projection is designed to work with arbitrary objects. As a result, switching between Projections is simple. No changes are required with respect to the proto-agent code for a

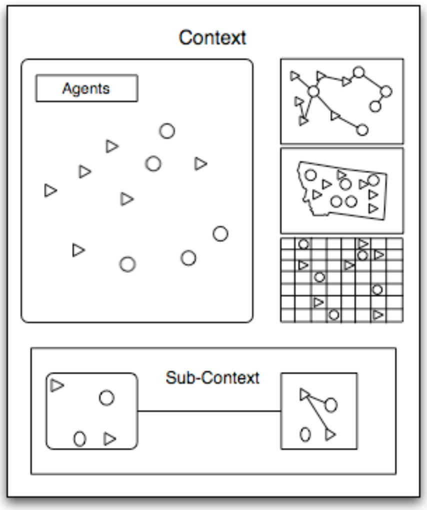


**FIGURE 1** Context with agents (left), projections (e.g., the network, map, and grid on the right), and a sub-Context (bottom)

Projection to be able to work with a proto-agent. In the worlds of Swarm-like simulation toolkits, changing the type of relationships used by proto-agents once required changing the proto-agents to work with the new types of relationships. Projections are designed to be agnostic to the proto-agents to which Projections provide structure. While this means that no code changes are required to allow a Projection to work with a particular proto-agent, it does not completely account for the changes required for the proto-agent to work with the new projection.

In Ascape, Parker (2000) implemented a framework that allowed for polymorphic spaces and for agents to be agnostic about their spatial environment. Interaction topologies could be easily selected, though only at compile time, without modifying agent search methods. Gulyás (2003) described a framework for creating topologies and spaces that were completely swappable. This framework described a container that would maintain relations between proto-agents and objects, which we refer to as *relation sets*. These objects could be other proto-agents or a location in space. As a result, if a proto-agent were trying to maneuver its way through a grid, for example, it would create a new relationship with the location it wanted to occupy. Then, to find its neighbors, it would grab the locations that its neighbors occupied at that time. Finally, it would discover the occupiers of those locations. This approach has the benefit that all relationships in space (or otherwise) are described by using network relationships. Just like in a social network, moving in space would be a matter of creating and breaking network ties.

While appealing, this approach has certain shortcomings. First, when a modeler wants a proto-agent to move in a grid or on a continuous space, Gulyás's approach allows it to use the same mechanism as it would use for changing its network relations. However, the method is somewhat cumbersome, as it does not allow the model developer to use the semantics of the Projection, for example, "move to cell (1, 2)." A framework developer could hide this kind of mechanism in methods that are more intuitive. However, doing so would yield performance penalties both in terms of space and speed. If modelers want to use a simple grid, they are required to create at least two additional objects: one for the location node and one for the relationship between the proto-agent and the location node. Similarly, when a proto-agent needs to query its neighbors, it needs to traverse the network links rather than executing a simple and fast matrix operation. Neither of these issues (i.e., cumbersome usage or performance penalties) is ideal for general-purpose modeling toolkits.

In spite of its shortcomings, Gulyás's framework presents important issues, produced interesting implementations, and influenced the design of Repast S Projections[2]. For instance, Gulyás describes a container that maintains relationships between objects such that the relationships can be accessed in a universal manner. We designed Projections with the same goal in mind. However, in an attempt to keep the Projection model as flexible as possible from the perspective of both the model designer and the Projection designer, we designed Projections so that they did not actually define a general method for retrieving relations or related entities. Instead, that mechanism is delegated to *Query* objects. Query objects are specialized to work with a particular type of Projection. However, all Query objects implement the same interface that returns a list of objects that meet a given criteria from a specified Context.

This approach has several advantages over embedding abstract relation retrieval directly in the Projection. First, it allows Projections to maintain their "natural" application programming

2 As a note, the original Projections prototype was an implementation of Gulyás's design (Howe and Diggory 2003).

interface (API). A grid is, fundamentally, a matrix even though it can be represented as a more general relation container or Projection. Many model designers want to treat a grid as a matrix rather than work through a layer of abstraction that may not be intuitive. This approach also avoids most of the previously discussed performance penalties. Second, creating the Query layer allows the model designer to create queries based on more than simply the relations in the Projection. Queries can be combined to create new Queries of arbitrary complexity. For example, one might want to query a Context by using multiple contained Projections or object attributes. Similarly, by creating a basic set of Queries that can be combined with an expression parser, a model developer could build a system where the relation sets that are retrieved by proto-agents can be specified and altered by using a declarative query language, without needing to change the code.

### Context-Sensitive Behavior

Contexts provide a way to create an environment and maintain a proto-space with a population. Projections create a way to define a wide variety of relation sets with a common method of interfacing those relations (i.e., Queries). However, these alone do not account for proto-agent behavior or how proto-agents will utilize information from the Context and Projections. Thus another key component of Contexts is that they are used to define localized behaviors for proto-agents. Localized behaviors are those in which proto-agents engage only under certain circumstances. Those circumstances are defined by the Context in which the proto-agent is located. For example, on a military base, a soldier may stand at attention as soon as a higher-ranked solider enters the room. However, at home with his or her family, the soldier will likely not engage in the same behavior. This type of localized behavior allows for rich and complex agents. The semantics of how the proto-agent interprets a given Context is currently left up to model designers.

Context-sensitive behaviors can be implemented by creating *watchers* or triggers for the behavior of the proto-agents. The modeler declares the particular circumstances under which a certain behavior is executed. These circumstances represent a “watch” or trigger. Generally, a behavior would be triggered when the state of another object in the simulation changes. Returning to the military example above, the change that would trigger the behavior would be when the room gains a higher-ranked solider. However, the watch would contain another constraint as well. The Context in which the soldier was currently interacting would need to be the “Military Base” Context. This behavior could be altered in different situations by changing the Context included in the watch. For example, the soldier could have another behavior called “Hello” which occurs whenever someone enters the room. If this watch is associated with all non-“Military Base” Contexts, then the soldier will exhibit the appropriate behavior at the right time. By creating sophisticated watch conditions which are specific to Contexts, the model designer can create agents that can change their behavior based on the particularities of the circumstances.

## CONCLUSIONS

To support the continued growth of agent-based modeling and simulation, there is a need to develop new data structures to model both increasingly sophisticated relationship sets and increasingly sophisticated behaviors. To support these needs, we have designed the Context dataset to maintain a meta-population of proto-agents. When a proto-agent exists in a particular Context, it is organized with the other members of the Context into various Projections that define and maintain relationships. The Context can maintain the behaviors of its component proto-agents and provide the support for triggering multiple behaviors from the same stimuli. These Context and Projection structures have been implemented as core modeling features of the upcoming Repast Simphony toolkit release. Hopefully, these structures will provide model developers with a new and rich set of tools to build advanced agent-based simulations.

## ACKNOWLEDGMENT

This work is sponsored by the U.S. Department of Energy, Office of Science, under contract W-31-109-Eng-38.

## REFERENCES

Flache, A., and R. Hegselmann, 2001, "Do Irregular Grids Matter? Relaxing the Spatial Regularity Assumption in Cellular Models of Social Dynamics," *Journal of Artificial Societies and Social Simulation* 4; available at http://jasss.soc.surrey.ac.uk/4/4/6.html.

Howe, T, and M. Diggory, 2003, "A Topological Approach toward Agent Relations," *Proceedings of the Agent 2004 Conference on Challenges in Social Simulations*, October 2–4, 2003, Chicago, IL, ISBN 0-9679168-4-4, published by Argonne National Laboratory, Argonne, IL, page 71.

Gulyás, L., 2003, "Relational Agent Models—A Framework for Modular and Topology-Independent Simulations," SwarmFest, Notre Dame University, Notre Dame, IN.

Luke, S., et. al., 2006, *MASON Home Page*, George Mason University, Fairfax, VA; available at http://cs.gmu.edu/~eclab/projects/mason/.

North, M.N., and C.M., Macal, In Press, *Managing Business Complexity with Agent-Based Modeling and Simulation*, Oxford University Press, New York, NY, USA.

Parker, M., 2000, "Ascape: Abstracting Complexity," *SwarmFest Proceedings 2000*, Utah State University, Logan UT.

ROAD (Repast Organization for Architecture and Design), 2006, *Repast Home Page*, Chicago, IL; available as http://repast.sourceforge.net/.

SDG (Swarm Development Group), 2006, *Swarm Home Page*, Ann Arbor, MI; available at http://www.swarm.org/.

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

# Domain-Support Methods

# INFORMATION RESONANCE

P. S. BARRY, MITRE Corporation, McLean, VA
M. T. K. KOEHLER,* MITRE Corporation, McLean, VA

## ABSTRACT

The DoD is moving to operations based on extensively networked information (netcentricity). Netcentricity implies pervasive information available to all participants in the enterprise. However, to date it is unclear as to what the effect of the availability of this information, and the reliance on this, will be on operations. To explore this we have developed the concept of information resonance. Information resonance posits that there are optimal areas of information flow through the network balanced against the affectivity of the operations. This paper provides a description of the concept of information resonance and demonstrates the approach in an agent-based model. Additional research directions are provided at the end of the paper.

**Keywords:** Agent based, netcentricity, NetLogo

*"Describing the Internet as the Network of Networks
is like calling the Space Shuttle a thing that flies."*
~ John Lester

## INTRODUCTION

With the development of first the internet and now the global information grid new paradigms have had to be developed to understand the effect of information moving about various social networks that overlay the physical, technologic networks. The sheer vastness of these networks belies understanding, yet we seek to use this paradigm as a key enabler for DoD operations. It remains to be determined what the effect of pervasive information (Alberts 2000) will be on operations much less how it will be managed. Without this understanding, the possibility of unintended consequences from an operational perspective is likely to be significant.

The Global Information Grid (GIG) is defined as the globally interconnected, end-to-end set of information capabilities, associated processes, and personnel for collecting, processing, storing, disseminating, and managing information on demand to warfighters, policymakers, and support personnel. The GIG includes all owned and leased communications and computing systems and services, software (including applications), system data, security services, and other associated services necessary to achieve information superiority.[i] The GIG will enable network centric or netcentric operations to become a reality, that is, operations that are based on pervasive access to information.

One can view the flow of information through the GIG as a tool to maximize capabilities that can be used to achieve an effect. Effects are capabilities brought to bear in a situation to change its state. Individual nodes can be viewed as having capabilities. The ability for a given node to exercise its capability in a given situation is dependent upon it having the requisite information. For example, for an evacuation helicopter to provide evacuation capability it must know where the target is as well as the route to the target. This information is provided by other nodes within the network. We assert that for

* *Corresponding author address:* Matthew Koehler, MITRE Corporation, H305, 7515 Colshire Drive, McLean, VA, 22102; *email:* mkoehler@mitre.org.

each node in the network there is an optimal frequency by which to receive information. This optimal frequency provides information at a rate that can be processed with respect to the nodes processing capability as well as the likelihood that a given bit of information might have no utility or negative utility (i.e., be false). In the current context we define a network as a set of nodes joined together by a common purpose. Finally, we assert that the network has a set of optimal frequencies that can be measured as the deviation from the mean of the set of optimal frequencies for all of the nodes. We refer to these optimal frequencies as resonance frequencies, that is, the frequency of network information processing that most greatly enables capability deployment.

This paper will provide a model for information resonance. The next section provides a a framework for understanding information resonance by and minimal definition of a node, its associated information concepts and the network. Subsequent to this we describe the experiment, preliminary results and additional research directions.

## FRAMEWORK

An information network such as that which would be found in the netcentric environment consists of networked nodes which pass information and then take actions conditioned on basic behaviors or a set of goals. The effectivity of the actions at an aggregate level can be measured in a utility calculation. The aggregate amount of information passing around the network can then be graded. We suggest that there are frequencies for information being passed around the network that optimize the effectivity of the network in accomplishing actions. These frequencies are referred to as resonance frequencies.

### Node Description

We define nodes as computational entities or agents that can take actions. All nodes can read messages and send messages. Further, nodes can store information parsed from the messages. This information can be retrieved and used for decision making. Nodes may also have the capability to take an action that can affect its environment. These actions are taken with respect to the node's current state of understanding (defined by the union of the messages in the cache) as well as the goals or directions the nodes may have. Nodes may also have higher inferencing capabilities as well as learning capabilities but these are not discussed here.

More formally, each node has a worldview W that provides the information for taking decisions. At a given point in time the node examines W. If sufficiency conditions are reached the node will take an action; actions include sending messages as well as other more domain specific tasks. At the most basic level, W can be described as a vector of state space variables. However, more sophisticated belief-desire-intention models are certainly possible.

Each node n has a computing capability C that defines its ability to process new information. C is measured in terms of information elements/unit time, where an information element can be described as a message. C can be characterized by $C_{max}$, the maximum number of messages that can be processed per unit time. There is a finite queue $C_q$ that provides for message storage. Upon processing, the message is deleted and the nodes worldview is updated. If the arrival rate exceeds $C_{max}$, we can describe the expected value of the number of dropped messages as E(dropped) = $\beta(C_{max}, C_q,$ arrival rate, ) where $\beta$ is a stochastic distribution.

There are a finite number of orthogonal actions, A, which a node can take. All nodes can send and receive messages, designated as $a_s$ and $a_r$. Contingent upon the node, there are other actions the node can take, for example move or shoot. For any given action $a_i$ there are a set of information criteria $I(a_i)$ that will trigger the node to take the action. A node may also take actions predicated on a time trigger $a_s(t)$, such as periodic updating. There is a maximum rate of actions per unit time $A_{max}$. If the trigger conditions or time triggers exceed $A_{max}$ the actions are either queued or dropped.

## Action Description

An action results in some change of state for the node in a single time step. A task $\tau$ is a set of actions that a node may take over several time steps. A capability P is defined as a set of tasks completed within a set of constraints (e.g., time, money, etc.). Capabilities are accomplished by one or more nodes. An effect e is the measurable result of a set of capabilities where $e \in E$ for all e. The utility of actions taken over time is then $U(E,t)$.

For example, a node may have the capability to move. The action may be to move one step to the north. The task may be to move the node five steps north. The capability may be to move a node north 5 steps in less than 10 time steps. The effect is the relocated node 5 steps north. The utility function may be described as 1 if the node is there in less than 10 time steps and 0 if it is there in more than 10 time steps.

## Resonance

In any real context, messages have heterogeneous information content. A message may be relevant to a given node and its associated decisions or may have little relevance. There is a cost to both storing messages (whether they are relevant or not) as well as determining the information content for a given message. In the ideal each node will only get messages that are relevant to the decisions it can take. However, this would require either preprocessing or some sort of addressing by the sending nodes. In this paper we consider the effect of relatively simple processing nodes that do not have the capacity to excluded messages predicated on *a priori* rules or content searches. Further, as the netcentric environment is dynamic, restricted addressing flies in the face of the concept of the GIG where all information is available to each node. In the model we consider the effect of global addressing for all messages. Nodes have heterogeneous processing capacity resulting in differing optimal message receipt rates. Therefore, the optimal resonance frequency will also be related to the distribution of relevant information with respect to the nodes.

More formally, each node can send messages described as $a_s$. The number of messages N sent for node i is $N(a_{si})$. The frequency of messages is then $N(a_{si})/t$, where t is the time duration of interest. The optimal f is then the frequency that maximizes the utility of the effects created by the node, or max $U(N(a_{si})/t)$. At an aggregate level, we can describe the optimal frequency as max $U(\sum_i \sum_j N_i(a_{sj})/t)$. We suggest that there is an optimal for a given node contingent upon $C_{max}$ and how we describe U.

## Model

To experiment with these concepts we created a simple model in Netlogo 3.1.1 (Willensky 1999) as illustrated in figure 1. In the model there are three types of nodes represented by agents: targets, sensors, and guns. Agents are randomly instantiated on an eighty-one by eighty-one grid torus. Sensors have a limited range within which they can see.

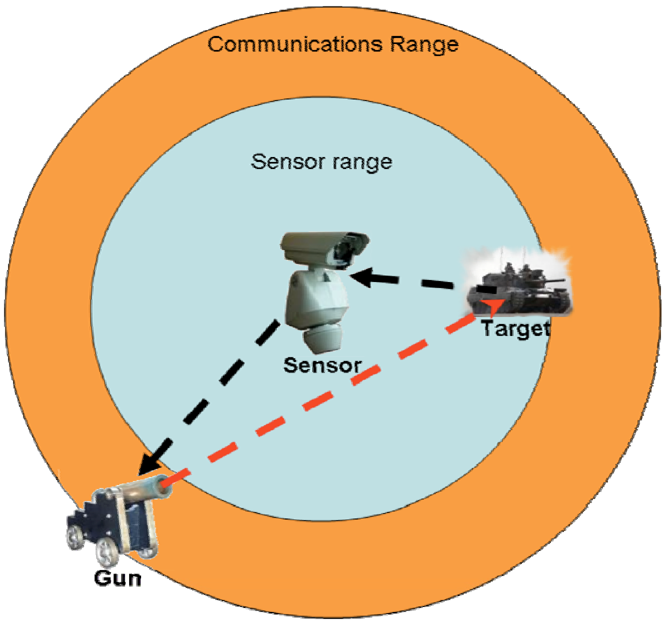


**Figure 1: Model**

Although sensors see everything that comes within their range the distance away the detected entity is will affect weightings assigned to the detected object by the guns. Sensors do not move and send messages to all guns in the network within their communications range. Targets move with a certain probability at each time step and are detected by sensors, with the sensors sending messages to the gun network. Guns receive messages and conduct actions predicated upon the information content of the message, such as moving towards their best target if there are no acceptable targets within their shooting range. Further, there is a specificity associated with guns and targets. Guns can only shoot at targets that have attributes within a certain tolerance of what the gun considers its ideal target. If a gun destroys a target the action contributes positively to the overall utility of the network. Finally, guns have no ability to find targets on their own; their world view is wholly defined by the receipt of messages from the sensors.

At each time step sensors see everything within their sensor radius. However, they are unable to process all of the information they have. Therefore, only a subset of data the sensors have is passed on to guns within the sensor's communications range. The rest of the targeting information that the sensors have is lost. The message includes: what was seen, where it was seen, how far away it was, and when it was seen. Furthermore, at each time step the guns are only able to process a subset of the information that they receive. Guns then use this information to determine which action to take. Guns will shoot at the most acceptable target if it is within weapons range. If the target is not within weapons range the gun will move toward the target. If no acceptable targets are found within the data subset the gun will do nothing. For this model, the only way to contribute to the overall utility of the network is to destroy targets. Finally, targets move with some probability per time step.

The model described herein was based upon Jeff Cares' information age combat model (IACM) (Cares 2004). The Cares IACM incorporates four types of entities: sensors, influencers (e.g. guns), deciders, and targets. It should be stressed that these categories are roles, not specific classes of agent. Therefore, an agent can be a decider and a gun and a target for other influencers. One of the more important issues of the IACM is that it diverges from the typical combat model in that, at its heart is not a Lanchester equation. The IACM focuses on the flow of information and the effect that that flow will have on the performance of the networked entities.

## EXPERIMENT

For this initial exploratory analysis we utilized a full factorial experimental design (summarized in Table 2). Eight parameters were used in the experiment, creating 32 design points. Each design point was run for 500 time steps. Each design point run was replicated 10 times using a standard set of random seeds. This created a total run of 176,000 iterations of the model. Collected data was then analyzed using MINITAB and other packages. Of note is the use of the high/low parameter values; in order to make the design tractable for a large number of runs we restricted the values. As the level of analysis increases, additional techniques will be warranted to deal with the computational expense. For example, the Naval Postgraduate School (SEEDS Lab) is engaged in research applying experimental design to models with massive parameter spaces. Researchers at SEEDS are developing methodologies to sample huge parameter spaces in a way that will highlight regions of interest to be explored subsequently with a higher resolution design (Sanchez 2006, 2005, Lucas 2002).

*Table 1. Summary of the experimental design*

| Parameter | High Value | Low Value |
|---|---|---|
| **Range of the Sensors** | **70** | **10** |
| **Communication Range of the Sensors** | **6** | **2** |
| **Throughput of Sensor Communications** | **50** | **10** |
| **Processing Capabilities of the Guns** | **25** | **5** |
| **Gun-Target Specialization** | **60** | **20** |

Full-factorial design, 10 replicates per design point using a standardized set of random-seeds, each replicate run for 500 time steps.

### Experimental Results

The results of the experiments were evaluated from the perspective of the most efficient execution of the mission. The mission parameters required that the targets be destroyed within 500 time steps. Based on this, an efficiency metric was described for the network which measured the number of targets destroyed divided by the time that was taken. In a number of cases the simulation runs hit steady state; for these situations the time used for the calculations was were the number of targets destroyed per time step hit zero.

Additionally, a metric was developed to describe the amount of activity in the network. This metric, flow/comp, refers to aggregate message activity in the network balanced against the average computational capability for the guns and was calculated by dividing the average number of messages/time step sent by the sensors by the computational capability of the guns. The concept of information resonance would indicate that there are periodic areas of optimal flow/computation as measured by efficiency that does not necessarily correspond to simple maximums.

Figure 2 shows a summary of statistically significant effects in the model. The dominant effect for the simulation is sensor range. In the simulation, sensor range directly determines the number of messages that a sensor will send out. As the number of messages increases the ability of the guns to engage the targets goes up. However, this situation is not unbounded and directly dependent upon the average computational capability of the guns. Within the model there is a clear positive correlation between both the sensor's ability to send messages (a calculated value considering sensor range, sensor bandwidth and sensor comm. range) as well as the gun's computing capability. While interaction effects are statistically significant, they are dominated.

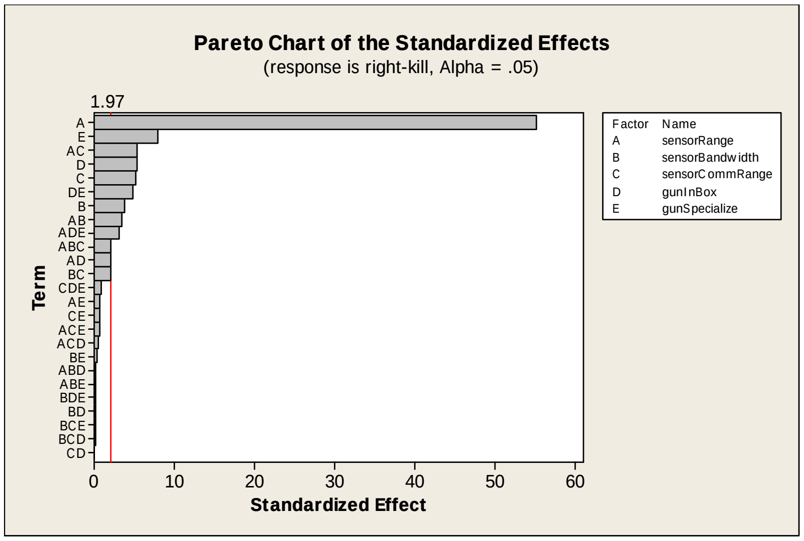


**Figure 2: Model Effects**

The concept of information resonance implies that there are optimal areas of network traffic with respect to computational capability of the network. As described above, resonance can be written as max $U(\sum_i \sum_j N_i(a_{sj})/t)$. In the model utility was measured as the ability to destroy targets quickly, in essence more efficiently. Higher efficiency required a faster ability to leverage the information provided to the network by the sensors. As shown in Figure 3 there is an apparent periodic behavior with clear optima, sub-optima and unacceptable regions. It is of interest that the optimal areas for efficiency appear to be on a rather narrow region. In particular, for this experiment the optimal efficiency region occurred at a flow/compute ratio 1.09, resulting in an efficiency of 3.57. However, moving slightly around this optimal area resulted in significant decreased efficiency (e.g., moving to a flow/compute ratio of 0.97 results an efficiency of 0.13). While this model was very simplified and very limited, the potential implications for actual information networks could be important. The design of information networks within a netcentric environment can potentially result in inefficiencies and an inability to accomplish the designated mission. It also implies that network performance may not be robust the changes in its structure or in the capabilities of the nodes contained within it.

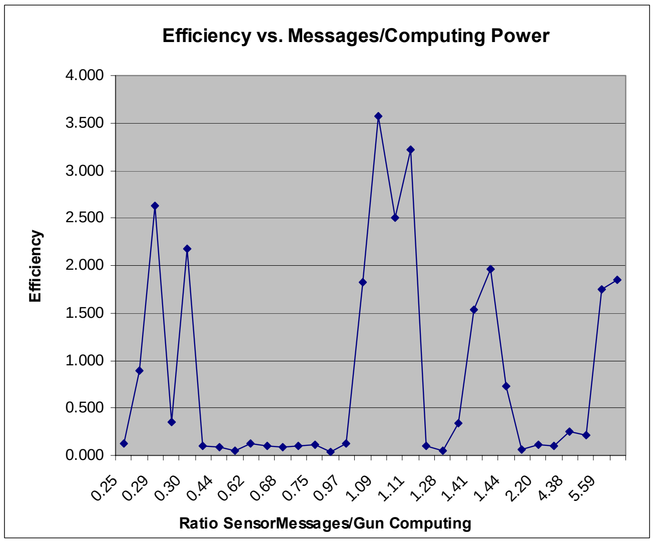


**Figure 3: Efficiency**

The behavior of the simulation indicating a narrow optimal area is further shown in Figure 4 which indicates the time to complete the mission balanced against the sensor messages and gun computing capability. In the figure, less time is better which corresponds to a more efficient use of resources towards completing the mission. The narrow region shown in the lower left hand corner of the figure indicates the shortest time to complete the mission. Small movements in either sensor frequency or gun computing capability quickly lead to non-optimal solutions.

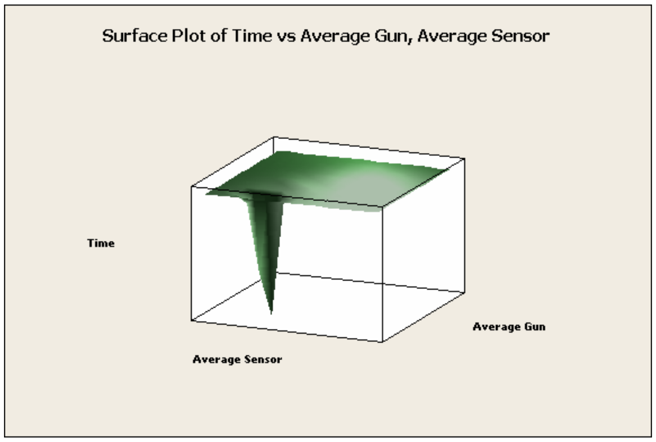


**Figure 4: Time**

The results of the experiments were evaluated against the simplified second order harmonic equations. A rough correlative effect was found to exist around the optimal flow/compute ratio as measured by efficiency, but the correlation was significantly weaker around lower and higher values of the flow/compute ratio. This is not unexpected and supports the idea that in a highly complex network with a large number of stochastic elements and individual actors its dynamics are not readily modeled by closed form solutions at the aggregate level. Additional work is indicated to assess whether a form of the harmonic oscillator is a viable model for the resonance behavior of the network or that another equation is more appropriate.

## RESEARCH DIRECTIONS

The above work represents the first steps of a larger research program. There are a number of issues we wish to explore. First of all, we plan to execute a much higher resolution experimental design. There is enough variation in performance and seemingly enough sensitivity that a higher resolution study is important to gain a reasonable understanding of the model. In addition to increasing the resolution of the experiments there are also planned structural changes. In the current model sensors broadcast their information indiscriminately. The next iteration of the model will incorporate specific network structures between agents to explore the effects of proposed netcentric structure on performance and the effect of alternative network structures. We also intend to explicitly incorporate the Cares IACM structure and purposed performance measures, especially the use of Perron-Frebonius eigenvalues as a method to quantify network effects.

Future work will also investigate the relationship between Cares' measurements of performance and those discussed in the paper. These measures will be used to understand the effect of information flow from the guns back into the network. This change will likely create cycles that should have specific effects on the functioning/performance of the network; as well as, allow us to create a more collaborative action space among the guns. As with the cycles in general, the performance of the gun collaboration will have a large impact on performance of the system as a whole and increase the verisimilitude of the model. Finally, mapping the model into a space with specific distances and explicit time frames will begin to create a framework to examine the performance of specific systems. Investigations such as these will increase our ability to make specific statements about the performance of a netcentric fighting force, issues it may have, and the overall abilities it may (or may not) possess.

## REFERENCES

Alberts, D., J. Garstka, and F. Stein, 2000, Network Centric Warfare, CCRP, Publication Series. Washington, DC.

Cares, J., 2004, "An Information Age Combat Model," *Report for the Director, Net Assessment, Office of the Secretary of Defense.* Alidade Incorporated, Newport, RI.

Lucas, T., S. Sanchez, L. Brown, and W. Vinyard, 2002, "Better Designs for High-Dimensional Explorations of Distillations," *Maneuver Warfare Science 2002,* United States Marine Corps Project Albert, Quantico, VA.

Sanchez, S.M., F. Moeeni, and P. Sanchez, 2006, "So Many Factors, So Little Time...Simulation Experiments in the Frequency Domain," *International Journal of Production Economics* (forthcoming).

Sanchez, S., and P. Sanchez, 2005, "Very Large Fractional Factorials and Central Composite Designs," *ACM Transactions on Modeling and Computer Simulation* 15(4):362-377.

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

[i] Fromhttp://en.wikipedia.org/wiki/Global_Information_Grid.

# EXPLORING CITIES USING AGENT BASED MODELS AND GIS

A.T. CROOKS,[*] University College London, UK

## ABSTRACT

Cities are faced with many problems such as urban sprawl, congestion, and segregation. They are also constantly changing. Computer modeling is becoming an increasingly important tool when examining how cities operate. Agent based models (ABM) allow for the testing of different hypotheses and theories for urban change, thus leading to a greater understanding of how cities work. This paper presents how ABMs can be developed by their integration with Geographical Information System (GIS). To highlight this, a generic ABM is presented. This is then applied to two model applications: a segregation model and a location model. Both models highlight how different theories can be incorporated into the generic model and demonstrate the importance of space in the modeling process.

**Keywords**: Agent Based Models, Repast, GIS, Segregation, Bid Rent

## INTRODUCTION

Cities are complex systems, with many dynamically changing parameters and large numbers of discrete actors. The heterogeneous nature of cities, make it difficult to generalize localized problems from that of city-wide problems. To understand cities' problems such as sprawl, congestion and segregation, we need to adapt a bottom-up approach to urban systems, to research the reasoning on which individual decisions are made. ABMs allow us to simulate the individual actions of diverse agents, measuring the resulting system behavior and outcomes over time; therefore providing a good test bed for developing models of cities. As cities are highly dynamic, both in space and time and secondly, as cities operate on a cross scale basis, propagating through urban systems from interactions between individuals in space to regional scale geographies. For example, it is easier to conceptualize, and model how individual vehicles move around on a road network, where each car follows a simple set of rules. For instance if there's a car close ahead, it slows down, if there's no car ahead, it speeds up and how this can lead to traffic jams without any obvious incident. Rather than producing a series equations that govern the dynamics of vehicles densities. Nonetheless, because vehicle density results from the behavior of individual vehicles interacting, the agent-based approach will also enable the user to study the aggregate properties of the system.

Najlis and North (2004) discuss that there is a growing interest in the integration of GIS and agent-based modeling systems (Brown *et al.*, 2005; Parker, 2004; Torrens and Benenson, 2005; to name but a few). Examples of recent applications include pedestrian dynamics, urban growth models and land use models. For agent-based modelers, this

[*] *Corresponding author address:* Andrew T Crooks, Centre for Advanced Spatial Analysis, University College London. 1-19 Torrington Place, London, UK, WC1E 7HB; email: andrew.crooks@ucl.ac.uk.

integration provides the ability to have agents that are related to actual geographic locations. For GIS users, it provides the ability to model the emergence of phenomena through individual interactions of features on a GIS over time and space (Najlis and North 2004).

In an agent simulation, agents often have some sort of spatial relationship to each other and are situated in an environment, while a GIS can contain multiple layers (e.g. a housing layer, a road network layer, a population layer), whereby each layer is made up of a series of features (e.g. points or polygons, each geo-referenced). This ability to include different features and their attributes of different layers in a GIS, allows for a greater representation of the system of interest when modeling. The combination of layers allows one to model agents of varying types situated within a geographical environment.

The use of vector GIS in ABM, specifically its use of polygons for representation of space is a movement away of from the regular lattice structures used in previous urban models (e.g. Wu 1998). As most features of cities are not regular. The use of GIS allows one to model cities using a variety of different land parcel shapes and sizes. We can deal with objects (such as people or houses) either as fixed or non fixed objects. Both can be coupled through spatial-specific functionality based on objects and their situation in space. Fixed objects are things such as parks which have transition rules and cannot move while non-fixed objects, such as firms or people have transition rules and can move. It is clear that areas do not just change; changes are normally associated with interactions taking place between agents and their environment (O'Sullivan and Torrens, 2000). This interaction allows the underlying urban fabric to change depending on what agents are on top of it. It is clear that fixed and non-fixed objects have close relationships and dependences. Therefore a change in variables of either type will have immediate changes on the other. This change can be detected by geo-referencing the objects and agents simply using x and y coordinates.

This paper presents the development of a conceptual agent based simulation model for the examination of urban issues, focusing on the integration of vector GIS and ABM where space plays a central role in the modeling process. A generic model has been implemented, which allows for global patterns to emerge from the local interactions of individual agents. The generic model was written in Java, an object orientated programming language and extends a number of basic operating classes from the Repast library, an open source agent based modeling environment. Within the model, Repast is primarily used for its display, scheduling, importing GIS vector data (ESRI Shapefiles) along with recording change classes. The program utilizes other Java based GIS libraries especially those from the Java Topology Suite[1] which provide general 2D spatial analysis functions such as line intersection algorithms and buffering, and OpenMap[2] which provides a simple GIS display with panning and zooming and querying of the GIS layers.

[1] Java Topology Suite: http://www.vividsolutions.com/jts/jtshome.htm
[2] OpenMap: http://openmap.bbn.com/

## THE GENERIC MODEL

The initial aim was to develop as much as generic functionality into a basic spatial ABM application, which could then be applied to any situation, utilizing the Repast framework specifically the support of vector GIS integration. The basic model can work on many different geographical scales (e.g. boroughs, wards, output areas) without the need for reconfiguration. The generic functions of the model include the reading in of any vector Shapefile (as most data comes in this format, for example: census and geo-demographic data) and then using this as a base for the display of the model and for data input (Figure 1A). The use of polygons to represent space has the added advantages of giving realism to models along with the ability to calculate of topology, for example adjacency of polygons. The model uses attribute data from the Shapefile to create the correct number of agents, whether these are polygons (which are fixed e.g. land parcels) or points (can be fixed or non-fixed e.g. residents located within the urban environment) as described in Najlis and North (2004) along with setting of the initial starting conditions of the model by reading the attributes from the Shapefile, for example land use or the predominant social class of the area. By allowing individual agents to move around and interact in an environment, clusters can emerge across boundaries which would be missed at a higher, more aggregate level data analysis. The ability to use Shapefiles also restricts the agents' movement to within the study area (Figure 1B). Other generic functionality built into the basic model is the interaction between individual agents and between the agents and their environment.

Moving away from the traditional cellular space leads to conceptual problems such as defining neighborhoods, new searching algorithms, and movement rules. Another problem linked to using vector space is the treatment of physical boundaries (e.g. rivers and motorways) when calculating neighborhoods. Figure 2 highlights how geographical features (such as a river) can be incorporated into the model when calculating neighborhoods. Within Figure 2, the black circle represents the agent of interest. This agent wants to know which agents are within a specified distance of itself and in the same geographical area. A buffer is created at this specified distance. However in this case, the buffer crosses the river. Therefore agents on the other side of the river (yellow squares) are not neighbors as there is no way for them to move directly to the agent; however they are within the buffered region (green line). Those agents (red squares) which are on the same side of the river as the agent and are within the agents defined buffer (red line) are classed as neighbors, any agents outside this area would not be classed as a neighbor. This creation of buffers also has the advantage of calculating local statistics such as density of small areas etc.

The generic model uses Repasts inbuilt functions for the recording of change during the simulation run. These include graphing options, screen capture, the creation of movies, and aggregate information in the form of text files, along with the creation of new Shapefiles (Figure 3). The latter allows for spatial analysis after the model simulation has been run, whilst at the same time keeping track of changes in between time steps within the model, for instance changes in population density within the environment as agents move into and out off an area. This recording of change at a higher, aggregate level as well as at the individual agent level, allows one to see how micro change affects higher level viewing. As the models are based on the concept of a tool to ‘think with’, user interaction is key. User interaction is in the form of parameter settings, the movement of agents via drag and dropping, and the ability to change the agent attributes, all via a graphical user interface which allows for sensitivity testing of model parameters.

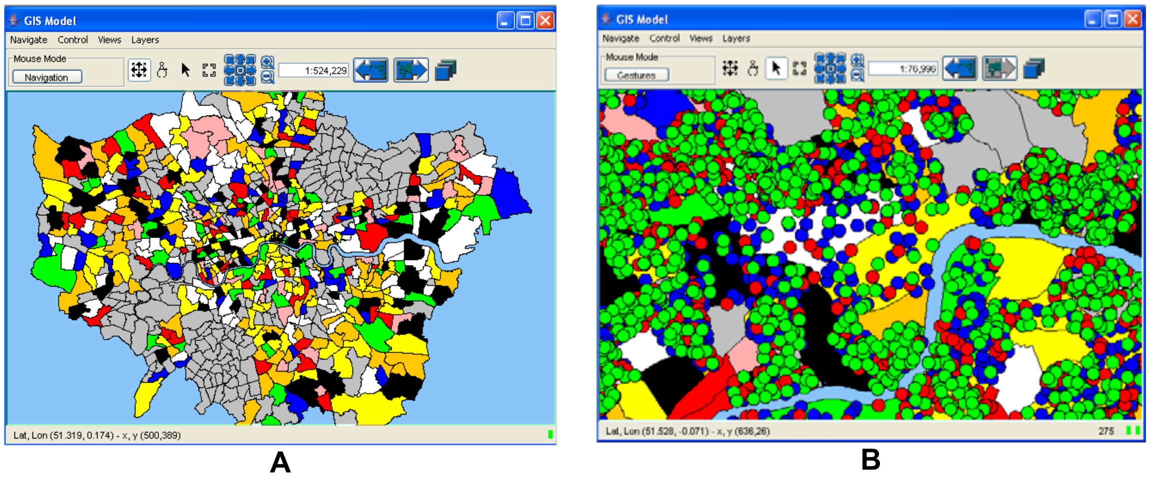

GIS Model
Navigate Control Views Layers
Mouse Mode
Navigation
1:524,229
Lat, Lon (51.319, 0.174) - x, y (500,389)
A
GIS Model
Navigate Control Views Layers
Mouse Mode
Gestures
1:76,996
Lat, Lon (51.528, -0.071) - x, y (638,26)
275
B

**Figure 1.** The use of polygons. **A**: the underlying polygon layer used to create and set the initial model conditions. **B:** a zoomed in section of A, where agents have been created using data from the underlying polygons attributes.

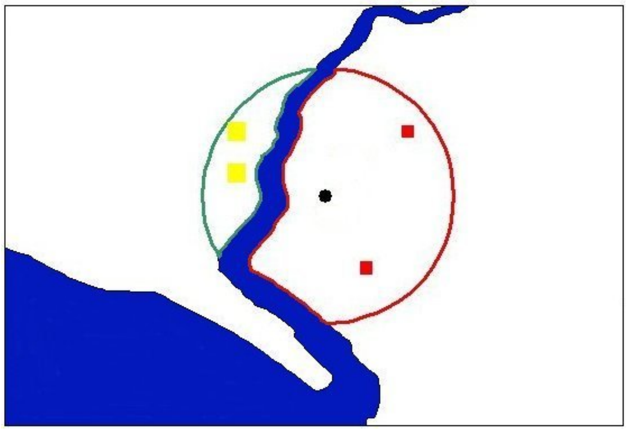


**Figure 2.** Defining Neighborhoods with the inclusion of geographical features

## APPLICATIONS

Using this generic framework, models can be developed rapidly to examine urban issues. Two different model types are presented as ‘proofs of concept’: a traditional segregation model and a bid rent style model (to analyze firm and residential interaction). Both models are made from a series of layers, composed of fixed and non-fixed objects.

All parameters and values used within the models are changeable, opening the model to sensitivity testing making the models exploratory in nature. The models are created with only minor alterations to the basic model structure, highlighting how this approach can be applied to different styles of urban models. Both models highlight how order is possible to emerge from a small number of rules applied locally among many actors which are capable of generating unexpected levels of complexity in aggregate forms.

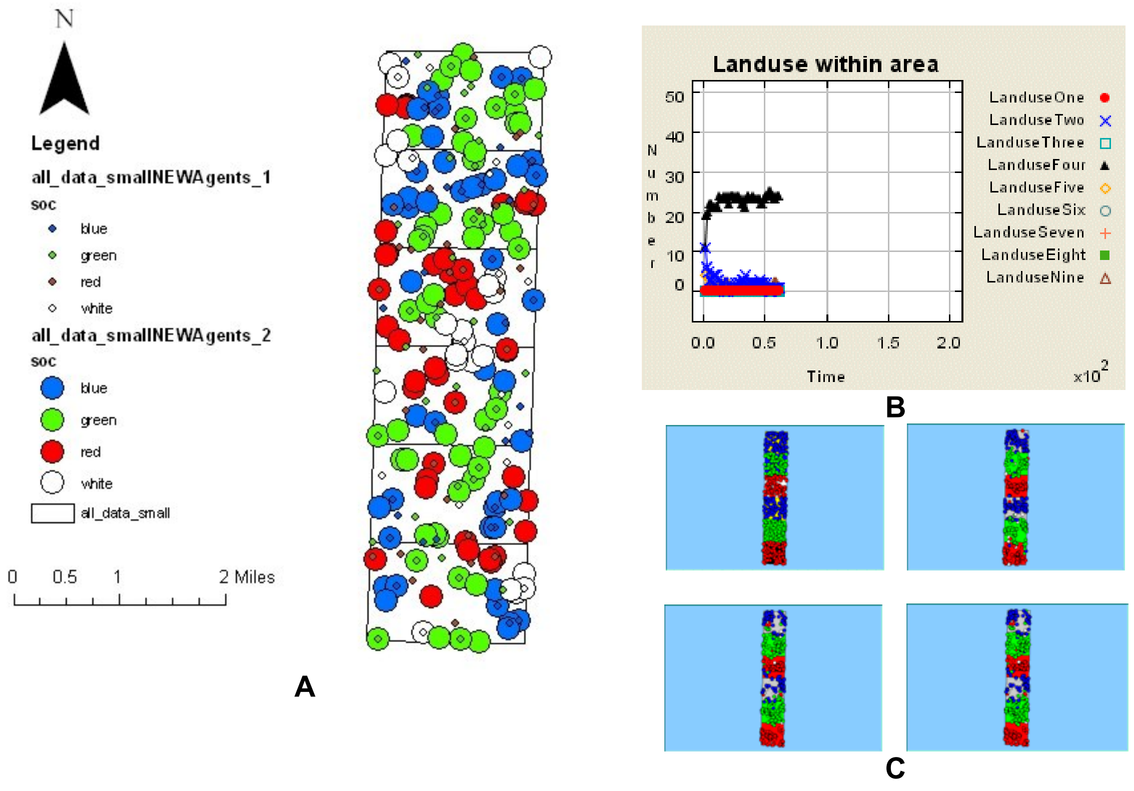


**Figure 3.** Examples of the different types of outputs from the model. **A:** point Shapefiles of locations of agents in two different time steps (step 2 are the larger circles), **B:** a graph showing land use change during a model run, **C:** a series of screen captures during a model run.

## The Segregation Model

There are many different types of segregation, e.g. sex, age, income, language, color, tastes etc. Although the type of segregation may not be clear or be seen at the individual level, it can lead to surprising and unexpected results once modeled. Some segregation is organized, some is economically determined, or results from specialized communication systems, or the interplay of individual choices that discriminate (e.g., choosing to live in certain places). Schelling believed "that the interplay of individual choices, where unorganized segregation is concerned, is a complex system with collective results that bear no close relation to individual intent" (Schelling, 1969, p. 488).

Schelling (1969) presented a model on the emergence of segregation where he showed that with mild preferences to locate amongst like demographic or economic activity groups, strict segregation would emerge unknowingly. This segregation is all too clear when one walks around the urban area, there are clusters of economic groups and residential groups based on ethnicity or social class. Batty *et al.*, (2004) write that it is hard to find clear examples of segregation process taking place, because it only becomes noticeable when it is clearly underway, and by then a detailed chronology becomes impossible to reconstruct. Thus segregation is an important issue especially when related to urban growth which can be studied through simulation modeling.

To highlight this idea of segregation based on simple tastes and preferences, a simple segregation model has been created. Individuals are given the same initial starting conditions but different preferences for their types of neighbors. What is clear is that with different percentages of similarity wanted, different patterns will emerge within the model run. However these patterns change as time passes, as the agents move to find areas in which their preferences are satisfied, thus changing the composition of the neighborhood and the overall appearance of the system. Unlike traditional segregation models, space is not restricted to discrete homogenous cells, more than one agent is allowed per area, and more intelligent movement and searching mechanisms are included (e.g. agents don't just move to random locations or the nearest empty cell). The incorporation of geographical features (such as rivers) within the model also affects the pattern of segregation that emerges. By altering the preferences within the model e.g. incorporating a density function into the model, can stop the agents from clustering all in one area. The segregation model is a good example of local searching based on small scale neighborhoods.

## The Bid Rent Style Model

The aim of this model application is to study human and firm behavior at the micro level in a spatial and dynamic context. Micro and macro interaction are linked together, as it is argued that responses to macro policy should be based on the individual (or groups of individuals) whose actions can lead to order at a higher level without structures being imposed from the outside.

The model is based on the ideas of Alonso's (1964) urban land market theory, which aimed at describing the optimum residential location pattern where land use organization within a city is based on the trade off between many factors (e.g. travel cost, rent and space required). The model presented extends the traditional model to incorporate issues such as time, therefore allowing the system to adapt and evolve to changes in the environment, for example infrastructure investment or population growth. The model also moves away from some of the restrictive assumptions purposed by Alonso, such as a centralized employment. Within the model there are two main groups of agents, the residents and the firms. These are further broken down into subgroups with different incomes and preferences for locations and space. Both of these groups of agents occupy a space and have global searching capabilities. This space contains information on the urban environment such as accessibility, social characteristics of the land and land use which is read directly from the Shapefile.

The ABM approach means households are not treated in the same manner, since each household will have different tastes and preferences, for example for space. The model explores issues such as the exclusion of individuals, and is based on plausible economic decision rules, while at the same time incorporating ideas of distance and price. Locations are determined by a process of bidding (competing) between firms and firms, residents and residents, and firms and residents which introduce many feedback loops into the system along with evolution over time. The model also allows for macro policy to be introduced (e.g. infrastructure investment causes an area to become more accessible). Interactions between individual agents show how local interactions can result in macro patterns emerging.

## CONCLUSION

The paper has presented how a simple conceptual model integrating ABM with GIS functionality can be developed and applied to different types of theoretical urban models, where space plays a central role. Changes within the model can be exported in a number of different formats giving the model extra spatial analysis functionality. Two types of model applications where presented as simple 'proofs of concept' to highlight how ideas and theories can be easily incorporated into the basic model structure that has been developed. Both models are used to represent dynamic processes of change within the urban environment at different geographical scales. Based on individual interactions we show how emergent structures develop, based on a small number of rules applied locally among many actors.

Currently these models have only been applied to abstract systems, which were designed to test how space affects the simulation outcome. Further investigation will apply the models to real world situations and to study these issues in greater detail at the same time testing different scenarios, such as the effects of different infrastructure investment plans in the bid rent style model. The use of GIS vector data removes some of the restrictive assumptions imposed by cellular automata style models (such as one agent per cell, space being arranged on a regular lattice or neighborhoods being composed of blocks of cells), whilst giving access to basic data for setting initial model conditions and parameters.

The integration of ABM and GIS gives modelers the ability to link agents to actual geographic locations. For GIS users, this provides an accessible way to model the emergence of phenomena through individual interactions of features on or related to real geographies over time and space. Combined, this can lead to a greater understanding of how cities work and function.

## REFERENCES

Alonso, W., 1964, *Location and Land Use: Toward a General Theory of Land Rent.* Cambridge, MA: Harvard University Press.

Batty, M., J. Barros, S. Jr. Alves, 2004," Cities: Continuity, Transformation and Emergence," *CASA Working Paper No.72*. Centre for Advanced Spatial

Analysis, University Collage London; Available at http://www.casa.ucl.ac.uk/working_papers/paper72.pdf.

Brown, D. G., R. Riolo, D.T. Robinson, M. North, and W. Rand, 2005, "Spatial Process and Data Models: Toward Integration of Agent-Based Models and GIS," *Journal of Geographical Systems, Special Issue on Space-Time Information Systems* **7**(1): 25-47.

Najlis, R. and M. J. North, 2004, “Repast for GIS,” *Proceedings of Agent 2004: Social Dynamics: Interaction, Reflexivity and Emergence*, University of Chicago and Argonne National Laboratory, IL, USA.

O'Sullivan, D., and P.M., Torrens, 2000, “Cellular models of urban systems,” *CASA Working Paper No.22,* Centre for Advanced Spatial Analysis, University Collage London; Available at: http://www.casa.ucl.ac.uk/cellularmodels.pdf

Parker, D. C., 2004, “Integration of Geographic Information Systems and Agent-based Models of Land Use: Challenges and Prospects,” in Maguire, D., J. M. F., Goodchild, and M., Batty, (Eds), *GIS, Spatial Analysis and Modeling,* Redlands, CA: ESRI Press.

Schelling, T. C., 1969, “Models of Segregation. *The American Economic Review*, **59**:488-493.

Torrens, P. M., and I. Benenson, 2005, "Geographic Automata Systems," *International Journal of Geographic Information Systems* **19**(4):385-412.

Wu, F., 1998, "An Experiment on the Generic Polycentricity of Urban Growth in a Cellular Automatic City", *Environment and Planning B: Planning & Design,* **25**:731–752.

Friday, September 22, 2006

# Computational Social Theory

# Addressing Agent Complexities

# MODELING AGENT EMOTIONS

H.V. PARUNAK,[*] NewVectors LLC, Ann Arbor, MI
R. BISSON, NewVectors LLC, Ann Arbor, MI
S. BRUECKNER, NewVectors LLC, Ann Arbor, MI
R. MATTHEWS, NewVectors LLC, Ann Arbor, MI
J. SAUTER, NewVectors LLC, Ann Arbor, MI

## ABSTRACT

Emotion is an essential human characteristic. In stressful situations such as combat, it is as important as rational analysis in determining behavior. DETT (Disposition, Emotion, Trigger, Tendency) is an environmentally mediated model of emotion that captures essential features of the widely-used OCC model in a computationally tractable framework that can support many agents. We motivate and describe this architecture, and report experiments that use it in simulating combat scenarios.

**Keywords:** Emotion modeling, pheromones, swarm intelligence, BDI

## INTRODUCTION

Most computational models of emotion (Picard 2000) require extensive symbolic reasoning, and are not practical in highly populous real-time domains. DETT (Disposition, Emotion, Trigger, Tendency) captures the essential features of the OCC (Ortony, Clore, Collins) emotion model efficiently, using digital pheromones as the agents' main source of perceptions. It extends current emotional models by defining a Disposition parameter that distinguishes agents' differing susceptibility to various emotions. DETT is designed for situated agents. An agent's emotions are triggered by its percepts, not just by internal reasoning.

We review previous work in computational emotions (particularly in combat modeling), describe the DETT model, and report on some experiments with the model.

## PREVIOUS WORK

### Emotional Modeling

The study of emotion has a rich literature, starting with Aristotle's *Rhetoric*. Different theories identify emotions with outward expressions, physiological responses, distinct behaviors, or cognitive processes, among others. Research in agent-based modeling and human interfaces (including gaming) has increased interest in computational models of emotion (Picard 2000), each drawing on different segments of this tradition.

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

For our purposes, a cognitive perspective on emotions (rather than an expressive or physiological one) is most appropriate, in particular the OCC model (Ortony, Clore et al. 1988), where emotions are "valenced reactions to events, agents or objects." That is, the strength of an emotion depends on stimuli in the agent's environment. Their presence is mapped to a "valence," a positive or negative score, by a process called "appraisal."

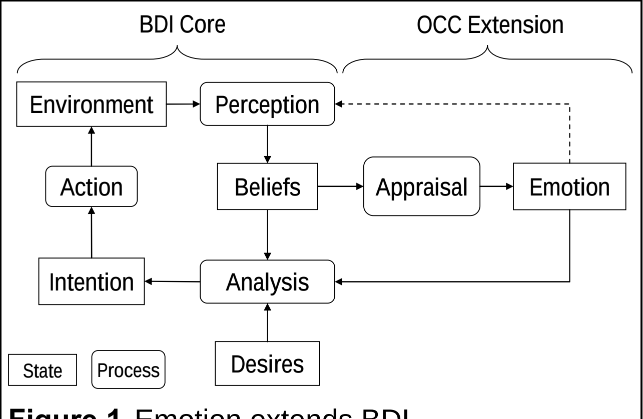


**Figure 1** Emotion extends BDI

Emotion can focus attention, increase the prominence of memories, affect cognitive style and performance, and influence judgments (Brave and Nass 2003). In OCC, "behavior is a response to an emotional state in conjunction with a particular initiating event." We focus on the impact of emotion on an agent's analysis, the process by which it selects intentions from desires. The left side of Figure 1 summarizes the Belief-Desire-Intention (Rao and Georgeff 1991) model. Beliefs (derived from the environment by Perception) and Desires (constant over the time horizon of our model) feed Analysis, which produces Intentions, which in turn drive actions that change the environment. The right side enhances this model with OCC emotion. Beliefs feed not only Analysis, but also Appraisal, which generates Emotions. These Emotions in turn influence Analysis and Perception. (We do not implement the latter link in our current system).

Gratch and Marsella (Gratch and Marsella 2004) offer one of the more mature current computational models of agent emotion. Table 1 summarizes the correspondence between the two models.

## Emotional Combat Models

Explicit emotional modeling is rare in combat models, but elements of two recent models anticipate DETT.

**TABLE 1** Comparison of models

| BDI + OCC | Gratch-Marsella |
|---|---|
| Environment | Environment |
| Perception | Causal Interpretation |
| Beliefs | Causal Interpretation |
| Appraisal | Appraisal |
| Emotion | Affective State |
| Analysis | Coping |
| Desires | ??? |
| Intention | Control Signals |
| Action | Action |

EINSTein (Ilachinski 2004) represents an agent's personality as six scalars in [-1, 1], describing the agent's response to six percepts: the number of alive friendly, alive enemy, injured friendly, and injured enemy troops within the agent's sensor rang, and the agent's distance from its own and its adversary's flag. (Each team seeks to capture the adversary's flag while protecting its own.) A positive weight indicates that the agent is attracted to the entity described by the weight, while a negative weight indicates that it is repelled.

MANA (Lauren and Stephen 2002) extends EINSTein. Friendly and enemy flags are replaced by the waypoints pursued by each side. MANA includes four additional components: low, medium, and high threat enemies, and triggers, events that shift the agent from one personality vector to another. A default state defines the personality vector when no trigger state is active.

The notion of being attracted or repelled by friendly or adversarial forces in various states of health is an important component of what we informally think of as emotion (e.g., fear, compassion, aggression), and the use of the term "personality" in both EINSTein and MANA suggests that the system designers are thinking anthropomorphically, though they do not use "emotion" to describe the effect they are trying to achieve.

EINSTein and MANA implement subsets of Figure 1. The personality corresponds to the agent's Desires. EINSTein's personality vector guides the agent's decisions, but does not change in response to the agent's beliefs about the events, objects, or agents in its environment. Thus it is not a "valenced reaction." EINSTein does not capture OCC emotion. MANA's personality vectors depend on a trigger state, and implement a valenced reaction.

Both EINSTein and MANA do analysis by multiplying a vector of percepts by the personality, yielding a movement vector. In both models, the agent has perfect knowledge within its vision radius. These processes are much simpler than those in (Gratch and Marsella 2004). The differences reflect the systems' objectives. Gratch-Marsella supports a training environment with few agents. Regular human interaction slows the pace, permitting significant computation. EINSTein and MANA manipulate dozens or hundreds of agents in non-interactive simulations, and must minimize execution time.

## THE DETT EMOTION MODEL

DETT supports the DARPA RAID program (Kott 2004), whose objective is to anticipate enemy actions and deceptions to provide real-time support to a tactical commander. Our module (Parunak and Brueckner 2006) evolves agents against observations (Parunak and Brueckner 2006) to learn their characteristics and determine which ones are most likely to reflect future behavior. These agents must execute faster than real time, so they cannot conduct complex symbolic reasoning, but use numerical computation via DETT. We also use the model in another project to automate the reactions of non-combatants with combatants. This capability requires us to recognize that non-combatants will have a range of personality types and to incorporate these differences in their behavior. Details of our computations are in (Parunak and Brueckner 2006).

Both applications require an efficient way to include emotions in modeling combat. This reasoning takes place at two locations in Figure 1: Appraisal and Analysis.

### Appraisal

MANA's triggered personality vectors are a useful model for appraisal, but have two limitations. First, MANA defines vectors and triggers for the squad, all of whose members share the same values. Individual combatants (e.g., new recruits vs. seasoned veterans) differ in their susceptibility to different emotions. To use evolution to learn the characteristics of entities, we must parameterize this difference. Second, MANA assumes that emotions arise and disappear immediately with the appropriate trigger. Empirically, emotions rise rapidly but not instantaneously, and persist after the trigger is removed.

To address the first concern we add a new component, Dispositions, to the model (Figure 2). Each Emotion has a corresponding Disposition. Like Desires, Dispositions are constant over a run. A Disposition modulates Appraisal to determine how much a Belief triggers the corresponding Emotion. The Emotion then modulates Analysis to impose a Tendency on the resulting Intention. The main elements of this model are thus the Disposition, Emotion, Trigger, and Tendency (the effect on intentions) (DETT). Table 2 gives two examples.

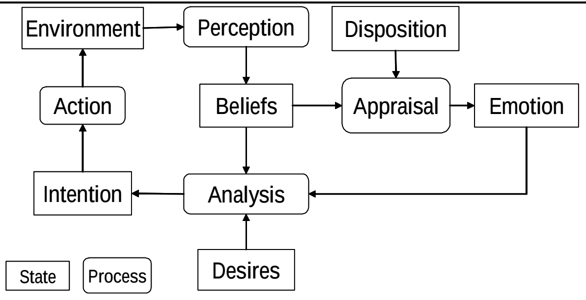


**FIGURE 2** Incorporating disposition in BDI + OCC

Agents sense one another through digital pheromones (Brueckner 2000), labeled scalars that they deposit in the environment and that diffuse and evaporate over time. An agent believes what it senses in the form of pheromones. Our pheromone vocabulary for RAID includes flavors for live and dead friendly, hostile, and neutral individuals, weapons fire, key sites, and topographical features that provide cover from fire or that enhance mobility.

An agent's desires are values in [-1, +1]: ProtectRed (the adversary), ProtectBlue (friendly forces), ProtectGreen (civilians), ProtectKeySites, AvoidCombat, AvoidDetection, and Survive. Negative values reverse the sense: a negative ProtectRed is a desire to harm Red. Table 2 in (Parunak, Brueckner et al. 2005) shows which pheromones attract or repel an agent with a given desire, and how that tendency translates into action. For example, an agent with a high positive ProtectRed will be attracted to RED-ALIVE, RED-CASUALTY, and MOBILITY pheromone, and will move at maximum speed.

Let $\boldsymbol{P}$ be the vector of pheromone strengths at an agent's location. The agent's Disposition is a matrix $\boldsymbol{D}$. $D[i,j] \in [0,1]$ is the relevance of the $i$th pheromone to the $j$th emotion. The $j$th emotion depends (nonlinearly) on the $j$th element of $\boldsymbol{P}^T\boldsymbol{D}$.

A digital hormone (Parunak, Brueckner et al. 2005), one per agent per Emotion, models change in emotion with time. $\boldsymbol{P}^T\boldsymbol{D}$ determines the current deposit to the hormone vector $E$, so the longer an agent is exposed to a trigger pheromone, the higher the level of the associated emotions. When the trigger is removed, its emotions decay exponentially. The higher the disposition, the more quickly the emotion grows in the presence of a trigger.

## Analysis

Analysis draws on the same pheromone vector $\boldsymbol{P}$ of beliefs as does Appraisal, and takes

**TABLE 2** Sample DETT semantics

| Disposition | Emotion | Trigger | Tendency |
|---|---|---|---|
| Cowardice | Fear | Presence of armed enemy<br>Incoming attack | Less attention to orders<br>Tend to move away from threat |
| Irritability | Anger | Presence of enemy | More likely to engage in combat<br>Tend to move toward threat |

as input the current state of the emotion vector ***E***. In addition, it considers the values of the agent's vector of Desires or Wants ***W.*** The desires we are modeling are Protect Red, Protect Blue, Protect Green, Protect Key Sites, Avoid Combat, Avoid Detection, and Survive. Each has a real value in the range [-1,1], where a negative value indicates that the agent wants the opposite state of affairs described by the desire. A movement matrix ***M*** indicates whether a given Desire tends to attract or repel the agent toward a given flavor of pheromone: ***M***[*i*,*j*] is 1 if desire *j* is attracted to pheromone *i*, -1 if it is repelled, and 0 if the pheromone is irrelevant to the desire.

In the absence of emotions, the agent's behavior is a nonlinear function of $\boldsymbol{P}^{\mathrm{T}}\boldsymbol{MW}$. Emotions modulate these behaviors. Elevated Anger increases movement likelihood, weapon firing likelihood, and tendency toward an exposed posture, while elevated Fear decreases these likelihoods. Level of a particular emotion models the extent to which the emotion modulates the agent's behavior. Someone who experiences high fear, but continues to behave as if he were not afraid, would be modeled as having low fear. We are not trying to model emotion as experienced by an agent, only as perceived by its impact on the agent's behavior.

## EXPERIMENTAL RESULTS

We have tested the DETT model in wargames where humans make decisions that are played out in a simulator. The commander for each side (Red and Blue) has a team of human operators who set waypoints for individual units in the simulator. Each unit corresponds to a fire team. Each operator manages four to six units. The simulator moves the units, determines firing actions, and resolves the outcome of conflicts.

Our system fits the model to observed behavior of units, using evolution in a faster-than-real-time simulation of the battle (Parunak, Brueckner et al. 2006). To test our ability to fit personalities based on behavior, one Red operator responsible for four units is designated the "emotional" operator. He selects two of his units to be cowardly ("chickens") and two to be irritable ("Rambos"). He does not disclose this assignment during the run. He moves each unit according to the commander's orders until the unit encounters circumstances that would trigger the emotion associated with the unit's disposition. Then he manipulates chickens as though they are fearful (avoiding combat and moving away from Blue), and moves Rambos into combat as quickly as possible.

The difference between the two disposition values (Cowardice – Irritability) of the fittest ghosts proves a better indicator of the emotional state of the corresponding entity than either value by itself. To characterize a unit, we maintain a 800-second exponentially weighted moving average of the Delta Disposition, and declare the unit to be a Chicken or Rambo if this value passes a negative or positive threshold, respectively (currently, ±0.25).

In one series of experiments, we successfully identified 68% of the chickens played. The detection rate for Rambos was much lower (5%), because the brave die young and our algorithm does not have enough exposure to a brave unit's behavior to diagnose its emotional state. But we never called a Rambo a Chicken. In the one case where we called a Chicken a Rambo, logs show that in fact the unit was being played aggressively, rushing toward oncoming Blue forces.

Figure 3 compares our emotion detector with a human observer on a separate series of experiments. We detected cowards (= chickens) much earlier than the human, while missing only one chicken that the human detected.

In addition to these results on units intentionally played as emotional, we sometimes detect other units as cowardly or brave. Analysis of the behavior of these units shows that these characterizations were appropriate: units that flee in the face of enemy forces or weapons fire are detected as Chickens, while those that stand their ground or rush the adversary are denominated as Rambos.

We did not detect some units that were played as cowardly. Many of these non-identified cowards were red units that were far from a blue unit. This discrepancy arises from an instructive difference between our software and the emotional operator, which illustrates the situated nature of the DETT model.

In our software, an agent's knowledge of its environment is conveyed entirely through a digital pheromone field. If a red unit is beyond the propagation limit of the digital pheromone representing the blue unit, the red unit does not know of the existence of the blue unit. (The propagation limit on the pheromone is analogous to a limitation on a soldier's field of vision in the real world.) Thus even if the red unit has a cowardly disposition, it will not develop fear and will not behave in a fearful way.

The operator looks down on a map of the overall battlespace, and can see all of the units at once. Confronted with managing several units concurrently in the midst of an active battle, the operator can easily overlook the fact that though he can see both a red unit and a blue unit, the red unit might *not* be able to see the blue unit at a given moment. He knows that a fearful red should flee from blue. He can see both the red and the blue. So he moves the red away from the blue.

In the DETT model, emotions become active only when triggered. The inconsistency between what is played and what is detected is in what the cowardly agent knows about its environment. The operator imputes his knowledge of blue to the red unit, so from his perspective its behavior reflects fear. In the software, the red agent does not see the blue unit, and so does not sense fear or act in a fearful manner. This example makes clear that emotion is very much a situated concept. It cannot be detected by movement away from a threat, only by movement away from a threat of which the agent is aware. An emotion such as fear may well have triggers that we have not modeled, and our current approach would not detect it. The problem is circular in structure: we cannot recognize a behavior as evidence of fear unless we can associate it with a trigger, and we cannot learn that an environmental feature is a trigger unless we can detect that it causes fear. Breaking this closed loop is an interesting and challenging research question.[1]

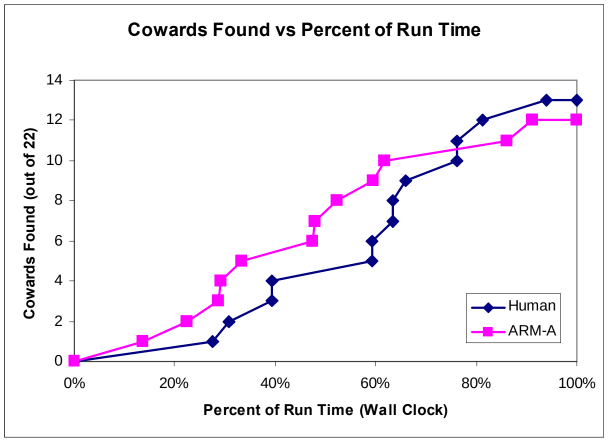


**FIGURE 3** BEE vs. human

## CONCLUSION

Emotion is a critical component of modeling agent behavior, particularly in

[1] We are grateful to a participant in the DAMAS 2004 workshop for a line of questions that motivates this discussion.

stressful environments such as combat. The Gratch-Marsella model offers a sophisticated implementation of current psychological theories of emotion, but is computationally too expensive to apply to large populations of combatant agents. Some fine-grained agent-based models embed a notion of personality (EINSTein and MANA), but do not recognize the important distinctions between individual combatants.

The DETT model (Dispositions, Emotions, Triggers, Tendencies) combines the theoretical richness of the Gratch-Marsella model with the computational efficiency of EINSTein and MANA. It also extends current emotional models with the notion of disposition, accounting for differences in the emotional susceptibility of various agents by reasoning about the reaction of an internal agent characteristic to the external environment in which the agent is situated. We are using the model in two different contexts, and have demonstrated the basic computational cycle in implemented software (and in our second application, in actual wargame experiments).

The model is still an approximation. It does not implement the known effect of emotion on perception, and does not consider other possible linkages (e.g., between emotion and desire). Such simplifications are in the nature of simulation, and are justified empirically by the notion of “universality”: the dynamics of a multi-agent simulation often depend more on the interactions of the agents than on the details of individual agents’ reasoning (Parunak, Brueckner et al. 2004).

## ACKNOWLEDGMENTS

This material is based in part upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. NBCHC040153. 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 DARPA or the Department of Interior-National Business Center (DOI-NBC). Distribution Statement “A” (Approved for Public Release, Distribution Unlimited).

This study was supported in part by the TRADOC Analysis Center, Naval Post Graduate School, Monterey under Contract No. GS-35F-4912H, Order No. GST0904BH6603. The views and conclusions in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the TRADOC Analysis Center the Naval Post Graduate School, the Department of Defense, or the US Government.

## REFERENCES

Brave, S. and C. Nass, 2003, Emotion in Human-Computer Interaction. *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications*. J. A. Jacko and A. Sears. Mahwah, NJ, Lawrence Erlbaum Associates, Inc.**:** 81-96.

Brueckner, S., 2000, Return from the Ant: Synthetic Ecosystems for Manufacturing Control. Thesis, Department of Computer Science, Humboldt University Berlin, Berlin, Germany, http://dochost.rz.hu-berlin.de/dissertationen/brueckner-sven-2000-06-21/PDF/Brueckner.pdf.

Gratch, J. and S. Marsella, 2004, "A Domain-independent Framework for Modeling Emotion." *Journal of Cognitive Systems Research* **5**(4): 269-306.

Ilachinski, A., 2004, *Artificial War: Multiagent-based Simulation of Combat*. Singapore, World Scientific.

Kott, A., 2004, "Real-Time Adversarial Intelligence & Decision Making (RAID)." Retrieved 20 January, 2005, from http://dtsn.darpa.mil/ixo/programdetail.asp?progid=57.

Lauren, M. K. and R. T. Stephen, 2002, "Map-Aware Non-uniform Automata (MANA)—A New Zealand Approach to Scenario Modelling." *Journal of Battlefield Technology* **5**(1 (March)): 27ff. http://www.argospress.com/jbt/Volume5/5-1-4.htm.

Ortony, A., G. L. Clore, et al., 1988, *The cognitive structure of emotions*. Cambridge, UK, Cambridge University Press.

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.

Parunak, H. V. D., S. Brueckner, et al., 2005, Characterizing and Predicting Agents via Multi-Agent Evolution. Ann Arbor, MI, Altarum Institute.

Parunak, H. V. D., S. Brueckner, et al., 2006, *Real-Time Evolutionary Agent Characterization and Prediction*. Social Agents: Results and Prospects (Agent 2006), Chicago, IL, Argonne National Laboratory.

Parunak, H. V. D., S. Brueckner, et al., 2004, *Universality in Multi-Agent Systems*. Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2004), New York, NY, ACM. 930-937. http://www.newvectors.net/staff/parunakv/AAMAS04Universality.pdf.

Parunak, H. V. D. and S. A. Brueckner, 2006, Extrapolation of the Opponent's Past Behaviors. *Adversarial Reasoning: Computational Approaches to Reading the Opponent's Mind*. A. Kott and W. McEneany. Boca Raton, FL, Chapman and Hall/CRC Press.

Parunak, H. V. D., S. A. Brueckner, et al., 2005, "Pheromone Learning for Self-Organizing Agents." *IEEE SMC* **35**(3 (May)): 316-326. http://www.newvectors.net/staff/parunakv/ParunakIEEE.pdf.

Picard, R. W., 2000, *Affective Computing*. Cambridge, MA, MIT Press.

Rao, A. S. and M. P. Georgeff, 1991, *Modeling Rational Agents within a BDI Architecture*. International Conference on Principles of Knowledge Representation and Reasoning (KR-91), Morgan Kaufman. 473-484.

# THE SEMANTICS OF EMERGENT URBAN CENTERS

R.G. Reynolds,* Wayne State University, Detroit, Michigan
Patrick Franzel, Wayne State University, Detroit, Michigan

## ABSTRACT

Numerous models of modern and ancient urban landscapes have been proposed. While it is of interest to classify examples of early urban centers, it is even more interesting to model their origins. Since these emergent centers can be viewed not only as adaptations to their social and biological environments, but also as a source of further change. Thus, the meaning or semantics of an emergent center reflects the processes by which it was formed. In this paper, we propose a framework in which to extract the semantics of an archaic urban site from the archaeological data. An example using the site of Monte Alban is presented.

**Keywords:** Cultural change, evolution of urban centers, early Mesoamerican cities, agent-based modeling.

## INTRODUCTION

Wirth states that, "Each city, like every other object in nature, is, in a sense, unique" (Marcus, 1996). However, according to Blanton, "culture change in the direction of increased scale and complexity can occur in varied ways. I suggest that the cultural ecologists should do as others have and view this variety as a source of stimulation for theory-building". (Blanton 1980:148). In this paper we investigate the potential for building agent-based models of ancient urban centers by applying a suite of tools from Artificial Intelligence and Data Mining to existing archaeological data from a given site.

While there are numerous models of different urban structures, such as the single and multiple nuclei models (Marcus1996), one key question is the processes by which the structure is generated based upon the basic interactions of the decision agents. In other words, what decision rules for agents are necessary to capture the processes the produce each of these classes of models? Are certain types of decision-making rules necessary to attain a particular type of organizational structure in a given environment? Can we identify what decision-making rules have more of an impact on site structure? What changes in decision-making rules are required to produce changes from structural type to another?

While we expect our framework to apply to developing urban centers in general, we take one early Mesoamerican urban center, Monte Alban, as an example. We start by applying several different data mining techniques to a data set consisting of over 2000 terraces that make up the Monte Alban occupation. Each terrace is described in terms of hundreds of cultural and environmental variables. Each part of the site is designated as a terrace because the site is situated on a hill and occupational terraces are carved into the side of the hill.

In the following we describe how we determined which terraces were occupied in a given period and what variables we will use to describe the properties of selected terraces.. Next we apply several different data mining techniques to extract rules that can be used to predict which terraces are occupied in a given time period, along with what new terraces are added.

## THE MONTE ALBAN EXAMPLE

While the basic occupational phases of the valley extend from the early village formation in Tierras Largas to Monte Alban V, the urban site of Monte Alban emerged in period 1a and continued to be occupied through Monte Alban V. So in generating our data mining rules we will begin with Period 1a and ignore the previous phases.

**TABLE 1** The basic Occupational Phases of the Valley

| Period | Approximate Date |
|---|---|
| Tierras Largas | 1400 - 1150 BC |
| San Jose | 1150 - 850 BC |
| Guadalupe | 850 - 700 BC |
| Rosario | 700 - 500 BC |
| Monte Alban Ia | 500 - 300 BC |
| Monte Alban Ic | 300 - 150/100 BC |
| Monte Alban II | 150/100 BC - AD 200 |
| Monte Alban IIIa | AD 200 – 500 |
| Monte Alban IIIb | AD 500 - 700/750 |
| Monte Alban IV | AD 700/750 |
| Monte Alban V | AD 1000 – 1521 |

Table 1 gives all of the relevant periods of social evolution in the valley. Tierras Largas marks the beginning of early village settlement there. The state emerged at Monte Alban in period Monte Alban 1a. The valley came under control of the state by Monte Alban II, and Monte Alban IIIa signaled the decline of the state and its succession by a collection of city-states localized in different parts of the valley. The phases as described there represent uneven slices through time.

To illustrate our approach we will develop a set of rules to determine whether or not a terrace was likely to be occupied in each of its established periods (Ia, Ic, II, IIIa, IIIb-IV, and V). These rules will come from a dataset or 2073 sites that were surveyed by Blanton et. al (). While many environmental and cultural variables were collected for each terrace at the site, only the environmental variables that describe terrace location will be examined here. The question to be answered here is what locational features of the environment were preferred for settlement at each time period? And, there consistencies in these rules from period to period?

The first thing that must be done is to determine which of the terraces in the database is occupied in each of the time periods. There are certain ceramics that are indicative of a specific time period based upon their stylistic attributes. These stylistic categories were developed by Caso, Bernal, and Acosta and called CBA. For a given category, when the piece is broken it can produce pieces of different categories, each of which is related to the original CBA. That is, for a given style there can be rim pieces, base pieces, handles, etc. The ceramics in the table below are the ones that indicate Period Ia. The CBA class is given in column 1, the minimum number of pieces needed to represent the presence of a particular stylistic type is given in column 2. Column 3 gives the categories associated with pieces of that pottery type. The fourth column eclues categories whose pieces may overlap with, or look like that of another CBA category.

**TABLE 2** Ceramic categories that are used to indicate a Period 1a occupation.

| **CBA designation** | **Count Special Requirements** | **Included Categories** | **Excluded Categories** |
|---|---|---|---|
| C-2 | Only Count if There are 4 or More Pieces | *0008*, 0022, 0031, 0032, 0038, *0056*, 0121, 0122, 0123, *0381*, *0382*, 0383, 0384, 0385, 0386, 0561 | None |
| C-4 | None | 0016, *0018*, 0387, 0389, 0390, 0391, 0393, 0394, 0395, *0396* | None |
| K-3 | Only Count if There are 2 or More Pieces | 2010, 2042, *2064*, 2065, 2072, 2076, *2077*, 2080, *2411* | None |
| K-8 | None | 2079 | 2052, 2078, 2085 |
| G-15 | None | *1319*, *1333*, *1336*, *1337*, 1342, 1343, 1345, 1346, 1347, 1348, 1357, *1358*, 1361, 1362, 1363, 1364, 1365, 1367, 1369, 1370, *1373* | None |
| G-16 | None | 1332, 1339, 1340, 1342, 1343, 1344, 1347, 1357, *1358*, 1364, 1365, 1366, 1368, 1369, 1370, *1373* | None |
| G-17 | None | *1331*, 1332, *1334*, 1340, 1366, *1372* | 1338 |

Ceramic categories in *italics* are not included in the database.
Ceramic categories underlined are listed multiple times and should only be counted once.

The following ceramic categories for the rest of the time periods considered here are given in table 3

**TABLE 3** Ceramic indicators for the remaining periods.

| Time Period | Indicator Ceramic Categories |
|---|---|
| Ic | 1297, 1338, 1353, 1355 |
| II | 0001, 0002, 0003, 0004, 0005, 0006, 0021, *0023*, 0407, 1194, 1419, 1420, *2061*, *2416*, *2417*, 3408, *3409* |
| IIIa | 1264, 1265, 1312, 1421, 3410, 3411 |
| IIIb-IV | 1120, 1122, 1123, 1125, 1126, 1137, 1138, 1140, 1259, 1263, 1274, 1277, 1422, 2418 |
| V | 1102, 1104, 1105, 1106, 1107, 1109, 5007, 5329 |

Ceramic categories in *italics* are not included in the database.

There are 566 sites with at least one piece of ceramic from the diagnostic categories that indicate the Ia Period. 390 of those sites have two or more, and 304 of them have 3 or more. I think it is safe to say that one piece of ceramic could easily have been deposited on a site through some means other than occupation. As the number of pieces found on a site increases though, the likelihood that they were accidentally deposited decreases.

It could also be argued that two pieces of ceramic could easily be accidentally deposited on a site. However, if we only consider the terraces with three or more pieces, then we will exclude almost half the terraces that could be considered occupied during the Ia Period. Therefore, we decided to classify any site with two or more Ia Period indicating ceramic pieces as being occupied at that time and all the rest as unoccupied. The number of occupied terraces in each of the periods under study is given in table 4.

**TABLE 4** The number of occupied terraces in each phase.

| Time Period | Ia | Ic | II | IIIa | IIIb-IV | V |
|---|---|---|---|---|---|---|
| **Number of Occupied Sites** | 390 | 220 | 260 | 149 | 1166 | 343 |

The environmental variables used to predict the location the occupied terraces in the next section are given below.

Location (Column T1 in the dataset), North grid coordinate (Column T10-12), East grid coordinate (Column T13-15), elevation (Column T24-26), topography (Column T27), soil type (Column T28), soil depth (Column T29), silting (Column T30), presence of a spring (Column T32), barranca or wash adjacent (Column T33), type of vegetation (Column T34), vegetation abundance (Column T35), special resources (Column T36), distance from road (Column T72), and the Ia Terrace occupation classification (sum of indicator ceramic greater than one).

## DATA MINING AND RULE EXTRACTION

The following data mining techniques were used; the One Rule hypothesis, Naïve Bayes, Alternating Decision Trees, J48 Decision Trees, and the Naïve Bayes leafed Decision Tree techniques. Each of the techniques was used as implemented in the WEKA data mining suite. WEKA's default settings for each of the techniques.

It is important to note that the database being used is collection of terraces that were populated over a period of about 2000 years. Very few of the sites were inhabited throughout all six of the time periods that make up those 2000 years. In fact, during most of the time periods, less than 1/5 of the sites are inhabited (according to the current Period identifications). IIIb-IV Period is the only exception, with a little over half the sites occupied. It is also important to note that only residential sites were included in the database.

This means that, for a given time period, a site not being occupied does not mean that it is a site that would not have been occupied. It may just mean that the terrace is slightly less desirable than the ones already occupied. So we decided to use a training set that included the instances representing the Ia Period terraces and an equivalent number of randomly chosen non-Ia Period terraces from the remainder of the database. We then ran each of the machine learning techniques with the WEKA default parameters on this training set and tested the resultant rules' accuracies on all 2073 instances. Table 5 gives the results.

**TABLE 5** Comparison of 1a classification results.

| **Ia Terrace Classification – 402 'no' instances and 390 'yes' Instances (792 total) as Training Set, results are after test on all 2073 instances.** | | | | |
|---|---|---|---|---|
| **Classifier** | **Parameters** | **Total Percent Correct** | **'no' Class Percent Correct** | **'yes' Class Percent Correct** |
| One Rule | -B 6 | 70.6% | 65.3% | 93.3% |
| Naïve Bayes | | 70.8% | 64.9% | 96.2% |
| AD Tree | -B 10 -E -3 | 76.5% | 71.3% | 98.7% |
| J48 Tree | -C 0.25 -M 2 | 81.7% | 78.3% | 96.2% |
| NB Tree | | 78.4% | 74.1% | 96.9% |

All of the techniques did quite well at predicting the location of occupied Phase 1a terraces. They were less successful in predicting unoccupied terraces. Since it is our assumption that the factor that limited the number occupied terraces in the area during the Ia Period is the lack people and not the lack of desirable terraces, then the percent of correctly identified occupied Ia Period terraces is more important than the percent of correctly identified unoccupied Ia Period terraces. That being said, we still consider the correct identification of unoccupied terraces to be important and the J48 decision tree did this better then the others. Since all of the techniques performed at a high enough level to be considered successful at identifying occupied terraces, the J48 decision tree technique was selected on that basis to be the best at identifying which terraces were desirable and which were undesirable during the Ia Period.

We are looking at which sites were occupied in an effort to try to identify which sites were desirable enough to occupy. It is safe to assume that any site that was occupied was desirable, but it is not safe to assume that any site not occupied was undesirable. By looking at some of the instances we can see that some unoccupied sites have very similar attributes to occupied sites. Examples of this can be seen in table 6 below. Instances 20 and 1234 have the same attributes and are located very near to each other, yet one is occupied and the other is not. The same can be seen with instances 272 and 260. It might be said that the sites are too close together and that when the one became occupied, the other become undesirable because of its proximity to an occupied site. However, there are sites that are closer together than these and are still both occupied. This could mean that there is some unknown social factor that is influencing site choosing, or that there simply weren't enough people in the area to occupy all the desirable sites. We shall assume for the purposes of this paper that these sites would have been occupied had there been more people. Therefore, classifying more terraces as desirable than there are terraces that were occupied is not indicative of erroneous results.

**TABLE 6** Example instances from Peroid 1a.

| **Attributes** | **Instance 20** | **Instance 1234** | **Instance 272** | **Instance 260** |
|---|---|---|---|---|
| Location | Monte Albán | Monte Albán | Monte Albán | Monte Albán |
| North Grid Coordinate | 189 | 183 | 175 | 171 |
| East Grid Coordinate | 340 | 339 | 311 | 308 |
| Elevation | 375 | 375 | 400 | 400 |
| Topography | Sloped | Sloped | Near Flat | Near Flat |
| Soil Type | 1 | 1 | 1 | 1 |
| Soil Depth | 0 | 0 | 0 | 0 |
| Silting | None | None | None | None |
| Presence of a Spring | Absent | Absent | Absent | Absent |
| Barranca or Wash Adjacent | Absent | Absent | Absent | Absent |
| Type of Vegetation | Grass and Brush | Grass and Brush | Grass and Brush | Grass and Brush |
| Vegetation Abundance | Moderate | Moderate | Moderate | Moderate |
| Special Resources | Quarryable Stone | Quarryable Stone | None | None |
| Distance from Road | Close | Close | Close | Close |
| Occupied in Ia Period | No | Yes | No | Yes |

The same set of techniques were applied to each of the other five time periods with the results shown in table 7.

**TABLE 7** Terrace classification for the remaining phases.

| **Ic Terrace Classification – 222 ‘no’ instances and 220 ‘yes’ Instances (442 total) as Training Set, results are after test on all 2073 instances.** | | | | |
|---|---|---|---|---|
| **Classifier** | **Parameters** | **Total Percent Correct** | **‘no’ Class Percent Correct** | **‘yes’ Class Percent Correct** |
| One Rule | -B 6 | 68.1% | 65.8% | 86.8% |
| Naïve Bayes | | 69.3% | 66.8% | 90.0% |
| AD Tree | -B 10 -E -3 | 81.4% | 81.7% | 78.6% |
| J48 Tree | -C 0.25 -M 2 | 79.7% | 78.8% | 87.3% |
| NB Tree | | 77.2% | 75.9% | 87.7% |

| **II Terrace Classification – 242 ‘no’ instances and 260 ‘yes’ Instances (502 total) as Training Set, results are after test on all 2073 instances.** | | | | |
|---|---|---|---|---|
| **Classifier** | **Parameters** | **Total Percent Correct** | **‘no’ Class Percent Correct** | **‘yes’ Class Percent Correct** |
| One Rule | -B 6 | 86.8% | 88.9% | 72.3% |
| Naïve Bayes | | 79.9% | 78.3% | 91.2% |
| AD Tree | -B 10 -E -3 | 82.0% | 81.3% | 86.9% |
| J48 Tree | -C 0.25 -M 2 | 87.0% | 87.1% | 85.8% |
| NB Tree | | 82.4% | 81.4% | 89.6% |

| **IIIa Terrace Classification – 157 ‘no’ instances and 149 ‘yes’ Instances (306 total) as Training Set, results are after test on all 2073 instances.** | | | | |
|---|---|---|---|---|
| **Classifier** | **Parameters** | **Total Percent Correct** | **‘no’ Class Percent Correct** | **‘yes’ Class Percent Correct** |
| One Rule | -B 6 | 84.1% | 85.2% | 70.5% |
| Naïve Bayes | | 77.0% | 76.4% | 84.6% |
| AD Tree | -B 10 -E -3 | 75.1% | 74.0% | 89.9% |
| J48 Tree | -C 0.25 -M 2 | 78.1% | 77.3% | 87.9% |
| NB Tree | | 80.0% | 79.7% | 83.9% |

| **IIIb-IV Terrace Classification – 907 'no' instances and 1166 'yes' Instances (2073 total) as Training Set, results are after test on all 2073 instances.** | | | | |
|---|---|---|---|---|
| **Classifier** | **Parameters** | **Total Percent Correct** | **'no' Class Percent Correct** | **'yes' Class Percent Correct** |
| One Rule | -B 6 | 71.2% | 55.0% | 83.7% |
| Naïve Bayes | | 68.7% | 69.0% | 68.5% |
| AD Tree | -B 10 -E -3 | 71.5% | 72.7% | 70.6% |
| J48 Tree | -C 0.25 -M 2 | 82.0% | 73.3% | 88.7% |
| NB Tree | | 75.4% | 70.7% | 79.2% |

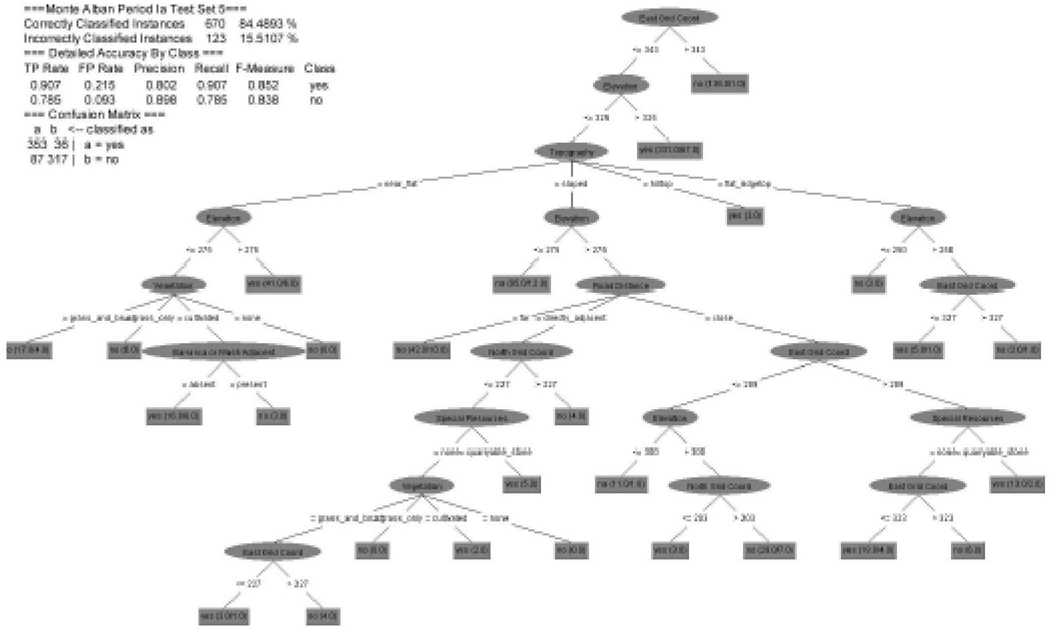
| **V Terrace Classification – 337 'no' instances and 343 'yes' Instances (680 total) as Training Set, results are after test on all 2073 instances.** | | | | |
|---|---|---|---|---|
| **Classifier** | **Parameters** | **Total Percent Correct** | **'no' Class Percent Correct** | **'yes' Class Percent Correct** |
| One Rule | -B 6 | 66.7% | 62.6% | 87.2% |
| Naïve Bayes | | 60.8% | 54.3% | 93.6% |
| AD Tree | -B 10 -E -3 | 71.8% | 68.4% | 88.6% |
| J48 Tree | -C 0.25 -M 2 | 73.3% | 69.3% | 93.3% |
| NB Tree | | 72.5% | 70.2% | 84.0% |

The results highlighted in yellow are those that best classify which sites are desirable and which are not in a given period. The best was chosen by taking the one that had the highest total percent correct among those that were within 6% of the best 'yes' identifier. Those highlighted in red are the second best. It should be noted that the rules produced by the J48 machine learning technique performed the best for five of the six time periods and was a close second for the other time period. Therefore, we will use the rules produced by the J48 method to do my analysis of the results.

The J48 method produces C4.5 trees for each of the periods. The one for period 1a is given in figure 1. When we use the extracted rules from the tree to predict the terraces occupied by agents in Phase 1a using RePast we get the results shown in figure 2 where occupied terraces are given in red and unoccupied in blue. The key variables here are elevation and location near the top of the hill.

## CONCLUSIONS

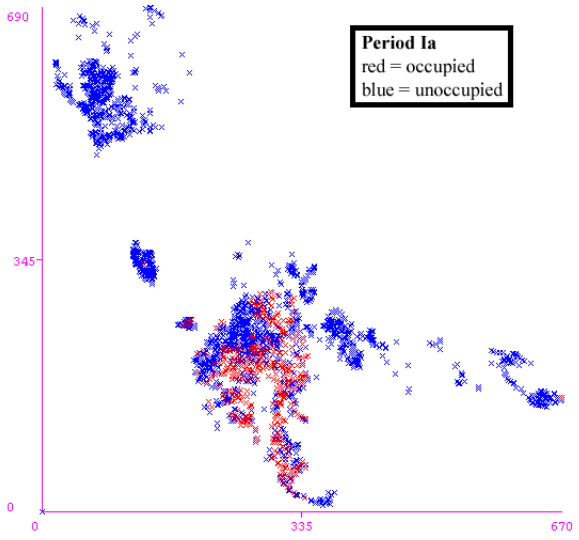
In this paper we demonstrate the ability of an agent-based approach to modeling the emergence of an archaic urban center using data mining techniques. In this approach we only investigated what terraces were occupied in a given phase. In subsequent work we can begin to predict the functionality of the terraces based on their contents of those of its neighbors.



```
===Monte Alban Period Ia Test Set 5===
Correctly Classified Instances     670    84.4893 %
Incorrectly Classified Instances   123    15.5107 %
=== Detailed Accuracy By Class ===
TP Rate  FP Rate  Precision  Recall  F-Measure  Class
 0.907    0.215     0.802     0.907    0.852     yes
 0.785    0.093     0.898     0.785    0.838     no
=== Confusion Matrix ===
  a   b   <-- classified as
 353  36 |  a = yes
 87 317 |  b = no
```

**FIGURE 1** The decision tree for Period 1a.

**FIGURE 2** Simulated occupation of Monte Alban in Phase 1a based on the extracted rules.



# BIBLIOGRAPHY

Blanton, R. E. 1978, *Monte Albán: Settlement Patterns at the Ancient Zapotec Capital*. Academic Press.

Blanton, R. E., S. Kowalewski, G. Feinman, and J. Appel, 1982, *Monte Albán's Hinterland, Part I, the Prehispanic Settlement Patterns of the Central and Southern Parts of the Valley of Oaxaca, Mexico*. The Regents of the University of Michigan, The Museum of Anthropology.

Crowston, K., 1994, "Evolving Novel Organizational Forms," *Computational Organization Theory,* K. M. Carley and M. J. Prietula, eds., Lawrence Erlbaum Associates Publisher, Hillsdale, New Jersey.

Fogel, D. B. 1995, *Evolutionary Computation - Toward a New Philosophy of Machine Learning*. IEEE Press.

Fox, M., M. Barbuceanu, M. Gruninger, and J. Lin, 1998, "An Organizational Ontology for Enterprise Modeling," *Simulation Organizations Computational – Models of Institutions and Groups,* M. J. Prietula, K. M. Carley, and L. Grasser, eds. , AAAI Press/ MIT Press Menlo Park, California, Cambridge Massachusetts.

Goldberg, D. E., 1989, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Company, Inc.

Holland, J. H., 1975, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: University of Michigan Press.

Kohler, T., Gummerman, G., and Reynolds, R. G., 2005, "Virtual Archaeology", *Scientific American*, Vol. 293, no. 1, pp: 76-84.

Kowalewski, S. A., G. M. Feinman, L. Finsten, R. E. Blanton, and L. M. Nicholas, 1989, *Monte Albán's Hinterland, Part II, Prehispanic Settlement Patterns in Tlacolula, Etla, and Ocotlan, the Valley of Oaxaca, Mexico*. Vol. 1. The Museum of Anthropology, University of Michigan, Ann Arbor.

Marcus, J. and K. V. Flannery, 1996, *Zapotec Civilization: How Urban Societies Evolved in Mexico's Oaxaca Valley*. Thames and Hudson Ltd, London.

Reynolds, R. G., 1984, "A Computational Model of Hierarchical Decision Systems," *Journal of Anthropological Archaeology*, No. 3, pp. 159-189.

Reynolds, R. G., 1994, "An Introduction to Cultural Algorithms," pp. 131-139 in *Proceedings of the Third Annual Conference on Evolutionary Programming,* A. V. Sebald and L. J. Fogel, eds., River Edge, NJ, World Scientific Publishing.

Reynolds, R. G., 1986 "Multidimensional Scaling of Four Guilá Naquitz Living Floors", in *Guilá Naquitz: Archaic Foraging and Early Agriculture in Oaxaca, Mexico*, K. V. Flannery, Editor, Academic Press, pp: 439-500.

Reynolds, R. G. and H. Al-Shehri, 1998, "Data Mining of Large-Scale Spatio-Temporal Databases Using Cultural Algorithms and Decision Trees", in *Proceedings of 1998 IEEE World Congress on Computational Intelligence*, Anchorage, Alaska, May 4-9.

Reynolds, R. G. 1999, "The Impact of Raiding on Settlement Patterns in the Northern Valley of Oaxaca: An Approach Using Decision Trees," in *Dynamics in Human and Primate Societies,* T. Kohler and G. Gumerman, eds., Oxford University Press.

Reynolds, R.G. and A. Lazar, 2002, "Simulating the Evolution of the Archaic State", in *Proceedings of World Congress on Computational Intelligence*, Honolulu, Hawaii, May 12-19.

Reynolds, R.G., Kobti, Z., Kohler, T., and Yap, L., 2005, "Unraveling Ancient Mysteries: Re-imagining the Past Using Evolutionary Computation in a Computer Gaming Environment", *IEEE Transactions on Evolutionary Computation*, Vol. 9, No. 6, pp: 708-720.

Scacchi, W.,1998, "Modeling, Simulating, and Enacting Complex Organizational Process – A Life Cycle Approach," *Simulation Organizations Computational – Models of Institutions and Groups.* M. J. Prietula, K. M. Carley, and L. Grasser, eds., AAAI Press/ MIT Press Menlo Park, California, Cambridge Massachusetts.

Spencer, C. S. and E. M. Redmond, 2001, "Multilevel Selection and Political Evolution in the Valley of Oaxaca, 500-100 B.C.", in *Journal of Anthropological Archaeology* 20:195-229.

Spencer, C. S. and E. M. Redmond, 2003, "Militarism, Resistance, and Early State Development in Oaxaca, Mexico. *Social Evolution & History* 1(2):26-72.

Utgoff, P. E., 1989, "Incremental Induction of Decision Trees," in *Machine Learning*, P. Langley, ed., Boston Kluwer.

Weiss, S. M. and N. Indurkhya, 1998, *Predictive Data Mining A Practical Guide.* Morgan Kaufman Press.

# EXPLORATORY MODELING OF ENSEMBLES FOR TESTING DECISION THEORY PARADIGMS

L.A. KUZNAR,[*] Indiana University – Purdue University, Fort Wayne, Fort Wayne, IN

## ABSTRACT

Purely textual or mathematical representations suffer from intractability and vagueness respectively, making the rigorous comparison of theories, or of paradigmatic approaches, virtually impossible in traditional social science. Computational models provide an unprecedented ability to represent and test social theory. Different theories, represented as algorithms, can be tested alone or against one another *in silico* and in any combination imaginable. The advantages of this new capability are counteracted by the burden of trying to compare the effects of so many combinations of theoretical elements. Exploratory modeling systematically searches large parameter spaces of an ensemble of different models, and identifies models with the most explanatory power. This approach is used when competing models are characterized by "deep uncertainty" in their parameters and variables. Social scientists' often disagree strongly on what variables are relevant or what values parameters should have in social scientific theories. Therefore exploratory modeling provides a tool that can enhance scientific decision-making. I provide an example of exploratory modeling for testing the validity of decision theories from rational choice, sigmoid utility, bounded rationality, and prospect theory paradigms. Political alliance formation in an Irian Jaya tribe is used as an empirical test case. In this case, exploratory modeling provides a way of comparing the validity of theories derived from different paradigms, and also suggests new hypotheses that may better explain the data.

**Keywords**: exploratory modeling; ensembles; decision theory; theory testing

## INTRODUCTION

Traditional representations of theory in the social sciences rely on textual descriptions or mathematical representations. Textual descriptions suffer from the ambiguity of natural language. Mathematical modeling forces a rigor upon social theories, but many social phenomena are too complex or path dependent to allow tractable solutions (Holland, 1998). Furthermore, when theories from different paradigms (for example rational choice vs. bounded rationality) incorporate different variables and assumptions, it may be impossible to represent them in the same mathematical framework. Computational approaches in the social sciences are more rigorous than text but more flexible than mathematical formulations, and so appear to present a workable compromise for representing social theory (Sallach, 2003). An advantage of computational social science (CSS) is that seemingly incommensurable theories can be represented algorithmically and placed in the same simulation environment where their implications can be explored and their relative explanatory power compared. Rigorous testing of these competing theories would involve a thorough search of all potential variables and parameters.

---

[*] *Corresponding author address*: Lawrence A. Kuznar, Dept. Anthropology, Indiana University – Purdue University, Fort Wayne 46805; e-mail: kuznar@ipfw.edu.

The promise of CSS comes, however, at a cost. While it may be possible to represent competing theories and varying parameter levels in a computational framework, the resulting space of possibilities (theory space) can be infinite. Realistic theories may possess hundreds or thousands of variable combinations. Furthermore, if the theories differ on the value of continuously varying parameters, the resulting parameter spaces can be infinite. Exploratory modeling is a potential solution to this problem, and employs an ensemble of models representing different theories that is explored for the set of models that may best represent the state of scientific knowledge (Bankes, 1993).

In this paper, I will present an application of exploratory modeling for testing competing decision theories that are derived from very different paradigms. 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.

## EXPLORATORY MODELING

Bankes (1993) proposed exploratory modeling as an aid for decision makers who are very uncertain about their models of the world. Scientists are often ignorant about some phenomena because they happen too infrequently (nuclear catastrophes, meteor strikes, class 5 hurricanes). In other cases, paradigmatic differences are so great that scientists cannot agree on how to approach an explanation (economic behavior, the evolution of human behavior, causes of terrorism). Low probability events and scientific controversies are characterized by "deep uncertainty," or ignorance of what variables and causal relationships hold or what parameters may characterize complex systems (Bankes, 2002:7263). Exploratory modeling provides a potential method for comparing models characterized by deep uncertainty.

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.

The technical details of how to search vast parameter spaces are open to discussion, but sampling strategies such as Monte Carlo and Latin Hypercube sampling (Kleijnen, 1997), patient rule induction methods (PRIM) (Lempert, et al., 2006), the use of neural networks (Bankes, 1993), and genetic algorithms (Miller, 1998) have been used or proposed. 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 and Frederick, 2003). 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). The omniscience implied by these assumptions is an overstatement of human capabilities (Gigerenzer and Selten, 2001; Klein, 2001). The assumption of strict self-interest has also been strongly criticized and demonstrated to be limited in its applicability (Fehr and Schmidt, 1999). 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). 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). The Arrow-Pratt measure of risk aversion measures an individual's risk sensitivity. It is calculated as: $AP=-W''/W'$, Where W is a wealth distribution function estimated, in this case, based on Kapauku men's wealth. Positive values indicate risk aversion and negative values indicate risk proneness. In the model's coordination game the risky choice is to cooperate, so the Join probability is altered as a proportion of an individual's risk proneness to the overall risk proneness of the population (Kuznar and Kobelja, 2006a). The most risk prone individuals always join, the least never 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. These functions adjust the Nash optimum join probability by weighting it, or by adjusting the game's payoffs according to the disutility of losses or utility of gains. 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.

## 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). Note that this does not represent all of the possible and scientifically reasonable ways that these theories might be combined. Instead, this reflects the state of debates among social scientists. The theory space that results from cross-comparison of these 24 models is a 24X24 matrix of 576 outputs, indicating how rapidly the theory space of an ensemble can grow. 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) |

Several metrics were used to evaluate the efficacy of each model, including: the squared error of predicting the number of coalitions, the squared error of predicting mean coalition size, and the squared error of predicting the frequency distribution of coalition sizes (see Kuznar and Kobelja, 2006b). The best models predicted coalition number and size within 10-12% of the observed metric (number of coalitions, coalition size), whereas poor models typically predicted metrics to only 30-40%.

Most models did not perform very well, and for brevity are not presented here. Six models performed well, including the Nash optimum (N), sigmoid utility (S), 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. Four models showed the most promise, including SG, PT, sP3, and sC2. The naïve agent conformism 2 model (nC2) is included in this analysis as a typical example of a poor model. Models are compared by examining differences in their squared errors from actual data. Models that are statistically significantly different from the poor nC2 model provide especially close fits to the original data (Table 2).

**Table 2.** Model Performance in the Kapauku Simulation.

| | Metric Differences p-value | | |
|---|---|---|---|
| Model Comparisons | No. Coalitions SE | Mean Coalitions SE | Coalition Size Distribution SE |
| NC2 – SG | **0.003** | **0.003** | **0.000** |
| NC2 – PT | **0.005** | **0.003** | **0.001** |
| NC2 – sP3 | **0.005** | **0.026** | **0.000** |
| NC2 – sC2 | **0.003** | **0.007** | **0.000** |
| SG – PT | 0.827 | 0.509 | 0.159 |
| SG – sP3 | 0.694 | 0.990 | 0.331 |
| SG – sC2 | 0.397 | 0.713 | **0.007** |
| SP3 – sC2 | 0.494 | 0.616 | **0.015** |

The four best models fit the data much better than the vast majority of models as represented by naïve conformism 2 (nC2), with each model showing very strong and statistically significant differences from the poorer model on all metrics. The best of all the models, smart conformism 2 (sC2) additionally demonstrated statistically significantly better fits to the distribution of coalition sizes than either the sigmoid group (SG) or smart prestige bias 3 (sP3) models.

I would caution against concluding that the best fitting model, sC2, is confirmed and its competitors falsified, since it outperformed on only one metric presented here, and provided fits closer by a factor of at most 5%. 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, "Genesis of Suicide Terrorism," *Science* **299**(5612):1534-1539.

Bankes, S., 1993, "Exploratory Modeling for Policy Analysis," *Operations Research* **41**(3):435-449.

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.

Boyd, R. and P. J. Richerson, 1985, *Culture and the Evolutionary Process*, Chicago: Unversity of Chicago Press.

Cowell, F., 1986, *Microeconomic Principles*, Oxford: Oxford University Press.

Fehr, E. and K. M. Schmidt, 1999, "A Theory of Fairness, Competition, and Cooperation," *Quarterly Journal of Economics* **114**:817-868.

Friedman, M. and L. J. Savage, 1948, "The Utility Analysis of Choices Involving Risk," *Journal of Political Economy* **4**:279-304.

Gigerenzer, G. and R. Selten, 2001, "Rethinking Rationality,"in *Bounded Rationality: The Adaptive Toolbox*, Gigerenzer, G. and R. Selten (Ed.), pp. 1-12, Cambridge, Massachusetts: MIT Press.

Holland, J. H., 1998, *Emergence: From Chaos to Order*, New York: Basic Books.

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, "Prospect Theory: An Analysis of Decision under Risk," *Econometrica* **47**(2):263-291.

Kahneman, D. and A. Tversky, ed. 2000, *Choices, Values, and Frames*, Cambridge: Cambridge University Press.

Kleijnen, J. P. C., 1997, "Sensitivity Analysis and Related Analyses: A Review of Some Statistical Techniques," *Journal of Statistical Computer Simulation* **57**:111-142.

Klein, G., 2001, "The Fiction of Optimization," in *Bounded Rationality: The Adaptive Toolbox*, Gigerenzer, G. and R. Selten (Ed.), pp. 103-121, Cambridge, Massachusetts: MIT Press.

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. 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 N. Kobelja, 2006a, Simulating Tribal Politics: An Agent-Based Model of Kapaukuan Bigman Coalition Building, Paper Presented at the 2nd Annual Society for Anthropological Sciences (SASci) Meeting, February 25, 2006, Savannah, Georgia.

Kuznar, L. A. and N. Kobelja, Year, "Validating Social Simulations: A Case of Political Alliances in Tribal New Guinea," *Proceedings of the 2006 NAACSOS Conference* Notre Dame University, South Bend, Indiana.

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*.

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.

Miller, J. H., 1998, "Active Nonlinear Tests (Ants) of Complex Simulation Models," *Management Science* **44**(6):820-830.

Palmer, T. N., 2000, "Predicting Uncertainty in Forecasts of Weather and Climate," *Reports of Progress in Physics* **63**:71-116.

Pospisil, L., 1963, *The Kapauku Papuans of West New Guinea*, New York: Holt, Rinehart and Winston.

Pospisil, L., 1972, *Kapauku Papuan Economy*, New Haven, Connecticut: Human Relations Area Files.

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.

Sallach, D. L., 2003, "Social Theory and Agent Architectures: Prospective Issues in Rapid-Discovery Social Science," *Social Science Computer Review* **21**(2):179-195.

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, "Advances in Prospect Theory: Cumulative Representation of Uncertainty," *Journal of Risk and Uncertainty* **5**(4):297-323.

# Simulating Social Games

# IDEALIZATION, UNCERTAINTY, AND HETEROGENEITY: GAME FRAMEWORKS DEFINED WITH FORMAL CONCEPT ANALYSIS

M.T. RACOVITAN,[*] Northern Illinois University
D.L. SALLACH, Argonne National Laboratory, Argonne, IL,
and The University of Chicago, Chicago, IL

## ABSTRACT

The present study begins with Formal Concept Analysis, and undertakes to demonstrate how a succession of game frameworks may, by design, address increasingly complex and interesting social phenomena. We develop a series of multi-agent exchange games, each of which incorporates an additional dimension of complexity. All games are based on coalition patterns in exchanges where diverse cultural markers provide a basis for trust and reciprocity. The first game is characterized by an idealized concept of trust. A second game framework introduces uncertainty regarding the reciprocity of prospective transactions. A third game framework retains idealized trust and uncertainty, and adds additional agent heterogeneity. Cultural markers are not equally salient in conferring or withholding trust, and the result is a richer transactional process.

**Keywords:** Trust, formal concept analysis, games, idealization, uncertainty

## INTRODUCTION

Tension exists between formal models and the phenomena they are intended to represent, especially in the social sciences. The former are powerful but often fail to map well to empirical patterns, while the latter frequently teem with diverse complexities that elude effective representation. Investigating how such a tension may be overcome is a major arena of research in the computational social sciences, and such studies may start at either pole.

The present study begins with a powerful formalism, Formal Concept Analysis (FCA), and undertakes to demonstrate how successive game frameworks may address increasingly complex and interesting social phenomena by design. More specifically, we develop a series of multi-agent exchange games, each of which incorporates an additional dimension of complexity.

All games are based on coalition patterns in exchanges where diverse cultural markers provide a basis for trust and reciprocity. Trust is an idealized resource, where the term ‘idealization’ indicates that neither perfect trust nor a complete lack of trust occurs in the exchange process. On the contrary, as an idealized concept, trust serves as a referent for empirical choices but never occurs in its pure form. The role of such an idealized concept characterizes the first game. A second game framework retains idealized trust dynamics, but adds additional agent heterogeneity. In particular, cultural markers are not equally salient in conferring or withholding trust, and the result is a richer transactional process. A third game

[*] *Corresponding author address*: Mihai T. Racovitan, Northern Illinois University, Department of Mathematical Sciences, DeKalb, IL 60115; e-mail: racovita@math.niu.edu.

framework introduces uncertainty regarding the efficacy of the cultural markers of prospective trading partners.

## Formal Concept Analysis

We use Formal Concept Analysis (Ganter and Wille 1999) in two ways: (1) to bind together what is encountered in our model (e.g., hierarchies, logic, trust values, and vector spaces of solutions or payoffs), and (2) to exploit the duality of concept lattices. We can prove one statement and get another one that is equivalent and complementary using duality.

A set together with a binary ordering relation that satisfies the properties of reflexivity, antisymmetry, and transitivity is called a partially ordered set (or a poset). A *lattice L* is a poset, in which any two elements $v_1$, $v_2 \in L$ have a unique greater lower bound (**glb**) $v_1 \wedge v_2$ and a unique lowest upper bound (**lub**) $v_1 \,\hat{}\, v_2$ in $L$.

In a *complete lattice L*, any subset $\underline{S} \subseteq L$ has an infimum and a supremum. A lattice can be viewed as having the structure of an algebra, with these two binary operations $\wedge$, $\vee$. Later, we can introduce distributivity and complementation, if needed. For example, the power set is a distributive, complemented lattice with respect to inclusion.

Define a *context* as the tuple K: = (G, M, I), where G is the set of objects, M the set of attributes that the objects in G may have, and $I \subseteq G \times M$ (i.e., $(g, m) \in I$ means that the object g has the attribute m). For $A \subseteq G$ and for $B \subseteq M$ denote

$$A' := \{b : \forall\, a \in A, (a,b) \in I\}\ ,\ B' := \{a : \forall b \in B, (a,b) \in I\}.$$

By a *formal concept* we understand a pair (A, B), where $A \subseteq G$ is an *extent*, $B \subseteq M$ an *intent*, and $A'$ = B, B′ = $A$. We can now define the *concept lattice* to be the set of all concepts of K, $\underline{C}(K) := \{(A, B)\} \in 2^G \times 2^M$ with the ordering

$$C_1 \prec C_1 := A_1 \subseteq A_2 \text{ (which implies } B_1 \supseteq B_2).$$

$\underline{C}$ is a complete lattice with infimum and supremum:

$$O(\underline{C}) = \bigwedge_{C \in \underline{C}} C = \left( \bigcap_{C \in \underline{C}} A\ ,\ \left( \bigcup_{C \in \underline{C}} B \right)^{''} \right)\ ,\ 1(\underline{C}) = \bigvee_{C \in \underline{C}} C = \left( \left( \bigcup_{C \in \underline{C}} A \right)^{''}\ ,\ \bigcap_{C \in \underline{C}} B \right)$$

The central notion of FCA is the duality between what we call "formal objects" or "extent" and "formal attributes" or "intent." Because of this duality, concept lattices form what we call a "Galois Connection."

## Relational Data

One way to represent data is by using a relational data representation, as in Codd (1970) or Joslyn and Mniszewski (2002). We present here the mathematical aspect of relational data. Given sets $S_1, S_2, \ldots, S_n$ (not necessarily distinct), $R$ is a relation of these $n$ sets if it is a set of *n*-uples, each of which has its first element from $S_1$, its second element from $S_2$, and so on. As defined above, *R* is said to have *degree n*. We can represent the data by arrays that represent *n*-ary relations as in Codd (1970). The rows represent *n*-tuples of the relation, and each column is labeled with the name of the corresponding domain$S_j$. All rows are distinct, and the ordering of the columns is significant. For example, consider a small set of five agents that have a specific religion and ethnicity,

$$ID := \{\text{Agent A, Agent B, Agent C, Agent D, Agent E}\}\ ,$$

$$\text{Religion} := \{\text{Religion 1, Religion 2, Religion 3}\}\ ,$$

$$\text{Ethnicity} := \{\text{Ethnicity 1, Ethnicity 2, Ethnicity 3}\}\ .$$

and we construct a relation of degree 4 called 'type' by adding a new attribute called 'quantity:'

**TABLE 1** The relation of degree 4, 'type'

| type | (agent | religion | ethnicity | quantity) |
|---|---|---|---|---|
| | A | 1 | 1 | 20 |
| | B | 1 | 1 | -10 |
| | C | 2 | 2 | 22 |
| | D | 2 | 3 | -12 |
| | E | 3 | 3 | -10 |

Relational data can be analyzed by Formal Concept Analysis. This may be different from the common operations on relations, like join. We identify a binary relation between the fields, and we accommodate multiple fields by using *unions* of binary relations as in Joslyn and Mniszewski (2002). We then have to scale to recover the Boolean Relation (True/False):

**TABLE 2** The scaled version of Agents (Religion ∩ Ethnicity)

| | Religion 1 | Religion 2 | Religion 3 | Ethnicity 1 | Ethnicity 2 | Ethnicity 3 |
|---|---|---|---|---|---|---|
| Agent A | X | | | X | | |
| Agent B | X | | | X | | |
| Agent C | | X | | | X | |
| Agent D | | X | | | | X |
| Agent E | | | X | | | X |

The scaled table is called now a context where, in FCA terms, the rows are the objects and the columns are the *attributes*.

## Game Theory

We consider a private market with two types of goods: religious and ethnic artifacts, respectively. The agents are characterized by their religion, ethnicity, their location, their quality as sellers, producers, or buyers, and their history of trades. Each agent faces a decision problem when s/he enters in the market and interacts with the other agents. We posit a small network, which means that s/he has at least partial knowledge of the others' religion, ethnicity, and history of trades. Each agent has her own strategy with respect to the situation in which s/he is involved in the market.

First, decision problem involves a finite set of histories, $H$. A history in $H$ is a possible sequence of actions that can be taken by the decision maker or by chance. The empty sequence is contained in $H$, and we build $H$ by appending the action taken at the moment $t$. The history $(\alpha_1,\ldots,\alpha_t) \in H$ is *terminal* if there is no $(\alpha_1,\ldots,\alpha_t,\alpha) \in H$. The set of actions available to the decision maker or to chance is denoted $A(h) = \{\alpha \,|(h, \alpha) \in H\}$, where $h$ is nonterminal.

A *(pure) strategy* for the decision maker is a function $f$ that assigns an element of $A(h)$ to each nonterminal history $h$, with the restriction that if $h$ and $h'$ are in the same information set, then $f(h) = f(h')$ . The more natural definition of a strategy is a *plan of action*: a function that assigns an action only to histories reached with positive probabilities. There are two extensions of the notion of strategy: *behavioral* and *mixed* strategies.

The last components of a game are the decision maker's belief about the chance player's behavior $\rho$ , and a utility function $u$ that assigns a number (payoff) to each terminal history. For more details, see Rubinstein (1998). Game theory offers conditions for a strategy to be optimal for the player, and these conditions vary for each type of game or strategy used.

## THE CONCEPT OF TRUST IN A TRADE NETWORK

Our aim is to represent *trust* as a dynamic concept and to reason about attitudes of trust with regard to a base set of trustees, including the idealization of trust. We start with a conceptual analysis of trust, and we study what is called *relative trust*: of an agent that trusts a variety of other agents in its neighborhood but in various settings and with different degrees of confidence.

### The theory of trust — what do we formalize?

The extant research literature about trust suggests that the concept of trust is not easy to define. Diverse meanings of trust do not converge to a precise notion of trust.

By analyzing a core common to most of the attempts to define trust, Jones (2002) suggests that aspects as risk, reliance/dependence, and desired/intended goal are not necessary features of trust, except in some particular kinds of contexts or scenarios. Jones conjectures that the core of agent $x$'s trusting attitude lies in two beliefs: a "'rule belief'" and a "'conformity belief'." According to Jones, the formalization depends on this core; we say that it does tend to be neither unique, nor complete, nor a core.

Second, an important distinction between trust and trustworthiness arise in many papers. According to Ben-Ner and Putterman (2003), '*A trusts B*' means that "A believes that the probability that B will not harm her, or that B will live up to his commitment, is high enough to warrant some potentially risky action on A's part." Ben-Ner and Putterman (2003) continue:

> The *extent* of trust is a function of both characteristics of A and B and characteristics of the situation. Among these characteristics is B's *degree of trustworthiness*, an *attribute* that is determined by the costs and benefits associated with building and maintaining a reputation, the cost and ability to deceive, the agent's preferences and values regarding acceptable modes of behavior, and the anticipated actions of other parties that bear on the costs of engaging in prohibited behavior.

The authors consider here two explicit meanings of trust: trust as a probability, and trust as an attribute, and an implicit meaning of trust as a state. However, the relation called trustworthiness in Ben-Ner and Putterman (2003) actually means trust as *relationship*, only that it is directed one way, from agent A to agent B. The meaning of trust as a state is also problematic: agent A's disposition to trust is becoming transparent only inside the social events, and it is subject to change. Trust as a state is considered most likely as a unary relation, or, if we have in mind Peirce's categorization (1868), firstness.

The *trustworthiness relation* in the case of Table 2 (above) can be extended to a relation between the subsets of the following Cartesian product:

$$RE = \{Religion_1, Religion_2, \ldots, Religion_n\} \times \{Ethnicity_1, Ethnicity_2, \ldots, Ethnicity_n\}.$$

Each agent in the network is characterized by a pair (Religion, Ethnicity) $\in$ RE at this time. S/he maps each other agent pair to the set of probabilities within the trust lattice values.

Moreover, the agents develop trustworthiness relations based on their history of trades. We hope to capture the essential attributes of a cohesive community through games and trades, by considering that having the same religion and/or same ethnicity has a stronger effect on the agent actions than the history of trades. We cannot neglect the fact that an agent forms a *reputation* inside the society, related to her history of trades. Suppose agent A trades with agent B. Reputation acts here as a meaning of trust: agent A's trust in agent B because of agent B's reputation. Agent B's *reputation* is also called "'type'" from the game theorists' point of view.

Concerning game theory, we can analyze the trust by specifying specific concepts of game theory, such as preferences and values, as related to trust. Ben-Ner and Putterman (2003) make a distinction between preferences: self-interest, other-interest, and process-regarding (a distinction that is very useful). We usually associate the payoffs, maximization of profit, or minimization of loss in game theory to the agent's self-interest. Members of the same ethnic or religious community do have other-regarding preferences, like the concern for others' well-being, or process-regarding preferences, like the adherence to norms, rules, or principles.

The dynamic concept of trust tries to capture also what kind of incentives agent B faces to live up to her word given to agent A, or what affects the trust of agent B in agent A. Social scientists often refer to community-based distinctions (language, nationality, ethnicity, religion, etc.) as "cultural markers." However, this description is analytical, i.e., from the perspective of one for whom the markers are relatively arbitrary, but are attributed diverse kinds of significance by the endogenous participants. In this sense, ethnicity and religion in the present model can be considered cultural markers.

However, it is also important to recognize that treating cultural markers as variables in a model is reductionist, in the sense that it flattens the social basis of such distinctions. For each culturally significant marker, there is a variety of rituals, collective activities, interlocking roles, shared institutions, etc., that gives it meaning and emotional foundation. Thus, while the transactions may be a source of bonding for traders and other economic actors, this bond actually arises out of a shared community that collectively participates in multiple roles and activities. Our model is not sufficiently complex to represent these processes directly, but we do recognize that there are deeply interactive social processes out of which all such significant symbols arise. This aspect is suggested by Garfinkel (1963), who suggests that trust may be regarded as a product of situations governed by constitutive expectancies.

The reputation of agent B as an "'ineffective trader'" is important if agents A and B have different religions and ethnicities. However, the situation changes if they belong to same religion or ethnicity; 'community norms' could form an incentive for agent A to trade with agent B. This is a deviation from "'agent rationality'" in games; it is bounded by societal norms, rules, or specific religious or ethnic business aspects.

The *uncertainty* plays an important role in trades: that agent A is uncertain of agent B's history of trades, religion, ethnicity, or even of "'community norms.'" Including uncertainty in social games will reveal more about the community markers, and it is probably the most important part of our analysis because it shows how idealizing trust can work inside community. For example, when agent A is aware of community norms and decides to abide by them, we could expect that agent A will help agent B, regardless of the reputation that agent B might have. The fact that an agent expect the other agent's behavior according to the 'community norms' facilitate the idealization of trust; on the contrary, when uncertainty is a factor that each agent has to overcome, could reveal the tensions inside the community, and the relation with the idealized trust. By idealized trust, we mean not the state of trust, but the *relationship*.

To resume, trust has multiple meanings coming from: state, attribute, relationship, subjectiveness, emotion, cognition, inclination, probability, scope, reputation, norm, etc. Our model capture the relationships between agents, their orientation (emotion and cognition), the probabilities associated to relationships, and (implicit) reputation.

## Games and trust

We have five players that can be buyers, sellers, or producers, which approximates a cohesive trade network. We consider that there two types of goods on the market, ethnic and religious artifacts, respectively; the game concerns quantities of these artifacts, which can range from cheap to expensive. Each player initially chooses his or her strategy based on self-interest, but each may be motivated by other things, considering that cultural markers are important. They could be motivated by the fact that they are penalized by the community norms as they follow only their self-interest or because stability is more important than short-term gain.

In cohesive communities, players learn rapidly about others' characteristics, such as religion, ethnicity, or history of trades: agent A learns quickly if he can trust agent B or not.

However, the players cannot know how the market will evolve: the trust value of the other players may change. Agents could be brothers, or one of the agents may not live up to his commitment in the next period. Therefore, agents play repeated games with observable actions, and their strategies vary due to their characteristics (buyer or seller, religion, ethnicity, history), trust lattices, and position in the network.

We have three types of games characterized by increasing complexity:

1. The agents follow only their self-interest and have possible knowledge of the others' history of trades.

    a. Agent A supplies quantities *s* and *t* (according to a supply curve for religious and ethnic artifacts, respectively) to maximize his or her profits; one of the quantities could be zero, meaning that the agent produces or sells only one type of artifact.

    b. Agent B demands quantities *d* and *e* (according to a demand curve) to minimize her expenses.

    c. Transaction could be mediated by a reseller, e.g., agent C.

    d. These are simple games based on trust and reciprocity. The lattices are considering "'idealized trust'" as a comparison for trust as *relationship*.

2. We introduce uncertainty regarding the efficacy of the history of prospective trading partners.

3. We vary the uncertainty concerning trading processes and add the heterogeneity of agent cultural markers.

We use game theory as a tool; our objective here is analyzing the concept of trust. Therefore, this is not a study of so-called "'trust games'" in the economic literature. Rather, we are interested in formalizing the concept of trust in the social games' development.

## Formalizations of trust

Suppose, for the remainder of this discussion, that we have a network with the five agents depicted in Figure 1:

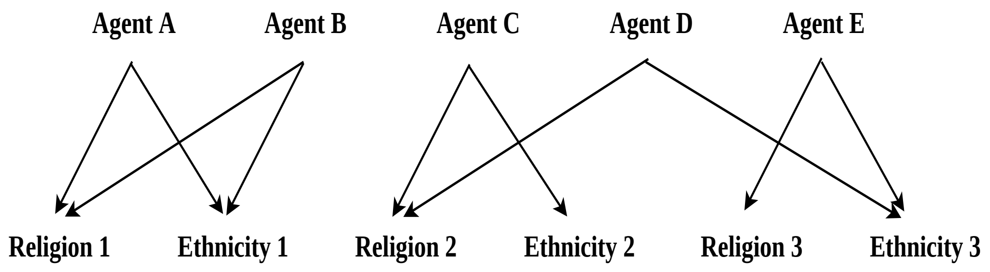


**FIGURE 1** One case of five agents, three religions, and three ethnicities

This bipartite graph is associated with our example from Table 2 from the section "Relational Data" (above).

Trust can be simply represented in a lattice according to the probabilities that agents associate to the *trustworthiness relation*. For the first case in Figure 2, we consider a trust value of 0.4 when two agents have the same religion or the same ethnicity. If agents have different religion and ethnicity, or if they are uncertain of these, the trust value is 0.05. For the second case in Figure 2, we associate a probability of 0.45 to religion.

If two agents are similar, having the same religion and ethnicity, we sum over the probabilities in both cases in Figure 2:

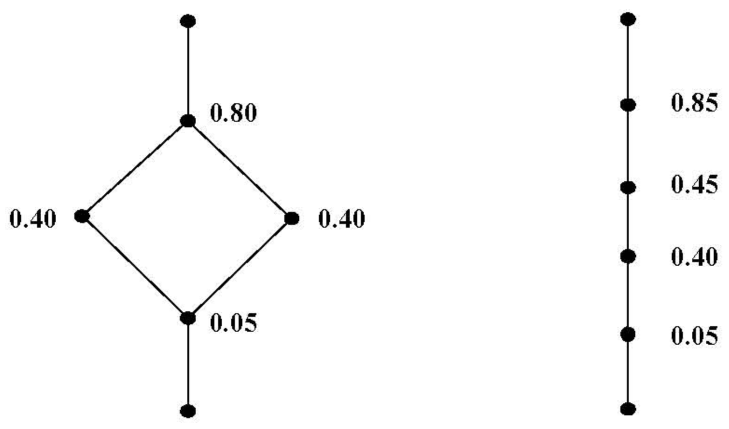


**FIGURE 2** Simple trust lattices. First, religion and ethnicity contribute with same probability, 0.40. Second, religion has probability 0.45.

Trust as a probability usually simplifies the 'relationship' between two agents, because it considers only one direction: agent A trusts agent B; distrust is incorporated here in the same values interval as trust. We use these simple lattices for two purposes: they constitute a first level in formalizing trust, and for games. If agents A and B play a zero sum game (trades or supply-demand), the probabilities represented in Figure 2 are associated with mixed strategies.

To increase complexity, we can introduce a pair of values by constructing a trust score space, as in Victor *et al.* (2006) (see Figure 3). The trust score space is the set [0,1]×[0,1] with two ordering relations, $\leq_t$, $\leq_k$, where the former orders the trust score from complete distrust (0,1) to complete trust (1,0), and the latter evaluates the amount of available trust evidence, going from "'shortage of evidence'," $x_1+x_2<1$ (incomplete information), to an "'excess of evidence'," $x_1+x_2>1$ (inconsistent information):

$$(x_1, x_2) \leq_t (y_1, y_2) \quad iff \quad x_1 \leq y_1 \quad and \quad x_2 \geq y_2$$
$$(x_1, x_2) \leq_k (y_1, y_2) \quad iff \quad x_1 \leq y_1 \quad and \quad x_2 \leq y_2$$

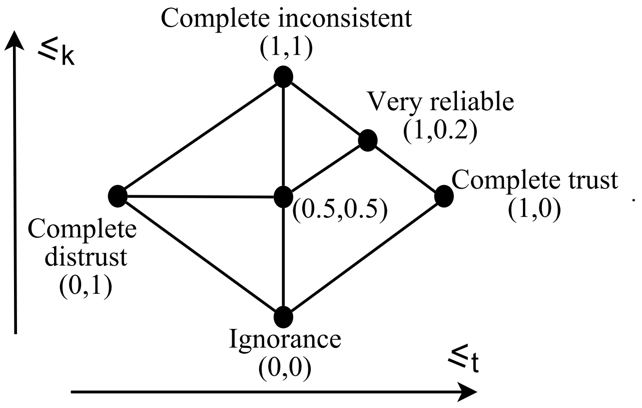


**FIGURE 3** The model in Victor et al. (2006)

We introduce more complexity by using FCA with a set of objects, a set of attributes, and the incidence relation. We have two ways of representation; we consider the set of agents to have different attributes "'to trust'," or to treat the concept of trust directly as an object, and define its attributes. The simple lattices from Figure 2 are used to characterize the network in Figure 1 from agent D's perspective.

Agent D categorizes the other agents by their characteristics. We can order agent D's preferences. First, the case when agent D has no information on the other agents A, B, C, and E (s/he trusts them equally, with probability 0.05):

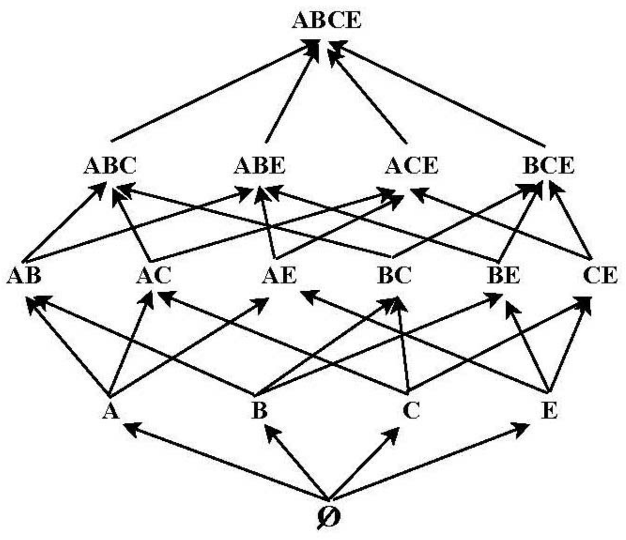


**FIGURE 4** Agent D has no information (s/he trusts all the others equally)

We use the lattice trust values from Figure 2 when agent D has information about her network. Because these are probabilities, and by the fact that 'agent D trusts agent B' and 'agent D trusts agent C' are independent events (we do not consider third party influences here), we define the operators $\vee$ and $\wedge$ as follows:

$$x \vee y = x + y - x \cdot y \ , \quad x \wedge y = x \cdot y \ .$$

Then, the lattice for the first case in Figure 2 is based on the following relation:

$$\varnothing \le A, B \le AB \le C, E \le AC, AE, BC, BE \le ABC, ABE \le CE \le ACE, BCE \le ABCE,$$

and for the second case in Figure 2 on:

$$\varnothing \le A, B \le AB \le E \le AE, BE \le C \le ABE \le AC, BC \le ABC \le CE \le ACE, BCE \le ABCE$$
(see Figure 5).

$$\varnothing \longrightarrow A, B \longrightarrow AB \longrightarrow C, E \longrightarrow AC, AE, BC, BE \longrightarrow ABC, ABE \longrightarrow CE \longrightarrow ACE, BCE \longrightarrow ABCE$$

$$\varnothing \longrightarrow A, B \longrightarrow AB \longrightarrow E \longrightarrow AE, BE \longrightarrow C \longrightarrow ABE \longrightarrow AC, BC \longrightarrow ABC \longrightarrow CE \longrightarrow ACE, BCE \longrightarrow ABCE$$

**FIGURE 5** Applying trust lattice to agent D's preferences, as in cases from FIGURE 2

We have shown how lattices of trust are constructed at two simple levels, as in Figure 2 and Figure 5 (and we suggested an alternative: Figure 3), and we make use of them in our model, to study the games described in the "Games and Trust" section (above).

## SUMMARY AND CONCLUSIONS

To study the development and idealization of trust inside small communities, we have modeled the strategic situation as an *n*-person game with two types of goods: religious and ethnic artifacts. Agents have two types of choice: the choice whether to accept or reject the transaction, and the choice of whether or not to enforce or not the trust in partnership according to partner decisions. We use FCA to describe the agents' preferences and also to study the meanings of trust and cultural markers inside a five-agent network. The agents' dispositions in the network are shown in Figure 6 (and are related to our examples in the previous section).

The first question we faced is rather philosophical: can the different meanings of trust be formalized together? Recall that the different meanings of trust are situated at different ontological levels (including idealized trust). The second question has social connotations: does a formalized model capture the richness of a social network which has complex relationships and is subject to change over time?

Our current conclusion is that different meanings of trust may be able to be integrated, by using geometrical meanings represented by layers of lattices that capture different levels of trust: trust as a subject, a relation, or a predication. This would mean philosophically accepting Peirce's categorization (see Peirce, 1868; Ketner 1986). We capture here the transition from trust as relationship in trade networks, where agents are the objects, to a more formal concept of trust as an object having different meanings as attributes. Figure 9 suggests such a geometrical meaning.

The answer to the second question is still under investigation: our model currently considers only trust as a state, trust as relationship, trust as reputation, and each of their relations to idealized trust.  Exploration of trust as constitutive expectations (Garfinkel 1963) remains as an issue to be addressed in future work.

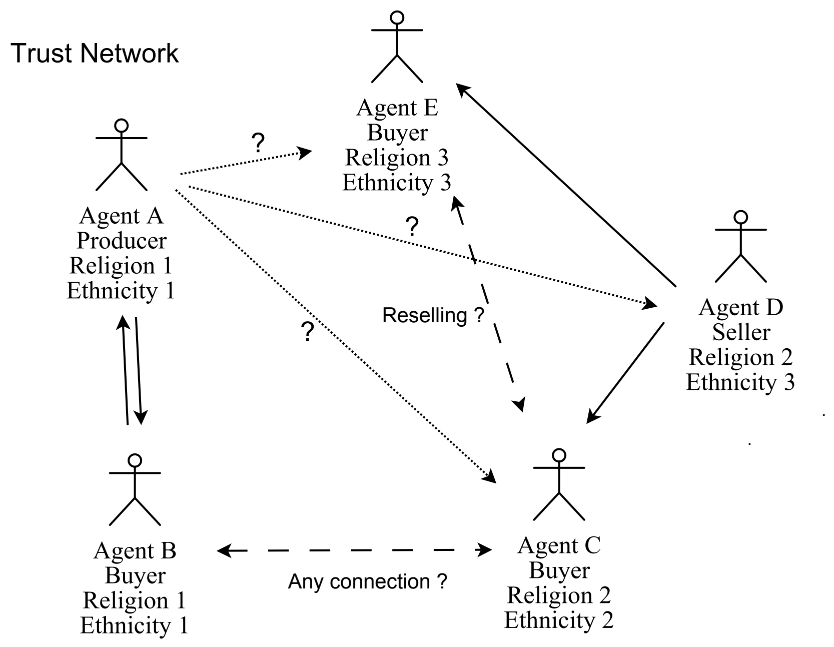


**FIGURE 6** Social agents' network and trades

To amplify Figure 9, we introduce the following preferences for agent D (the agent is interested in selling religious artifacts of type 2 and ethnic artifacts of type 3, the types according to her religion and ethnicity), and we will order them in a lattice (we recognize that agent D's religion prohibits her from selling other religious types of artifacts):

**TABLE 3** Agent D preferences for the religious and ethnic artifacts

| | Religious | Ethnic | Cheap | Expensive | Preferred | Acceptable | Prohibited |
|---|---|---|---|---|---|---|---|
| Artifact 1 | X | | | X | | | X |
| Artifact 2 | X | | X | | X | | |
| Artifact 3 | X | | | X | | | X |
| Artifact 4 | | X | | X | | X | |
| Artifact 5 | | X | | X | | X | |
| Artifact 6 | | X | X | | X | | |

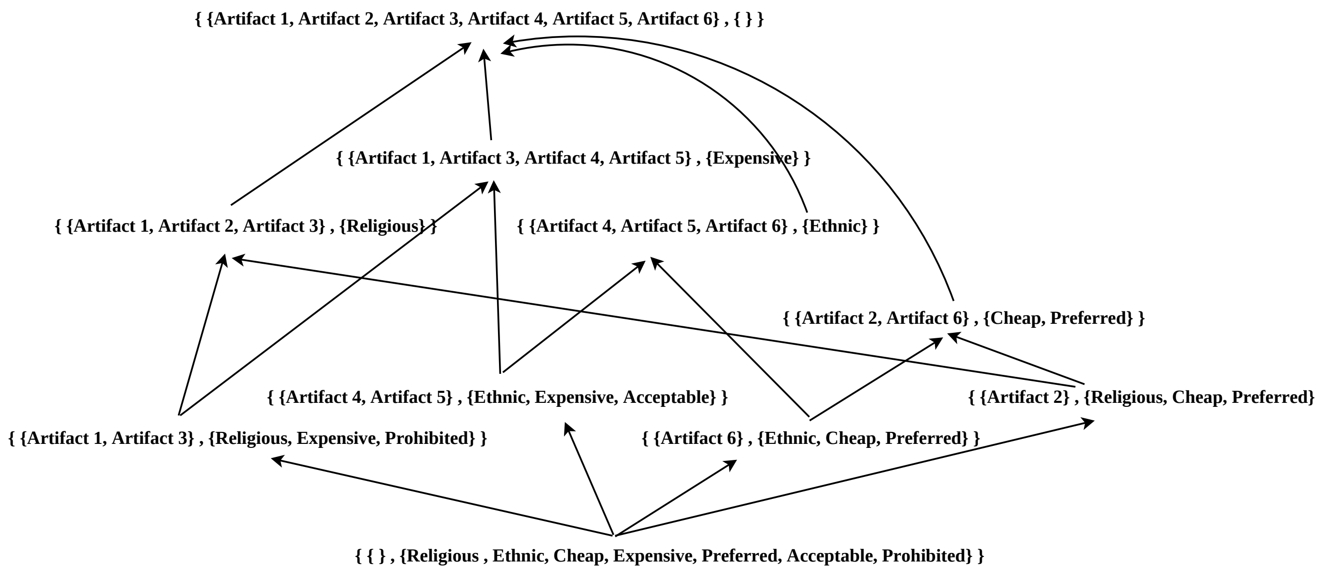

{ {Artifact 1, Artifact 2, Artifact 3, Artifact 4, Artifact 5, Artifact 6} , { } }
{ {Artifact 1, Artifact 3, Artifact 4, Artifact 5} , {Expensive} }
{ {Artifact 1, Artifact 2, Artifact 3} , {Religious} }
{ {Artifact 4, Artifact 5, Artifact 6} , {Ethnic} }
{ {Artifact 2, Artifact 6} , {Cheap, Preferred} }
{ {Artifact 4, Artifact 5} , {Ethnic, Expensive, Acceptable} }
{ {Artifact 2} , {Religious, Cheap, Preferred}
{ {Artifact 1, Artifact 3} , {Religious, Expensive, Prohibited} }
{ {Artifact 6} , {Ethnic, Cheap, Preferred} }
{ { } , {Religious , Ethnic, Cheap, Expensive, Preferred, Acceptable, Prohibited} }

**FIGURE 7** FCA for TABLE 3

This FCA diagram is used for agent D's preferences in the games, when agent D's knows the possibilities that s/he has on the market.

We can be more precise about agent D's trust network (as a refinement of the first case in Figure 5) after a few rounds. Agent D partially knows the trust values of the other agents with respect to agent D's artifacts (see Table 4).

**TABLE 4** Agent D's knowledge of her trade network after a few rounds

| | Producer | Seller | Buyer | Same religion | Same ethnicity | Very Trustworthy | Trustworthy | Not trusted | Unknown |
|---|---|---|---|---|---|---|---|---|---|
| Agent A | X | | | | | | | | X |
| Agent B | | | X | | | | | | X |
| Agent C | | | X | X | | | X | | |
| Agent E | | | X | | X | | X | | |

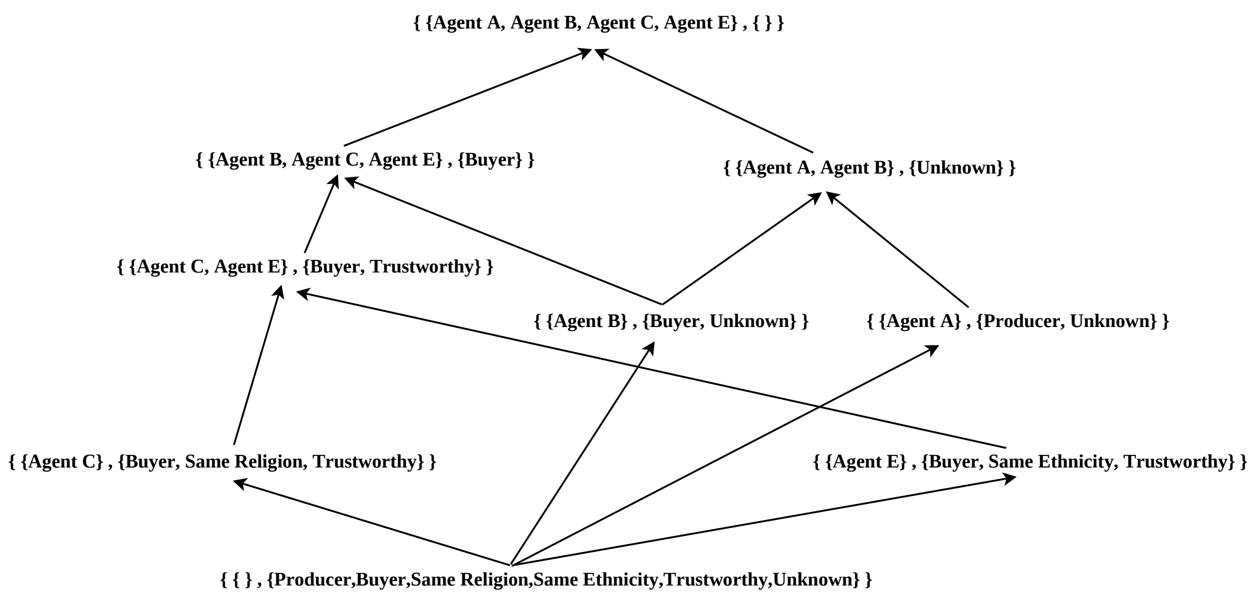

{ {Agent A, Agent B, Agent C, Agent E} , { } }
{ {Agent B, Agent C, Agent E} , {Buyer} }
{ {Agent A, Agent B} , {Unknown} }
{ {Agent C, Agent E} , {Buyer, Trustworthy} }
{ {Agent B} , {Buyer, Unknown} }
{ {Agent A} , {Producer, Unknown} }
{ {Agent C} , {Buyer, Same Religion, Trustworthy} }
{ {Agent E} , {Buyer, Same Ethnicity, Trustworthy} }
{ { } , {Producer,Buyer,Same Religion,Same Ethnicity,Trustworthy,Unknown} }

**FIGURE 8** FCA for TABLE 4

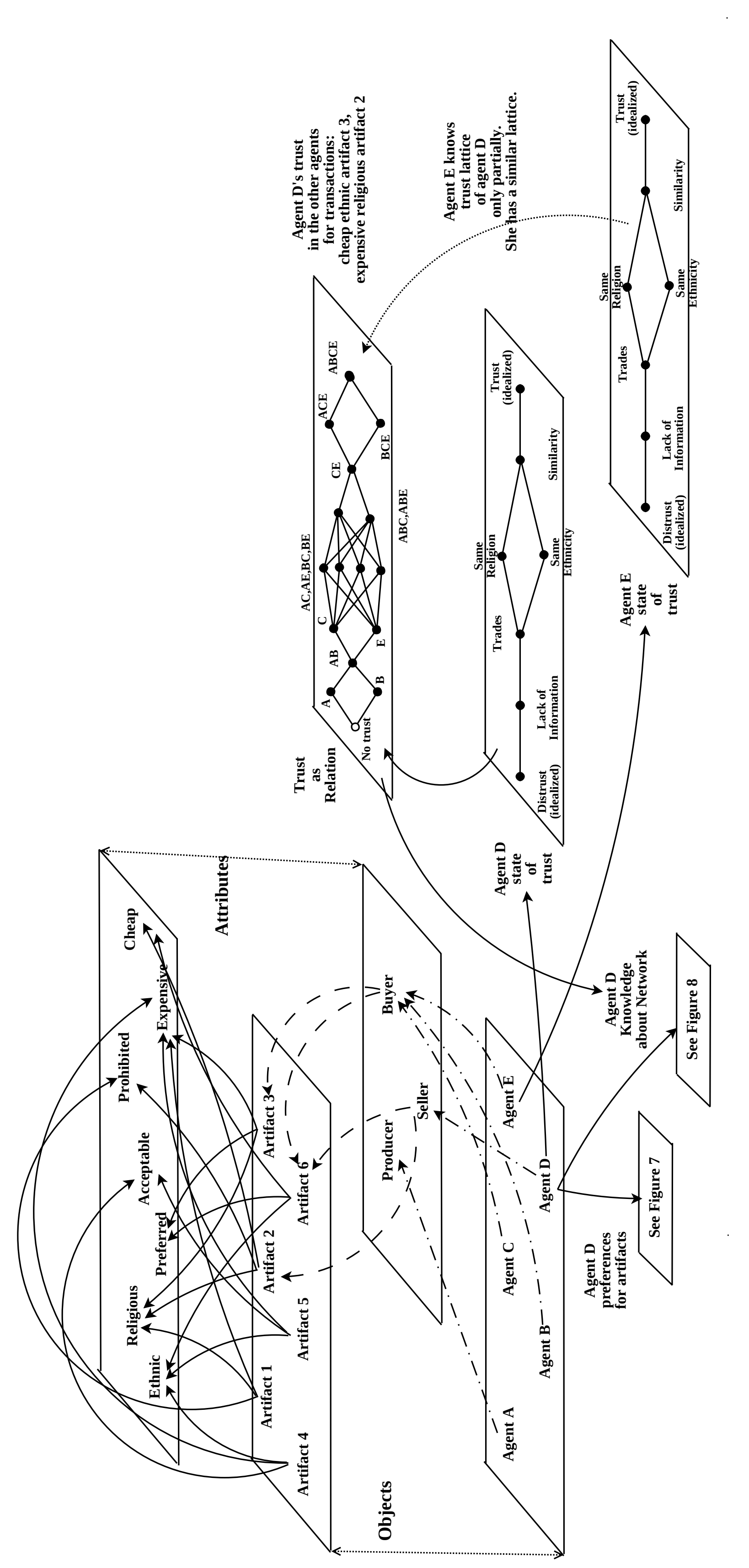


**FIGURE 9** An example of FCA for our network

## ACKNOWLEDGMENT

This work is sponsored by the U.S. Department of Energy, Office of Science, under contract W-31-109-Eng-38.

## REFERENCES

Ben-Ner, A. and L. Putterman, 2003, "Trust in the New Economy" in *New Economy Handbook*, edited by D.C. Jones. San Diego, California: Academic Press.

Codd, E.F., 1970, "A Relational Model of Data for Large Shared Data Banks," *Communications of the ACM* 13:377–387.

Ganter, B., and R. Wille, 1999, *Formal Concept Analysis*. Berlin: Springer.

Garfinkel, Harold. 1963. "A Concept of, and Experiments with, "Trust" as a Condition of Stable Concerted Actions." Pp. 187-238 in O.J. Harvey, Ed., *Motivation and Social Interaction: Cognitive Determinants.* New York: Ronald Press.

Jones, A.J.I., 2002, "On the Concept of Trust," *Journal of Decision Support Systems* **33** (3):225–232.

Joslyn, C., and S. Mniszewski, 2002, "Relational Analytical Tools: VisTool and Formal Concept Analysis," in *LANL Technical Report 02-7697*, Los Alamos National Laboratory.

Ketner, K.L., 1986, "Peirce's 'Most Lucid and Interesting Paper': An Introduction to Cenopythagoreanism," *International Philosophical Quarterly* **26** (4):375–392.

Peirce, C.S., 1868, "On a New List of Categories," Proceedings of the American Academy of Arts and Sciences 7:287-98.

Rubinstein, A., 1998, *Modeling Bounded Rationality*, Cambridge, Massachusetts, London, England: The MIT Press.

Victor, P., C. Cornelis, M. De Cock, and P. Pinheiro da Silva, 2006, "Towards a Provenance-Preserving Trust Model in Agent Networks," in *Proceedings of the WWW'06 Workshop on Models of Trust for the Web,* May 22-26, 2006, at Edinburgh, Scotland.

# COMPLEX MULTIGAMES: TOWARD AN ECOLOGY OF INTERACTION ARTIFACTS

D.L. SALLACH,* Argonne National Laboratory, Argonne, IL,
and The University of Chicago, Chicago, IL

## ABSTRACT

Interaction is increasingly viewed as a basic foundation of computer science. While this view has important implications for various distributed, pervasive, and immersive systems, it is also significant for the design and representation of agent simulation and the modeling of social processes.

During the twentieth century, social interaction became increasingly recognized as a foundation underlying social systems of widely varying scales. The parallel foundational role of interaction in computer and social systems has potentially important implications for both. The diversity and fluidity of social interaction might be considered to provide a design horizon and possible source of insight for computational interaction. At the same time, the progressive definition of interaction formalisms in computer science has the potential to contribute to social modeling. These two processes converge in the area of social agent interaction.

Software based on multi-agent interaction has begun to identify what might be called "interaction artifacts" in the areas of communication and coordination. However, multi-agent software has contributed relatively little in the area of adversarial interaction. The latter has been the focus of game theory; however, game-theoretic preference for analytical solutions has resulted in the use of highly stylized games. This paper focuses on the need to develop artifacts that formalize high-dimensional games in rich social ecologies for social agent simulation.

**Keywords:** Interaction, interaction artifacts, ecology of games

## INTRODUCTION

In the analysis of the foundations of computer science, focus has shifted from algorithms to interaction (Milner 1993; Wegner 1997; Wegner 1998; Wegner and Goldin 1999). Underlying this shift are issues such as: 1) the inability to express concurrency with sequential algorithms, 2) the artificial constraints of discrete time steps, and 3) the interruptive nature of simultaneously competing processes. In this context, the dialogue among the computational and social sciences appears to acquire ever more potential.

However, if the need to provide a formal basis for interaction is evident, successfully accomplishing this objective remains challenging. A formalization of open-ended processes helps (Aczel 1988; Wegner and Goldin 1999), the nature of interaction is significantly shaped by

---

* *Corresponding author address*: David L. Sallach, Decision and Information Sciences Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, IL 60439-4832; email: sallach@anl.gov.

the social context, including the capability and motive structure of the participants. Domain-specific social modeling has the potential to assist in exploring these issues and developing effective interaction artifacts.

## INTERACTION ARTIFACTS

Among recent multi-agent innovations are the development of computational fields (Mamei and Zambonelli 2004; Nagpal and Mamei 2004) and coordination artifacts (Omicini et al. 2004). While such innovations may be natural extensions of distributed software mechanisms, such as blackboards (Corkill 1991) and Linda parallelism (Leichter and Whiteside 1986), their use in social roles suggests that sociological concepts and theories may be increasingly relevant to their future evolution.

Multi-agent uses of interaction artifacts, such as tuple centres (Omicini and Denti 2001) and agent coordination contexts (Omicini 2002), generally emphasize the effective coordination of distributed processes. However, for social simulation applications, it is not sufficient to reduce all forms of interaction to coordination. On the contrary, coordination is interleaved with a variety of adversarial processes. Accordingly, to generalize the useful concept of "coordination artifact" to a broader concept of "interaction artifact," the latter will need to incorporate a variety of adversarial dynamics, as well as communication and coordination.

In social simulation, the purpose of any such artifacts is to control the computational complexity of the potential interaction stream while, at the same time, providing a capability with which to express a range of interactions that is sufficiently rich to support credible social dynamics. The nature of the range may vary, of course, based on the domain in question.

## COMPLEX MULTIGAMES

To extend artifacts from coordination to social interaction, adversarial processes must be incorporated. The formalisms most frequently associated with adversarial interaction are derived from game theory. However, the traditional game-theoretic focus on analytical solutions has correspondingly tended to emphasize formalisms that are relatively simple and static. To be of theoretical and practical use in social simulation, game-theoretic models must be more intricate and richly dynamic. The resulting models should be elegant and guided by social theory, but inevitably they will be considerably more complex. The necessary complexities are a major focus of this paper.

Human actors play multiple games, select from among available games, shift from one to another, misunderstand what games their counterparts are playing, and act in ways that are (more or less) effective simultaneously within multiple games, etc. Therefore, a game artifact should be able to support simultaneous agent interaction via multiple interwoven games. An early qualitative study (Long 1956) describes the concept:

> [A] local community can be usefully conceptualized as an ecology of games. In the territorial system a variety of games goes on: banking, newspaper publishing, contracting, manufacturing, etc. The games give structure, goals, roles, strategies, tactics, and publics. Players in each game make use of players in the others for

> their particular purposes . . . The interaction of the games produces unintended but systematically functional results for the ecology.

Long focuses primarily on industry and professional roles but, as described below, it may be more useful to conceive of the skein of games with greater abstraction. Specifically, for present purposes, games are classified into three categories: altruistic, economic, and coercive.

## Types of characteristic games

Three types of reciprocal games, each with their own benefits and relevant resources, provide a starting point of a prospective ecology of games. Altruistic games involve a kind of mutual nurturance that is seen in families and tribes, among neighbors, and within communities. Types of nurturance in altruistic games may vary, but accounting is not strict and the games tend to be mutually reinforcing over time, commonly resulting in virtuous spirals (Carse 1986). Economic games are familiar; they involve complementary benefit and relative advantage, and arm's-length accounting, and they are often self-reinforcing (Osborne and Rubinstein 1990). Coercive games involve the exchange (or threat) of force or violence. Reciprocity is frequently anticipatory, and comparative accounting tends to be exaggerated, resulting in vicious spirals. Complementarity usually takes the form of innovative tactics or novel defenses. When a recurrent and balanced interchange emerges, any of the three types of games can become relatively stable over time.

In addition to reciprocal interchanges, where the details primarily concern frequency, quantity, and/or quantitative conversion, there are off-diagonal interchanges in which one resource is used to acquire or respond to a different one. Actors can use force to gain material benefits, for example, or money to gain affinity benefits. Therefore, each game resource requires two (directional) dimensions. It appears that such games are less likely to stabilize than are the reciprocal games, but they are nonetheless interesting because of their familiarity.

The client of a courtesan, for example, can be regarded as using economic resources to acquire intimate nurturance, while one providing such services is generating economic resources. Armed robbery is an example of using coercive means to acquire economic resources. Mercenary armies and hitmen are examples of their employers using economic means to acquire coercive resources. Revolutionaries and terrorists who provide food, welfare, and health services use economic and/or support services for the purpose of receiving assistance in making or hiding coercive acts. Finally, instances of the abduction of children to raise them (as opposed to demanding ransom) can be viewed as the use of coercive means to acquire nurturing or pseudo-nurturing services.

Off-diagonal games also figure in situations in which there is ambiguity or deception concerning which games are being played. Contributions to charities (nurturance) may be deceptively converted for terrorists' use (coercion). The sale of contraband can be an economic means of achieving the same result. Examples abound. Indeed, determining the game that one is being invited (or enticed) to play is an important human capability. Computational models will need to be able to endogenously assemble and parameterize particular games from components (Sallach 2000).

A full-featured game artifact will require communication and coordination capabilities. For present purposes, an artifact that combines: 1) communication, 2) coordination, and 3) moves within a game will be regarded as a general purpose interaction artifact as defined (and designed) in the present paper.

As described above, an interaction artifact will be built around games based on the reciprocal exchange of three types of resources: altruistic, economic, and coercive. The three resources form a diagonal on a 3x3 grid, with altruism in the upper left and coercion in the lower right (Figure 1). Altruistic games involve mutual support where close accounting is not maintained. Economic games involve the familiar arm's-length, rationally balanced interchange of materials, products, and/or services. Coercive, or existential, games (involving force) involve an interchange of threats and/or life-threatening attacks.

This framework necessarily incorporates a communication mechanism so that requests and negotiation can be integrated into the prospective games. For simplicity, the communication framework at this level is based on agent information that is as accurate as possible.

## THE MULTIGAME HORIZON

The dimensions that define game resources do not exhaust the possible variations. Given the same set of games, another level would involve the deliberate distortion of information (misrepresentation for instrumental benefit). Supported capabilities would necessarily include distortion detection. A third level might add collective action as meta-strategy within the same cluster of available games. A number of additional levels are possible and sometimes necessary (Figure 2).

| | **Altruistic** | **Economic** | **Coercive** |
|---|---|---|---|
| **Altruistic** | **Reciprocity of nurturance** | **Nurturing means to acquire economic resources** | **Nurturing means to acquire coercive resources** |
| **Economic** | **Economic means to acquire nurturing resources** | **Reciprocity of goods, services** | **Economic means to acquire coercive resources** |
| **Coercive** | **Coercive means to acquire nurturing resources** | **Coercive means to acquire economic resources** | **Reciprocity of force** |

**FIGURE 1** An ecology of multigames

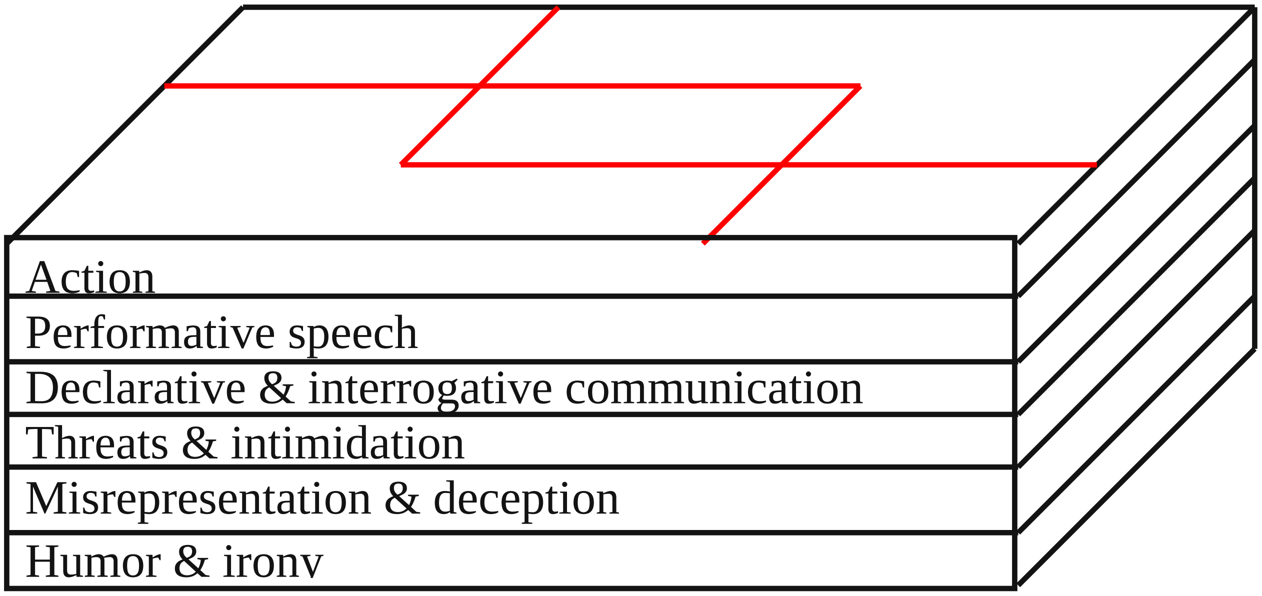

Action
Performative speech
Declarative & interrogative communication
Threats & intimidation
Misrepresentation & deception
Humor & irony

**FIGURE 2** Multigame levels

The multigame grid is a discrete structure. Transforming this determinant framework into a radial geometry (Sallach 2003; Sallach and Mellarkod 2004) will allow the introduction of fine-grain game variants and, thereby, more effectively model the rich calibration of naturally-occurring human choice.

The focus of interaction artifact design combines computational design and social expressiveness. A discourse between the computational and social sciences is necessary and potentially highly productive.

## ACKNOWLEDGMENT

This work is sponsored by the U.S. Department of Energy, Office of Science, under contract W-31-109-Eng-38.

## REFERENCES

Aczel, P., 1988, *Non-well-founded Sets*, Stanford, CA: Center for the Study of Language and Information.

Carse, J.P., 1986, *Finite and Infinite Games*, New York: Ballantine.

Corkill, D.D., 1991, "Blackboard Systems," *AI Expert* **6**:40–47.

Leichter, J.S. and R.A. Whiteside, 1986, "Implementing Linda for Distributed and Parallel Processing," in *Third International Conference on Supercomputing* (41–49), Crete, Greece.

Long, N.E., 1956, “The Local Community as an Ecology of Games,” *American Journal of Sociology* **64**:251–261.

Mamei, M., and F. Zambonelli, 2004, “Self-maintained Distributed Tuples for Field-based Coordination in Dynamic Networks.” in *ACM Symposium on Applied Computing*. Nicosia, Cyprus.

Milner, R., 1993, “Elements of Interaction,” in *Communications of the ACM* **36**:78–89.

Nagpal, R.. and M. Mamei, 2004, “Engineering Amorphous Computing Systems,” in *Methodologies and Software Engineering for Agent Systems: The Agent-Oriented Software Engineering Handbook*, edited by F. Bergenti, M. Gleizes, and F. Zambonelli. Boston: Kluwer Academic Publishers (303–320).

Omicini, A.,, 2002, “Towards a Notion of Agent Coordination Context” in Process Coordination and Ubiquitous Computing, edited by D.C. Marinescu and C. Lee. Boca Raton, FL: CRC Press (187–200).

Omicini, A., and E. Denti, 2001. “From Tuple Spaces to Tuple Centres,” *Science of Computer Programming* **41**:277–294.

Omicini, A., S. Ossowski, and A. Ricci, 2004, “Coordination Infrastructures in the Engineering of Multiagent Systems,” in *Methodologies and Software Engineering for Agent Systems: The Agent-Oriented Software Engineering Handbook*, edited by F. Bergenti, M. Gleizes, and F. Zambonelli. Boston: Kluwer Academic Publishers (273–296).

Osborne, M.J., and A. Rubinstein, 1990, *Bargaining and Markets*. San Diego: Academic Press.

Sallach, D.L., 2000, “Games Social Agents Play: A Complex Form,” in *Joint Conference on Mathematical Sociology in Japan and America*. Honolulu, HI.

—, 2003, “Interpretive Agents: Identifying Principles, Designing Mechanisms” in *Proceedings of the Agent 2003 Conference on Challenges in Social Simulation*, edited by C. Macal, M. North, and D. Sallach. Argonne: Argonne National Laboratory (345–353).

Sallach, D.L., and V.S. Mellarkod, 2004, “Prototype Inference and Social Interaction,” in *Proceedings of the Agent 2004 Conference on Social Dynamics: Interaction, Reflexivity and Emergence*. Argonne: Argonne National Laboratory.

Wegner, P., 1997, “Why Interaction Is More Powerful Than Algorithms,” *Communications of the ACM* **40**:80–91.

—, 1998, “Interactive Foundations of Computing,” *Theoretical Computer Science* **192**:315–351.

Wegner, P., and D. Goldin, 1999,“Mathematical Models of Interactive Computing,” *Brown Technical Report CS* **99-13**.

# Modeling Organizational Processes

# AN AGENT-BASED APPROACH TO EXAMINE THE NETWORK KNOWLEDGE ADVANTAGE IN OPEN INNOVATION NETWORKS: FIRM OPENNESS AND INTERORGANIZATIONAL NETWORK PERFORMANCE

J.A. VILLARROEL *, Ecole Polytechnique Fédérale de Lausanne,
University of Lausanne, Switzerland &
University of Pennsylvania, USA

J.E. TAYLOR, The University of Texas at Austin, USA

## ABSTRACT

In this paper we present the results of an agent-based model of open innovation, which suggest the ***network knowledge advantage*** underlying open interfirm networks of innovation, as opposed to an advantage arising from the interfirm network structure. To date, open innovation research has primarily been developed from firm-level case studies. Innovation network researchers have focused on cross-sectional studies of the strategic advantages of network structure and firm positioning as determinants of performance. However, the dynamics of learning and the accumulation of knowledge in open interorganizational networks of innovation and their effects on firm and system-level performance remain relatively unexplored, particularly given the impact of resource interdependencies that exist at the boundaries between firms. To address this important question, we explore the impact that open flows of knowledge have on productivity performance, given that firms must address the impact of interdependencies at the boundaries of such exchanges. In previous research we found that interdependencies can have a negative impact on productivity performance in relationally unstable networks. In the current model we find that a greater degree of firm openness towards interfirm exchange of knowledge has a positive impact on performance, and that even a small degree of firm openness towards building network level codified knowledge has a significant positive impact on performance. In conclusion, firm openness compensates for learning discontinuities that occur when network affiliations are loosely coupled. These findings on network-level knowledge dynamics have important implications for innovation theory and for the strategic management of collaborative innovation initiatives.

**Keywords**: open innovation networks, network knowledge advantage, interorganizational learning, knowledge transfer, relational instability

**Acknowledgement**: The authors would like to thank Prof. Paul David and Prof. Steve Klepper for their enlightening comments during the informal discussions we had regarding our on-going research work at the 13th Conference for Competitiveness and Cooperation (CCC) held in Switzerland.

* *Corresponding author*: J. Andrei Villarroel, Ecole Polytechnique Fédérale de Lausanne, ODY1.19 – Station 5, CH-1015 Lausanne, Switzerland ; email: andrei.villarroel@epfl.ch

## INTRODUCTION

To date, open innovation research has primarily been developed from firm-level case studies, suggesting the strategic advantages of firm openness regarding the creation and exchange of intellectual property (Chesbrough 2003a, 2003b, 2003c; Chesbrough and Crowther 2006, Christensen et al 2005). Parallel to this, innovation network researchers have focused on the strategic advantages of network structure and firm positioning as determinants of innovation performance (Gulati et al 2000, Uzzi and Spiro 2005; Schilling and Phelps 2005). At the same time, interorganizational research unveiled the complexities of dealing with interdependencies and relational instability in networks (Staudenmayer, Tripsas, and Tucci 2005; Taylor, Levitt, and Villarroel 2006).

Nonetheless, the dynamics of learning and the accumulation of knowledge in open interorganizational networks of innovation and their *effects* on firm and system-level performance remain relatively unexplored, particularly in the presence of interdependencies at the boundaries between firms participating in relationally unstable interorganizational networks. As open innovation networks arise in the biotechnology industry –e.g. BIOS BioForge–, following the example of the software industry –e.g. IBM Open Invention Network–, creating knowledge commons of patents and other intellectual property, the question of the implications of a hypothesized *network knowledge advantage* in these loosely coupled innovation networks is of importance both theoretically as well as for practice.

To this end, we develop a dynamic model of a loose network of collaborating firms implementing open innovation policies, and study its productivity performance characteristics across time. Two forms of open innovation policies are considered: i) interfirm openness and ii) network openness.

## THEORETICAL BACKGROUND

### Innovation, Interaction and Open Innovation

In recent years, an interactive model of the innovation process emerged in the literature, whereby innovators tap more and more often from the interactions with their lead users (von Hippel 1988), their suppliers, and with a number of other actors in their innovation ecosystem (Lundvall 1992; Brown and Eisenhardt 1995). Essentially, the view provided by these models is that innovators rarely innovate alone and that they are likely to group together in an interorganizational network of interactions (Cook and Brown 1999; Brown and Duguid 2000).

Extending this research, Chesbrough (2003a, 2003b, 2003c) introduced the *open innovation* model, where he emphasized the interactive and distributed nature of innovation of the previous models. He described innovation as an open process; a contrast to closed models of internal R&D (Mowery 1983; Chandler 1990). In his model, Chesbrough argues that firms operating in an open innovation environment exploit external ideas by developing linkages to the market both from outside and from inside the firm. This ongoing process of opening up the firm to exchange knowledge with its environment is redefining the boundaries of the firm, hence making it more permeable and more dependent on its environment.

## Innovation Networks and Firm Performance

As a simultaneous development in the literature, management researchers have shown renewed interest in the study of firm networks (Gulati et al 2000; Kogut 2000) and their impact on innovation (Hargadon 2003; Cowan and Jonard 2003, 2004; Uzzi and Spiro 2005; Malerba 2006). It is in this sense that the seminal work on small world networks by physicists Watts and Strogatz (1998) -hypothesized to enhance innovative activity (Watts 2003, 2004)- has received much attention from scholars in the management field (Baum et al 2003; Verspagen and Duysters 2003; Schilling and Phelps 2005).

By the same token, biotechnology collaboration networks have been an important source of empirical evidence for students of innovation networks, who argued that the performance of such collaborative innovation processes lie in their social capital structure (Shan, Walker and Kogut 1994; Powell, Koput, and Smith-Doerr 1996). However, most empirical studies of interorganizational networks are cross-sectional, and miss on the fact that interorganizational innovation networks tend to be dynamic, as evidenced by a recent longitudinal study of the biotechnology industry performed by Powell et al (2005). Powell and his colleagues state:

*"Biotechnology is characterized by a high-rate of formation and dissolution of linkages. Connections are often forged with a specific goal in mind, such as taking a company public or selling and distributing a new medicine. Once the task is completed, the relationship is ended and successful collaborators depart gracefully… Moreover, many of the participants in the field are 'multivocal', that is, they are capable of performing multiple activities with a variety of constituents." (2005: 1138).*

In this sense, stable network relationship structures are not at the center of differences in innovation performance, rather, those performance differences must arise during the brief, goal-specific, collaborations between individual firms, which have been little researched as of yet. It is our aim to contribute to fill this gap in the literature, with a model exploring how openness towards knowledge exchange during collaboration projects may have a positive impact on performance, irrespective of favorable conditions provided by particular stable network structures addressed in previous studies.

## Interorganizational Learning and Network Knowledge

Penrose (1959), Barney (1986), and Grant (1996) discuss how firms can be seen as assets comprising tangible and intangible resources, and by extension, to comprise bundles of explicit and tacit knowledge (Polanyi 1966). Consistent with the literature on organizational learning, we consider learning across organizations as experiential (Herriott, Levinthal, and March1985; Levitt and March 1988). In this sense, learning affects change in organizational knowledge and interorganizational knowledge as a result from organizations gaining experience when working together in collaborative projects.

Powell and Brantley (1992), and Powell et al (1996), were among the first to provide empirical evidence that the locus of innovation could be found in the network, as opposed to in individual firms, and to argue for a network approach to organizational learning. However, researchers have argued that loose coupling and interdependencies in inter-firm networks can be a liability to learning and, hence, performance (Taylor, Levitt and Villarroel 2006). The question of how open policies towards serendipitous knowledge exchanges (interfirm openness), and codified knowledge sharing (network openness) impact learning discontinuities associated with loosely coupled interfirm networks are not addressed in the current literature.

## MODEL

In the present model, we extend upon the work of Taylor, Levitt and Villarroel (2006) to address the important research question about the *network knowledge advantage* underlying open innovation networks, by exploring the impact of interfirm openness and network openness on system-level performance. Our work takes an interorganizational learning approach (Argote 1999; Argote and Ingram 2000; Lane and Lubatkin 1998; Wright 1936; Yelle 1979) to measure productivity performance differences in networks of collaborating firms.

Embedded in these learning networks, knowledge becomes an object to be exchanged: i) at the boundary between firm dyads for the edification of one firm, which in the context of open innovation is reciprocated by a mutual knowledge exchange, and ii) at the boundary between a firm and the network for the edification of the network as a whole, which in the context of open innovation is aimed at building a knowledge base for the common benefit of all active network constituents.

In agreement with the empirical evidence found in Powell et al (2005), our agent-based model represents a loose network of learning firms, dealing with interdependencies at the boundaries of their knowledge exchanges (Staudenmayer, Tripsas, and Tucci 2005; Taylor, Levitt, and Villarroel 2006) and implementing two distinct open innovation policies: i) interfirm openness, and ii) network openness.

### Model Description

The base model is constituted of a total of N firms {1, …, $i$, …, N}, modeled as individual agents with learning capabilities. In the base model, learning occurs at the individual level as well as the level of the firm dyad. A given firm $i$, implements a specialized role $R_i$. Several firms may belong to the same role, and there are a total of M roles {$R_i$, $R_j$, …, $R_M$}. The execution of a project P, however, requires only one specialized firm from each role, for a total of M firms in a project (M<<N). Firms are randomly chosen to work on a project.

Our model borrows from the general form of the learning curve described in Yelle (1979), which is of the same form as general progress functions treated in Dutton and Thomas (1984), as well as that used by Levitt and March (1988) in explaining organizational learning.

Given a firm $i$, its individual productivity is:

$$\Pi_i = \Pi_0 \cdot (n_i)^{L_i} \tag{1.1}$$

| , where: | | |
|---|---|---|
| | $\Pi_0$ | = initial productivity factor for individual work |
| | $\Pi_i$ | = productivity factor for firm *i* |
| | $n_i$ | = number of individual tasks executed by firm *i* |
| | $L_i$ | = $\log_2 \lambda_i$ = characteristic learning index for firm *i* |
| | $\lambda_i$ | = learning rate |
| | $1 - \lambda_i$ | = progress ratio in the log-linear learning curve model |

In this sense, given a characteristic time $T_{Ri}$ for the execution of the first independent task by a firm *i* in role $R_i$, the actual execution time is given by:

$$T_i = \Pi_i \cdot T_{Ri} \tag{1.2}$$

In addition, given two firms (i,j) that interact with one another, we associate a productivity factor, $\Pi_{ij}$ , to the collaborative interdependent work developed by the two firms, with a dyadic learning curve as follows:

$$\Pi_{ij} = \Pi_{00} \cdot \left(n_{ij}\right)^{L_{ij}} \tag{1.3}$$

| , where: | | |
|---|---|---|
| | $\Pi_{00}$ | = initial productivity factor for collaborative work |
| | $\Pi_{ij}$ | = productivity factor for the collaborating firms (i,j) |
| | $n_{ij}$ | = number of interdependent tasks executed by firms (i,j) |
| | $L_{ij}$ | = $\log_2 \lambda_{ij}$ = learning index between firms (i,j) |
| | $\lambda_{ij}$ | = learning rate between firms (i,j) |

Similarly to the single firm case, given a level of collaborative interdependent work for the firm dyad (i,j), $X_{ij}$ , defined as a percentage of firm *i*'s characteristic execution time $T_{Ri}$, the actual one-sided execution time of this collaborative portion of the work is given by:

$$T_{ij} = \Pi_{ij} \cdot X_{ij} \cdot T_{Ri} \tag{1.4}$$

Finally, the total execution time of the project, $T_P$, results from the sum of all the independent tasks executed by each individual firm as given in (1.2), and all the collaborative interdependent tasks executed by all firms involved in the project as given in (1.4). This is:

$$T_P = \sum_{i}^{M} T_i + \sum_{i}^{M} \sum_{j \neq i}^{M} T_{ij} \quad , \ \forall i, j \in P \tag{1.5}$$

Note: the reason why the collaborative portion of the work is counted on both sides of the dyadic collaboration, or, on all sides of the network collaboration for that matter, is because labor hours are typically accounted for individually for all firms involved in the project, although the work may be performed in parallel.

## *Model Benchmarking*

In order to be able to compare the performance of the different models presented, given that each introduces new characteristics that affect the absolute execution times for projects, we have normalized their initial durations by dividing over the total characteristic execution time of a project. The normalizing factor is:

$$T_N = \sum_{i}^{M} T_{Ri} + \sum_{i}^{M} \sum_{j \neq i}^{M} X_{ij} T_{Ri} \qquad (1.6)$$

Therefore, the normalized performance of any given project is given by:

$$\overline{T_P} = \frac{T_P}{T_N} \qquad (1.7)$$

Following this normalization, the performance rates for all models can then be compared for relative differences as they evolve through the execution of projects.

### *Experimental Design*

There are two models discussed in the present work, one baseline model, and a newly introduced model, NKA (Network Knowledge Advantage). The baseline model refers to the work by Taylor, Levitt, and Villarroel (2006)[†], first presented at the NAACSOS'06 conference. This baseline model will be referred to here as the TIRI (Task Interdependence and Relationship Instability) Model, used for comparison and discussion of the new model. The new model, NKA, exploring open innovation, comprises two sub-parts. The first part referred to as Model 1, examining the effects of Interfirm Openness, and the second part, Model 2, exploring the implications of Network Openness. All these models are summarized in Table 1.

**TABLE 1** The experimental design comprises four types of simulation runs discussed in this article

| | Closed Innovation | Open Innovation | |
|---|---|---|---|
| Simulation Types | TIRI Model (Baseline) | NKA Model 1 Interfirm Openness | NKA Model 2 Network Openness |
| Independent Firms | • No interdependent tasks<br>• No interactive knowledge exchange | - | • Open codified knowledge exchange of all experiences via network commons |
| Interdependent Firms | • Interdependent tasks<br>• Interactive learning from one another | • Open interactive tacit knowledge exchange of third-party experiences | - |

## Model 1 : Interfirm openness – Interorganizational Learning by Contagion through Open Collaborative Work

In this model, when inter-role task interdependencies exist[‡], a firm (ego) learns –by contagion through the execution of collaborative interdependent work with another firm– from the experience cumulated by a second firm (alter) in alter's previous interdependent work with other firms in the same role than ego. Ego's learning by contagion occurs in addition to the dyadic learning that takes place when two interacting firms learn from one another. Because of the necessarily interactive nature of the interorganizational learning occurring in this setting, it can be assimilated to a tacit knowledge exchange.

[†] Taylor, Levitt, and Villarroel (2006), who developed a multi-agent simulation model to explore the insidious role of task interdependence and relational instability in system-level learning.

[‡] The main assumption for this model to have an effect on performance is that there ought to be interdependent work to be executed by the interacting firms.

To account for this tacit knowledge exchange, we have modified the learning equation (1.3) of the collaborative work between two firms (i,j) to account for the knowledge that firm *j* is open to transfer to firm *i* from its experience accumulated with other firms *k* in the same role $R_i$ as firm *i*, as follows:

$$\underset{\substack{\text{Open Innovation} \\ \text{with tacit dyadic knowledge exchange}}}{\Pi_{ij}} = \Pi_{00} \cdot \left( n_{ij} + a_{ij} \cdot \sum_{k \neq i}^{R_i} \alpha_j \cdot n_{jk} \right)^{L_{ij}} \quad , \; i,k \in R_i, \; j \in R_j \tag{1.8}$$

| | | |
|---|---|---|
| , where : | $\alpha_j$ | = degree of openness of firm *j* towards tacit knowledge exchange $0 \leq \alpha_j < 1$ |
| | $a_{ij}$ | = relative absorptive capacity of firm *i* relative to firm *j* |
| | $n_{ij}$ | = number of interdependent projects executed by firms (i,j) |
| | $n_{jk}$ | = number of interdependent projects executed by firms (j,k) |

In this paper, we are interested in exploring the performance effects of tacit knowledge exchange through open active collaboration as an innovation policy. Therefore, for the sake of clarity of the results presented here, and to produce a baseline for current and future research, we have set the degree of openness towards tacit knowledge exchange equal for all firms, namely *α*, as we have the learning index, *L*. Similarly, the relative absorptive capacity $a_{ij}$ between firm dyads has been set to 1 (one) in all cases. Hence, the resulting equation is:

$$\underset{\substack{\text{Open Innovation} \\ \text{with tacit dyadic knowledge exchange}}}{\Pi_{ij}} = \Pi_{00} \cdot \left( n_{ij} + \alpha \cdot \sum_{k \neq i}^{R_i} n_{jk} \right)^{L} \quad , \; i,k \in R_i, \; j \in R_j \tag{1.9}$$

### *Special Cases*

i) If $\alpha = 0$, then we fall back to the simple dyadic learning between two interacting firms.

ii) If $\alpha \rightarrow 1$, then every interaction between firms of different roles accrues indirectly to all firms of those roles, hence yielding a performance curve similar to the baseline of pure task independence case.

iii) If $0 < \alpha << 1$, a realistic open interfirm collaborative innovation policy is in place, and we obtain performance curves which lie between the two extremes of task interdependence without learning by contagion, and pure task independence where learning by contagion cannot occur.

The productivity curves that arise with contagion levels between 0 and 1, reflect learning as it is likely to occur in real-life situations.

## Model 2 : Network Openness - Interorganizational Learning from a Network Innovation Commons

This is an extension to the base model, where learning may occur through the network. The present model implements an interorganizational network's open policy whereby part of the knowledge gained by a network constituent is codified and shared with the community at large by means of an innovation commons. More specifically, firms share part of the learning they gain from executing a project with the rest of the network, and this, irrespective of there being inter-role interdependencies. As a result, although some firms may not be executing upon projects in a given iteration of the simulation, they can, and do, profit directly from the shared (codified) network knowledge that other firms contribute into the innovation commons.

This codified knowledge transfer has an impact on the productivity of a firm, and we have operationalized it by modifying the learning equation (1.1) describing the learning of the individual firm as follows:

$$\Pi_{\substack{i \\ \text{Open Source} \\ \text{with codified network knowledge exchange}}} = \Pi_0 \cdot \left( n_i + a_i \cdot \sum_{j \neq i}^{N} \beta_j \cdot n_j \right)^{L_i} \tag{1.10}$$

| , where: | $\beta_j$ | = degree of openness of firm *j* towards codified knowledge exchange $0 \leq \beta_j < 1$ |
|---|---|---|
| | $a_i$ | = absolute absorptive capacity of firm *i* |
| | $n_i$ | = number of individual tasks executed by firm *i* |
| | $n_j$ | = number of individual tasks executed by firm *j* |

In the model set forth here, $\beta_j$ represents the amount of codified knowledge that firm *j* contributed into the network commons, expressed as the percentage of the experience gained by firm *j* upon executing projects, which had been shared with the network at large.

In the present work, we are interested in exploring the performance effects of codified knowledge sharing through open source as an innovation policy. To make the results presented in this work clearer, and to provide with a baseline for the present, as well as upcoming work, we have set the degree of openness of firm *j* towards sharing codified knowledge equal for all firms, namely $\beta$, as we have the learning index, *L*. Similarly, the absolute absorptive capacity $a_i$ of all firms has been set equal to 1 (one)[§]. Thus, the resulting equation is:

$$\Pi_{\substack{i \\ \text{Open Source} \\ \text{with codified network knowledge exchange}}} = \Pi_0 \cdot \left( n_i + \beta \cdot \sum_{j \neq i}^{N} n_j \right)^{L} \tag{1.11}$$

*Special Cases:*

i) If $\beta = 0$, then we fall back to the model with learning by contagion through open collaborative work.

ii) If $\beta \to 1$, then the entire knowledge accumulated by every firm is made available to all other firms.[**]

iii) If $0 < \beta << 1$, a realistic open source innovation policy is in place, and we obtain a performance improvement for all firms across the board.

[§] Assuming an absolute absorptive capacity of 100%, depending on the circumstances, can be argued to be unrealistic. For example, the source code of a modular object-oriented software program written in Java could be 100% reused into a new Java program. However, the source code of a complex monolithic software program in Fortran, can be difficult to understand and reuse by modern programmers nowadays.

[**] Note that having β tend to 1 is unrealistic, since knowledge cannot be codified entirely (100%). In addition, although firms do seek to codify knowledge, most firms choose not to share all of the codified knowledge they have worked hard to develop.

This network innovation commons and the interorganizational knowledge spillovers that result from them can be characterized by: firms participating in open source projects, firms patenting their inventions, firm researchers publishing their work in academic journals. These are all examples of spillovers of codified knowledge of the firm that are voluntarily put in the public domain for the benefit of all.

## RESULTS

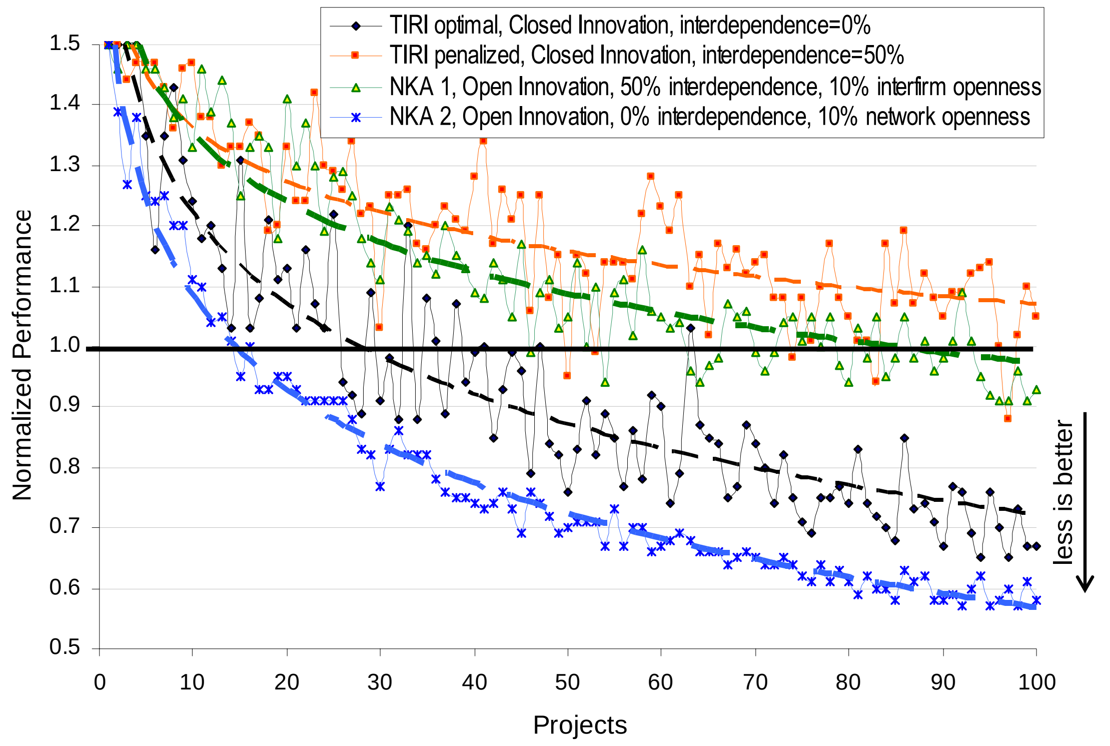


**FIGURE 1** Comparison of 4 scenarios[*] of the performance of a loose network of 40 firms (10 per role) interacting through the execution of collaborative innovation projects (less is better).

- *Black*: TIRI optimal baseline, firms with 0% interdependent task overlap, and closed innovation ($X_{ij}$=X=0 for all firms, $\alpha$=0, $\beta$=0).
- *Orange*: TIRI penalized baseline, firms with 50% interdependent task overlap, and closed innovation ($X$=0.5, $\alpha$=0, $\beta$=0).
- *Green*: NKA Model 1, firms with 50% interdependent task overlap, and $\alpha$=10% interfirm openness policy ($X$=0.5, $\alpha$=0.1, $\beta$=0).
- *Blue*: NKA Model 2, firms with 0% interdependent task overlap, and $\beta$=10% network openness policy ($X$=0, $\alpha$=0, $\beta$=0.1).

* Parameter values used are typical of industry ranges:

| | | |
|---|---|---|
| $\lambda$ = 0.8 | (typically between 0.7 and 0.9) | $T_{Ri}$ = 1.0 (for simplicity) |
| $\Pi_o = \Pi_{oo}$ = 1.5 | (initial post-innovation productivity). Normalized pre-innovation productivity is 1.0. | |

As depicted in Fig. 1, we find that, while interdependencies can have a negative impact on productivity performance when firms are relationally unstable (orange line), a policy of greater degree of interfirm openness towards the exchange of tacit knowledge gathered from third party firms –as defined by NKA Model 1– has a positive impact on performance (green line). Also, we find that a policy of greater degree of network openness towards building and sharing network level codified knowledge –as defined by NKA Model 2– has a significant positive impact on the evolution of productivity performance, hence compensating for the productivity performance liability associated with firm interdependences in a loosely coupled network (blue line).

Note that in all real-world situations, although some results of the execution of a given project can be made publicly available (e.g. Model 2 with β=10% as in Fig. 1), the actual "know how" (e.g. the remaining 1-β = 90% of the knowledge) that came from actually executing the project oneself remain with the firm, mostly in the form of tacit knowledge that other firms cannot tap upon without interacting with the knowledgeable firm; hence, requiring Model 1. This makes the two NKA models presented in the present work complementary to one another.

## Validation

The common underlying logic of the models presented here is that of established learning curve theory[††], as required by Carley (1996) and by Burton's notion of *informal docking* (2003: 102). The learning equations developed in this work are an extension of those introduced by Taylor, Levitt and Villarroel (2006), the results of which were backed by substantial empirical evidence from prior field studies (Taylor and Levitt 2004, 2005). The new extensions we made to the learning equations are indeed important, and they follow the knowledge transfer logic of open innovation as first introduced by Chesbrough (2003) for NKA Model 1, and that of open source as described in von Hippel and von Krogh (2003) for NKA Model 2.

Each of our individual sub-models addresses Burton's *what is* questions regarding open innovation policies in interorganizational networks of firms and their effects on performance (2003: 100). Most importantly, the coupling of the open innovation model discussed here, with the TIRI base model addresses the *what might be* question, which goes beyond the simple managerial intuition drawn from experience (Burton 2003: 100). In particular, the mix of parameters, relationship instability, task interdependence, interfirm openness, and network openness, are relatively new in the literature (Taylor, Levitt and Villarroel 2006; Chesbrough 2003), and little understood when taken together to analyze firm and interfirm network performance.

In addition, our model resonates with the empirical evidence found in Powell et al (2005) in their empirical study of biotechnology innovation networks, characterized by a high-rate of formation and dissolution of linkages, and higher innovation performance. Consistent with our findings, a network of loosely coupled firms, implementing complex projects with high levels of

[††] Learning curves were first introduced by Wright (1936), following his findings on the performance improvements in the production of airplanes. This theory was later found to be applicable at other levels, such as firms and nations, in various forms (Yelle 1979). For a good review on the tradition of organizational learning, see Levitt and March (1988) and for a more focused discussion on organizational learning as a basis for the competitive advantage of firms, see Argote and Ingram (2000).

interdependence (Taylor, Levitt, and Villarroel 2006) sees its performance improve by means of open innovation policies, and even achieve better performance through a combination of open innovation policies, particularly through network openness (open source). Given these observations, we argue that our model achieves both face and pattern validity (Carley 1996).

## Limitations

The model presented here is simple, and in spite of the insights it provides regarding the network knowledge advantage of open innovation, it is not an exact representation of the reality of the complex interactions taking place in loosely coupled innovation networks.

In our model, firms are chosen to work together on a project at random. While this was a simple way to address the empirical observations described in Powell et al (2005 :1138), it is a fact that firms cannot necessarily work well, if at all, with other firms completely at random. In reality, as they interact with one another, even as firms display a high-rate of formation and dissolution of links, they may tend to form communities of practice, within which they develop common organizational routines that: on the one hand, further facilitate their collaborative work, and on the other hand, make collaborative work incompatible, or even impossible, when their practices are too different (Brown and Duguid 1991, 1998). How the existence of competing communities of practice impacts performance in open innovation networks is an important research question to address in future research.

Another important limitation of the present simulation is that it does not address firms with heterogeneous levels of relative and absolute absorptive capacities (Cohen and Levinthal 1990; Lane and Lubatkin 1998), and how that would affect their strategic decision on the appropriate degree of openness towards the exchange of knowledge. In the presence of knowledge asymmetries, the degree of openness remains a strategic choice for the firm in its quest for developing an advantage relative to its competitors. Including these heterogeneities in our model will enrich our strategic understanding of open innovation networks as an organizational form.

## DISCUSSION AND IMPLICATIONS

Through our model exploring open innovation, here referred to as NKA (Network Knowledge Advantage), we demonstrate how simple policies towards the open exchange of knowledge, at the firm-dyad level (NKA Model 1) and at the network level (NKA Model 2), can create a performance advantage for firms embedded in a loose network of collaborative relationships.

In the baseline model (TIRI), comparing the productivity performance of self-interested task independent vs. task interdependent firms, we showed that task interdependence negatively affects network productivity performance in situations where network affiliations are unstable (Taylor, Levitt, and Villarroel 2006). In the new NKA models presented in this paper, we extend this previous work to show that firm openness towards the exchange of knowledge, tacit and codified, diminishes the negative impact of task interdependence on firm and network performance.

In particular, Model 1 demonstrates how task interdependent firms –disadvantaged in the TIRI model–, implementing an open innovation policy, find an opportunity for tacit knowledge exchange in their otherwise unavoidable interactions when working together. The result of such interactive knowledge exchange is an improvement in the productivity performance of the individual firms, and that of the interorganizational network as a whole. In the extreme (ideal) case of full knowledge transfer ($\alpha$=100% knowledge transfer), network productivity rates are effectively brought back to optimal levels as in the pure task independence baseline case.

In addition, Model 2 shows that even task independent firms, which would have no obvious incentive towards implementing an open innovation policy due to the nature of their limited interactions with one another, can actually benefit significantly from sharing codified knowledge with the rest of the interorganizational network. As a result from creating a network knowledge commons, every firm benefits from the accumulation of knowledge that it can tap upon to increase its productivity performance on the next project. The effects at the network level are productivity rates in excess of those of the self-interested independent firms.

These findings on network-level knowledge dynamics have important implications for innovation theory. They confirm the empirical findings of Powell and his colleagues (2005) that loose networks can be effective loci for innovation. Furthermore, the computational simulation presented in this paper contributes a model capable of trading off performance liabilities associated with loose coupling with the performance benefits of open knowledge exchanges in loosely coupled networks. These results provide valuable managerial insights for the strategic management of collaborative innovation initiatives, both at the firm and the network levels.

Future research should focus on calibrating the model parameters presented in this paper to empirically observed productivity performance data in loosely coupled interorganizational networks, to further enhance its validity, as suggested by Schreiber and Carley (2004).

## REFERENCES

Argote L., 1999, "*Organizational Learning: Creating, Retaining and Transferring Knowledge*", Springer.

Argote L., Ingram P., 2000, "Knowledge transfer: A basis for competitive advantage in firms", *Organizational Behavior and Human Decision Processes*, Vol. 82, No. 1, pp. 150-169.

Barney J., 1986, "Strategic factor markets: expectations, luck, and business strategy", *Management Science*. 32, 1231-41.

Baum J.A.C., Shipilov A.V., Rowley T.J., 2003, "Where do small worlds come from?", *Industrial and Corporate Change*, 12, 697-725.

Brown J.S., Duguid P., 1991, "Organizational Learning and Communities-of-Practice: Toward a Unified View of Working, Learning, and Innovation", *Organization Science*, Vol. 2, No. 1, 40-57.

Brown J.S., Duguid P., 1998. "Organizing Knowledge", *REFLECTIONS*, Vol. 1(2), pp. 28-44.

Brown J.S., Duguid P., 2000, "*The Social Life of Information*", Harvard Business School Press: Boston, MA.

Brown S.L., Eisenhardt K.M., 1995, "Product development: Past research, present findings and future directions", *Academy of Management Review* 20(2): 343–378.

Burton R.M., 2003, "Computational Laboratories for Organization Science: Questions, Validity and Docking", *Computational and Mathematical Organization Theory*, 9, 2, pp. 91-108.

Carley, K., 1996, “Validating computational models”, Working Paper, Carnegie Mellon University.

Chandler A., 1990, “*Scale and Scope*”, Harvard University Press.

Chesbrough H., 2003a, "*Open Innovation: the new imperative for creating and profiting from technology*”, Harvard Business School Press.

Chesbrough H., 2003b, "The era of open innovation”, *MIT Sloan Management Review* 44 (3): 35-41.

Chesbrough H., 2003c, "The logic of open innovation: Managing intellectual property”, *California Management Review* 45 (3): 33.

Chesbrough H., Crowther A., 2006, "Beyond high tech: early adopters of open innovation in other industries”, *R&D Management* 36, 3, 229-236.

Christensen J.F., Olesen M.H., and Kjaer J.S., 2005, "The industrial dynamics of open innovation - Evidence from the transformation of consumer electronics”, *Research Policy*, 34, pp. 1533-1549.

Cohen W.M., Levinthal D.A., 1990, “Absorptive capacity: A new perspective on learning and innovation”, *Administrative Science Quarterly*, Vol. 35, Issue 1, pp. 128-152.

Cook S., Brown J.S., 1999, "Bridging epistemologies: the generative dance between organizational knowledge and organizational knowing”, *Organization Science* 10(4): 381–400.

Cowan R., Jonard N., 2003, "The dynamics of collective invention”, *Journal of Economic Behavior and Organization*, 52, 513-532.

Cowan R., Jonard N., 2004, "Network structure and the diffusion of knowledge”, *Journal of Economics Dynamics and Control*, 28, 1557-1575.

Dutton J.M., Thomas A., 1984, “Treating Progress Functions as a Managerial Opportunity”, *Academy of Management Review*, Vol. 9, No. 2, 235-247

Grant R.M., 1996, "Prospering in dynamically-competitive environments: Organizational capability as knowledge integration”, *Organization Science* 7 (4): 375-387

Gulati R., Nohria N., Zaheer A., 2000, “Strategic networks”, *Strategic Management Journal* 21(3) 203-215.

Hargadon A., 2003, "*How Breakthroughs Happen: The Surprising Truth About How Companies Innovate*”, Harvard Business School Press.

Herriott S.R., Levinthal D., March J.G., 1985. “Learning from Experience in Organizations”, *American Economic Review*, Vol. 75, No. 2, pp. 298-302.

Kogut B., 2000, “The network as knowledge: generative rules and the emergence of structure”, *Strategic Management Journal*, 21: 405–425.

Lane P., Lubatkin M.., 1998, "Relative Absorptive Capacity and Interorganizational Learning”, *Strategic Management Journal*: 461-477.

Levitt B, March J.G., 1988, “Organizational Learning”, *Annual Review of Sociology*, Vol. 14, pp. 319-338.

Lundvall B.A. (ed.)., 1992, "*National Systems of Innovation: Towards a Theory of Innovation and Interactive Learning*”, London: Pinter Publishers.

Malerba F., 2006, “Innovation and the evolution of industries”, *Journal of Evolutionary Economics* 16 (1-2): 3-23.

Mowery, 1983, "The relationship between contractual and intrafirm forms of industrial research in American manufacturing 1990-1940”, *Explorations in Economic History*: 351-374.

Penrose E., 1959, "*The Theory of the Growth of the Firm*”, Oxford: Oxford University Press.

Polanyi, M., 1966, "*The tacit dimension*”, Garden City, NY. Doubleday Anchor.

Powell W.W., Brantley P., 1992, “Competitive cooperation in biotechnology: Learning through networks?”, In “*Networks and Organizations*”, Nohria N. and Eccles R., eds., Harvard Business School Press, pp.366-394.

Powell W.W., Koput K.W., Smith-Doerr L., 1996, "Interorganizational collaboration and the locus of innovation: networks of learning in biotechnology”, *Administrative Science Quarterly* 41: 116–145.

Powell W.W., White D., Koput K.W., Owen-Smith J., 2005, “Network dynamics and field evolution: The growth of interorganizational collaboration in the life sciences”, *American Journal of Sociology*, 110(4): 1132-1205.

Schilling M.A., Phelps C.C., 2005, "Interfirm Collaboration Networks: The Impact of Small World Connectivity on Firm Innovation”, *SSRN*, abstract No 564422.

Schreiber C., Carley K., 2004, “Going Beyond the Data: Empirical Validation Leading to Grounded Theory”, *Computational and Mathematical Organization Theory*. Vol. 10, Iss. 2; pp. 155-164.

Shan W., Walker G., Kogut B., 1994, "Interfirm cooperation and startup innovation in the biotechnology industry”, *Strategic Management Journal* 15(5): 387–394.

Staudenmayer N., Tripsas M., Tucci C.L., 2005, “Interfirm modularity and its implications for product development”, *Journal of Product Innovation Management*, 22:303-321.

Taylor J., Levitt R., 2004, “Understanding and managing systemic innovation in project¬based industries”, In D. Slevin, D. Cleland & J. Pinto (Eds.), *Innovations: Project management research*. Newton Square, Pennsylvania: Project Management Institute.

Taylor J., Levitt R., 2005, “Aligning innovations and networks: Toward a theory for innovation in interorganizational networks”, Collaboratory for Research on Global Projects. Working Paper.

Taylor J., Levitt R., Villarroel A., 2006, “Simulating learning in interorganizational networks: the insidious role of task interdependence and relational instability in system-level learning”, Best Paper Award, *NAACSOS Conference 2006*.

Uzzi B., Spiro J., 2005, "Collaboration and Creativity: The Small World Problem”, *American Journal of Sociology*, Vol. 111 No. 2: 447–504.

Verspagen B., Duysters G., 2003, "The small worlds of strategic technology alliances”, *Technovation*, 24, pp. 563-571.

von Hippel E., 1988, "*The Sources of Innovation*”, Oxford University Press: New York.

von Hippel E., von Krogh G., 2003, "Open Source Software and the ‘Private-Collective’ Innovation Model: Issues for Organization Science", *Organization Science*, Vol 14, Issue: 2, pp. 209-223.

Watts D.J., 2003, "*Six Degrees: The Science of a Connected Age*”, WW Norton.

Watts D.J., 2004, “The ‘new’ science of networks”, *Annual Review of Sociology*, Vol. 30: 243-270.

Watts D. J, Strogatz S., 1998, "Collective Dynamics of Small-world Networks”, *Nature*, 393, 440-442.

Wright, T.P., 1936, “Factors affecting the cost of airplanes”, *Journal of the Aeronautical Sciences* 3: 122-128.

Yelle L.E., 1979, “The Learning Curve: Historical Review and Comprehensive Survey”, *Decision Sciences*, Vol. 10, pp. 302-328.

# POST-MERGER INTEGRATION OF ORGANIZATIONS: A FRAMEWORK FOR REPRESENTING AGENT-BASED RELATIONSHIPS

T. FRANTZ,[*] Carnegie Mellon University, Pittsburgh, PA

## ABSTRACT

A general framework is put forth for representing a human goal-seeking organization in the confine of a multi-agent system, specifically for the construction of a computational model and computer simulation of a newly-merged organization's behavior—specific to the integration of the human resources. The concepts of this framework are grounded in Computational Organization Theory (COT) and are based on Dynamic Network Analysis (DNA) techniques. The complex, multi-level relational aspects of an organization and its agents are represented as a multidimensional social-network, consisting of dynamic agents, organizational tasks, and individual-agent's knowledge; and, being extendable to other entity classes. This paper provides a general overview of COT and DNA concepts, and presents the ideas by discussing how these concepts could be readily applied as a formalized representation in an agent-based computer model intended for simulating and studying organizational merger dynamics.

**Keywords**: Merger, integration, social network, agent-based modeling, simulation

## INTRODUCTION

This paper presents a framework for representing the numerous social and operative relationships in a human, goal-seeking organization. A comprehensive and effective framework for representing the web of relationships is essential to the development of an agent-based computer simulation that can be effective when modeling the dynamics following an organizational merger (or an acquisition, or formation of a joint venture). The framework described herein, enables the development of realistic computer simulations of complex post-merger dynamics, which ultimately enhance real-world decision-making by enabling risk-free trial-and-error and in-depth analysis of specific merger circumstances.

This framework is grounded in Computational Organization Theory (COT) and is based on Dynamic Network Analysis (DNA) techniques. A deep understanding of these two subjects is important to understand and evaluate this framework; following a brief introduction to the general concept of organizational mergers and the HR integration challenge, both COT and DNA are summarized in this section. The second section of this paper, *Representing Post-Merger Organizational Relationships*, presents how these concepts could be applied as a formalized representation in an agent-based model for the purposes of developing computer simulations intended to simulate a virtual organizational merger.

[*] Corresponding author address: Terrill L. Frantz, Ed.D., Center for Computational Analysis of Social and Organizational Systems (CASOS), Institute for Software Research International (ISRI), School of Computer Science, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA, 15213, USA; e-mail: terrill@cs.cmu.edu.

## Mergers and the HR Integration Challenge

Since the turn of the 19th century, organizational mergers have established themselves on the social landscape of our human experience; throughout our personal life, we are periodically affected, directly or indirectly, by an organizational merger of some variety. Since the first widely recognized wave of business mergers began in 1890, both the following waves and the underlying individual merger events, have become more frequent, larger, and recently, more multi-national. As the world becomes flatter (see Friedman, 2005), certainly a future wave, if not the very next wave, will likely involve more cross-border merger transactions. Perhaps the next wave will be the *big* one (Historically, each new wave ultimately proves to be bigger than the prior). Regardless, mergers are here to stay and will be part of the landscape as long as the organizational form continues to exist. In keeping with the prior trends, these mergers are certain to become increasingly complex, particularly with respect to their human and social aspects.

The general consensus, although strongly contested by many, is that most mergers ultimately fail to accomplish the stated long-term purpose for the union. To this day, 100 years since mergers became a part of the business landscape, Human Resource (HR) integration persists as the Achilles' heel of many mergers. "Cultural fit" consistently emerges as the point of failure for even the most promising mergers. Historically, most merger deals are conceived from the financial and operational perspectives, often with relatively little thought to the human integration risk. While the problems with HR integration have become common knowledge, deal-makers and managers continue to gloss over the human risks, even though the downside is clearly very large. It may be the complexity of human issues that leave the problems unresolved, or it may simply be the managers continuing to do what has worked for them in the past (a focus on finance and operations). This paper will not dwell on the reasons and causes any further.

Much of the current knowledge surrounding post-merger integration pertains to the technical aspects of the organizations' pre- and post-balance sheet and, in the case of a business organization, the equity and asset structure of the combined organization. The dynamics of the balance sheet and income statement are relatively easy to predict as projections are based on very few variables and have somewhat linear dynamics. Often, the softer political and social aspects of the integration are passed-over and lost in the complexity of the larger merger event and deal; perhaps this is because the social, or human, aspects are the most complex, are highly dynamic, and certainly, are the least understood part of the chore. While there exists *some* empirical research into how to address the human-social dimension of a merger, most publications on the subject are antidotal or are just case studies, and are usually geared towards the hands-on practitioner – the internal manager of the integration task, or the integration consultant. There is little actual scientific research using tangible computational tools such as agent-based simulation, mathematical modeling, or other more precise techniques.

The avoidance of the human integration problem cannot continue as the problem is becoming larger and the downside risks for failed merges even greater. Business leaders and managers must be presented with tools that better enable them to simplify the complexity of the human integration issue and position them for more "air-time" and better decisions in this regard. Quantitative models and computer simulations of the human integration dynamic during the merger are certainly called for. To date, just as the business community has avoided the complexity of HR integration, researchers and academics have also withdrawn from meeting the challenge. However, the tide is beginning to turn as the technology know-how and the behavioral theories are starting to meet, and the two disciplines themselves are beginning to merge. Cleary, we are still in the very early days of this intellectual meld. This paper contributes to meeting this challenge by introducing a framework of which the models can be represented and processed in a simulation.

## Computational Organization Theory

Computational Organization Theory (COT) is an interdisciplinary area that melds the advances in traditional organization behavior theory with recent advances in cognitive science, social networks, and computer science, among parts of several other sciences. COT presents a quantitative and exacting perspective on organization behavior by allowing for detailed computational models of organizations to be constructed that precisely capture their behavioral dynamics which, in turn permit realistic computer simulation of an organization's behavior; at least according to its grounding in organization theory and other sciences. Several examples of COT being applied to create predictive model can be found; first-rate examples can be found in Kunz, Levitt, and Jin (1998) and Lin and Carley (1997).

COT allows for combining multiple, complex models of organization behavior that, in the aggregate, is too complex for human thought and understanding. When combined, the nonlinearities of the individual models are just too complex to be useful, however, by applying COT, one can quantifiably predict outcomes from the interaction of numerous theories. Carley and Prietula (1994) provide three advantages of computational organization theory techniques: (a) theoretical precision, (b) predictive accuracy, and (c) facilitation of mechanism-oriented theories (the individual mechanically processes information and performs tasks).

According to Carley and Gasser (2000), the aims of COT are to: (a) develop new theories about organizations, (b) develop tools for the analysis of computational organizational models, and (c) reflect the abstractions of the models back to the actual organizational practice. COT has been used successfully to develop practical theory on organizational design (Lin, 1994), agents' reactive behavior to one another (Verkama, Hamalainen & Ehtamo, 1994), and team coordination (Shi, Kuh & Kleinman, 1994) among other organization behavior dynamics.

Over the four decades of the field (Carley & Wallace, 2000) several COT-based computer simulations have been constructed in this mold and include ORGAHEAD (Carley 1996; Carley & Svoboda 1996), CONSTRUCT (Carley, 2003), and others, and have been used as theory-building devices (Carley & Gasser, 2000; Lin, 1994; Frantz, 2004). To date, COT has not been applied for the specific purpose of theory construction or decision-making in the realm of organizational mergers and merger HR integration, per se.

## Dynamic Network Analysis

Dynamic Network Analysis (DNA) is an extension to the more widely recognized Social Network Analysis (SNA). Both SNA and DNA are approaches and a collection of techniques to fulfill the science of studying the *relationships* among and within groups of human actors and non-human entities. The two approaches advance the thesis that individual and group behaviors are influenced by the relationships people (and other entities) have with others in and outside of the group. For example, the various types and strength of relationship a person has affects how quickly they can find a job, locate an answer to a question, or even presents the likelihood of being invited to a party.

DNA builds on SNA by recognizing that an individual's behavior is based on a *multitude* of relationship types and at *multiple* levels of these omnipresent relations in the form of traditional SNA networks. DNA approaches greater realism by establishing that maintaining existing and establishing new relationships are a probalistic phenomena, rather than simply present or not; additionally, DNA acknowledges that change in one part of the holistic network can affect a change elsewhere in the same or possibly in yet a separate, other network. All of these advances found in DNA provide a foundation for developing a realistic representation of the complex relationships (human and non-human entities, alike) embodied in a real-world organization.

By utilizing the Meta-Matrix (Carley, 2002) data representation approach used in DNA, the complex relationships of multi-level and multi-mode networks can be represented in an analytic-friendly form. The complexity of an entire organization can be captured in an instant-in-time snapshot that reflects the current relationships among the human and non-human entities. Such relations often include workforce agents, organizational tasks (or roles) and bits of knowledge. This point-in-time snapshot is captured in a single meta-matrix. Multiple points in time are reflected in corresponding, multiple meta-matrices.

The Meta-Matrix is an extension of the PCANS model of structure in organization introduced by Krackhardt and Carley (1998). Krackhardt and Carley developed a network-based approach to the architecture of an organization by recognizing and capturing three important domain elements: individuals, tasks and resources. The relational combinations of these elements (or entities) result in five relational primitives: Precedence, Commitment of resources, assignment of individuals to tasks, networks (social relationships of individuals) and skills (linking the individuals to resources). Each of these distinctive relations are simultaneously captured as a distinct network within the same meta-network.

Carley (2003) notes that the advances in DNA techniques and the corresponding development of the Meta-Matrix provides a foundation for combining social networks with cognitive science in multi-agent systems. One such realization of this promise is the CONSTRUCT software developed at Carnegie Mellon University, under the direction of Carley (2003). In the CONSTRUCT model, agents interact in a world in which information is passed from person to person while an information based task is trying to be accomplished. As mentioned earlier, other computational models and computer simulations also embrace the DNA framework in an agent-based world and serve as useful examples of its value.

## REPRESENTING POST-MERGER ORGANIZATIONAL RELATIONSHIPS

The framework for representing the post-merger relationships has as its foundation the basic premise of Computational Organization Theory (COT) and Dynamic Network Analysis (DNA). The framework put forth here is entirely computational, as all the organizational behavior theories embodied in the merger modal are expected to be quantitative, which provides the precision expected in COT and lends itself to predictive accuracy. The framework, following the premise of COT, involves agents that process information and perform assigned tasks. The agents, tasks and resources of the merged organization are maintained following the principles of

DNA. That is, there are multiple relationships among the agents, tasks, and resources, and the relations are maintained in a Meta-Matrix. The over-time dynamics of the changing social network (the agent-to-agent interaction) is based on a Monte-Carlo, probalistic model of interaction.

The first, and possibly the most difficult, task in developing a post-merger model framework is to translate the relevant, *traditional* organization theory into the *computational* embodiment of the theory. Behavior theories involving the processing of information and the performance of tasks would need to be translated into precise computational processes, accordingly.

As the starting point for the development of this post-merger integration data representation, the complex and interdependent structure of the original two organizations are captured using the PCANS model of organizational structure. As with the PCANS model, the post-merger model would involve three elements—individuals, tasks, and resources—however, this likely will be broadened and expanded on to include other entities early on in the post-merger model development. To start, this framework would involve only the three described in PCANS. Adding any additional elements would automatically result in increasing the number of relational primitives form the current PCANS five—precedence, commitment of resources, assignment of individuals to tasks, social networks, and members' skills—, so the complexity of the model would increase rapidly when new elements are added.

The three elements and the various relations are represented by the Meta-Matrix construct, which is widely used in the work of Dynamic Network analysts. The structure of the organization(s), using dynamic network analysis, can be evaluated with standard and extended social network measures and principles. The meta-matrix is a multi-network snapshot in time structure that, in the over-time aggregate, allows for quantitative measurements of the change and dynamics of the relations represented in the meta-matrices. In an agent-based simulation, the agents' changing and emergent relations are represented in this meta-matrix. The collective relations and dynamics for each agent are maintained and quantified using standard social network measures and procedures. From this model, we progress to the merger event of two independent organizations (represented in separate meta-matrices) into a single organization (represented in a unified meta-matrix.).

The two originally independent organizations as captured in the two separate Meta-Matrix, single point-in-time constructs, are then combined into a single Meta-Matrix, which represents the newly merged and unified organization. The initial melding of the two organizations is performed without any changes to the pre-existing elements and relationships.

Artificial organizations embedded in computer simulations are formed by logically grouping artificial agents according to the organizational boundaries and with individual characteristics, such as behaviors and roles. Consistent with real-organizations (and theory), each individual agent has tasks that they are assigned and oriented towards accomplishing, and the agents have knowledge that is dynamic (new knowledge learned with time-based depreciation). In the form of an agent-based simulation model, the individual agents interact by performing task and communicating (passing knowledge) among the alters in their social network. From this complex interaction, organizational performance is manifested and available for measurement and evaluation.

## CONCLUSION

The merging of two or more distinct organizations into a newly-formed, single enterprise is an activity that has become commonplace in recent times. Announcements of market-moving business mergers are an expectation each Monday morning on Wall Street. Announcements of mega-deals are commonplace, as are the countless numbers of less-notorious mergers around the world being revealed each day. Not only are the atypical corporate mergers being consummated, but also merging are local governments, non-governmental organizations, religious groups, and even the occasional sports teams. Unfortunately, the high expectations of the newly formed organization are often dashed quickly as the cultures and personalities mixed into the new entity seem to clash more often than not. The complexity of post-merger integration most often leads to dashed business hopes and even can create human strife. Unfortunately, failure is an all-pervading reality of such idealized mergers. These organizational unions are usually announced with triumphant expressions of glorious levels of synergy that, with some hard work, will better the world for the stakeholders.

However, the on-paper and the in-principled claims rarely are realized and in reality the newly-merged organization fails to realize its expected promise or potential. To address this persistent problem and to formulate a better understanding the dynamics of post-merger integration, agent-based simulations may ultimately be able to provide insight into the complex interactions of the personnel and into the impact on the production goals of the newly merger organization.

As a first critical step toward developing agent-based models for this purpose, a framework for representing the relevant aspects of two organizations with distinct cultures, norms, skills, and knowledge must be developed. This paper presented a method for capturing and representing important aspects of the pre-merger and post-merger organizations. Based on Computational Organization Theory (COT) and Dynamic Network Analysis (DNA), a multi-layered, social-network approach is utilized to construct a point-in-time mathematical model of the agent relations, which in-turn can be used as the basis for developing agent-based models for the simulation of the merging of two organizations into one.

Using the framework described herein, we have a formalized computational model useful for developing future agent-based computer simulations that may be used for exploring the dynamics of post-merger integration in real-world mergers. These models and simulations may ultimately help improve the evaluation of integration strategies and ultimately improve the success-rate of the all-to-often failed organizational mergers; this is a first step.

## ACKNOWLEDGMENTS

This paper is part of the Dynamics Networks project at the center for Computational Analysis of Social and Organizational Systems (CASOS) of the School of Computer Science (SCS) at Carnegie Mellon University (CMU). This work is supported in part by the Office of Naval Research (ONR), United States Navy. Additional support was provided by National Science Foundation (NSF) Integrative Graduate Education and Research Traineeship (IGERT) program and CASOS. The views and proposal contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Office of Naval Research, the National Science Foundation, or the U.S. government.

## REFERENCES

Carley, K.M. (1996). Adaptive organizations: A comparison of strategies for achieving optimal performance. In *Proceedings of the 1996 International Symposium on Command and Control Research and Technology*. Monterey, CA.

Carley, K. M. (2003). Dynamic network analysis. In Ron Breiger & Kathleen M. Carley (Eds.), *Summary of the National Research Council of the National Academies (NRC) Workshop on Social Network Modeling and Analysis*, Washington, DC: The National Academies Press.

Carley, K. M. (2003). *CONSTRUCT software*. http://www.casos.cs.cmu.edu/projects/construct/

Carley, K. M. & Gasser, L. (2000). Computational organization theory. In Gerhard Weiss, *Mutliagent Systems*, (pp. 299-330). Cambridge, MA: MIT Press.

Carley, K. M. & Prietula, M. J. (1994). Computational organization theory – An introduction. In Kathleen M. Carley & Michael J. Prietula (Eds.), *Computational organization theory* (pp. xi – xvii). Hillsdale, NJ: Lawrence Erlbaum.

Carley, K. M. & Svoboda, D. M. (1996). Modeling organizational adaptation as a simulated annealing process. *Sociological Methods and Research, 25*(1), 138-168.

Carley, K. & Wallace, W. (2000). *Computational organization theory: A new perspective*. Norwell, MA: Kluwer Academic Publishers

Frantz, T. (2004). Patterns *of change in organization performance*. Doctoral dissertation, Pepperdine University.

Friedman, T. L. (2005). *The world is flat: A brief history of the twenty-first century*. New York: Farrar, Straus and Giroux.

Krackhardt, David & Carley, Kathleen. (1998). A PCANS model of structure in organization. *Proceedings of the 1998 International Symposium on command and Control Research and Technology,* 113-119. Monterey, CA.

Kunz, K. C., Levitt, R. E., Jin, Y. (1998). The virtual design team: A computational model of project organizations. *Communications of the Association for Computing Machinery, 41*(11), 84-92.

Lin, Z. (1994). A theoretical evaluation of measures of organizational design: Interrelationship and performance predictability. In Kathleen M. Carley & Michael J. Prietula (Eds.), *Computational organization theory* (pp. 113-159). Hillsdale, NJ: Lawrence Erlbaum.

Lin, Z., & Carley, K. M. (1997). Organizational decision making and error in a dynamic task environment. *Journal of Mathematical Sociology, 22.*

Shi, P., Kuh, P. B., & Kleinman, D. L. (1994). Team coordination under individual and team goals. In Kathleen M. Carley & Michael J. Prietula (Eds.), *Computational organization theory* (pp. 161-177). Hillsdale, NJ: Lawrence Erlbaum.

Verkama, M., Hamalainen, R. P., & Ehtamo, H. (1994). Modeling and computational analysis of reactive behavior in organizations. In Kathleen M. Carley & Michael J. Prietula (Eds.), *Computational organization theory* (pp. 161-177). Hillsdale, NJ: Lawrence Erlbaum.

# TOWARDS GENERATIVE ACTIVITY-BASED MODELS FOR LARGE-SCALE SOCIO-TECHNICAL SIMULATIONS (EXTENDED ABSTRACT)

G. ISTRATE, Los Alamos National Laboratory, Los Alamos, NM
A. HANSSON,* Los Alamos National Laboratory, Los Alamos, NM
C. D. TALLMAN, Los Alamos National Laboratory, Los Alamos, NM
L. CUELLAR-HENGARTNER, Los Alamos National Laboratory, Los Alamos, NM
N. HENGARTNER, Los Alamos National Laboratory, Los Alamos, NM

## ABSTRACT

An approach to activity modeling based on the theory of random graphs has been proposed in (Eubank et al., 2004a). The approach does not represent temporal duration and activity type information, and does not yield a full activity generator. We present a number of theoretical concepts and experimental results supporting this goal. They include (i) the use of multi-labeled bipartite graphs to represent the missing information, (ii) the automatic inference of the label set via temporal clustering of activities, (iii) we highlight the existence and role of clustering and community structure, a feature of real activity sets that needs to be incorporated in the model.

**Keywords:** Activity modeling, random graphs, cultural similarity, computational social theory

## INTRODUCTION

Activity-based approaches (Timmermans, 2005) emerged in the 1970s, and are becoming increasingly popular in Computational Social Theory, particularly in areas such as travel planning and forecasting, disease modeling and mitigation, and in several other problems related to homeland security. Indeed, since it is often the case (for instance when modeling disease transmission) that the social interaction network is the determining factor for the overall social dynamics, one could in principle replace (and greatly speed-up) agent-based simulations by simulating the underlying dynamics directly on the social network. Analytical models of the activity network have the additional benefit of leading (at least in principle) to *generic methods* for modeling social dynamics.

Developing generic models of human behavior for urban infrastructure simulations has a number of potential benefits to simulation-supported decision-making:

- Such generic models could replace nonparametric, purely data-driven activity-generation in social agent simulations, enabling rapid response, quick turnaround studies.
- Parametric models are more readily transferred to situations when detailed data is incomplete or lacking, and can overcome the inconsistencies of commercial and publicly available data sets.

*Corresponding author address:* Anders Hansson, Discrete Simulation Sciences (CCS-5), Los Alamos National Laboratory, P.O. Box 1663, MS M997, Los Alamos, NM 87545; e-mail: hansson@lanl.gov.

◦ Such models facilitate the computational assessment of the robustness of various general guidelines (normative properties or policies) with respect to variations in the quantitative properties of the simulations.

A recent article in *Nature* (Eubank et al., 2004a) proposes an approach to activity modeling based on ideas from the theory of random networks (Newman et al., 2006). The authors investigate properties of the bipartite people-location networks arising from computational runs of the multiagent simulation EpiSims.[1] The networks arising from EpiSims runs naturally encode physical contact patterns that arise from movements of individuals between specific locations. As shown in (Eubank et al., 2004a), the structural properties of these networks (e.g., the so-called *graph expansion*) have significant implications for the efficiency of disease mitigation and control. Experimental studies have further shown that the Aiello-Chung-Lu (ACL) model from random graph theory captures a number of structural characteristics of this activity-induced physical contact graph.

The basis for the use of random graph models in activity modeling is encapsulated in the following two hypotheses:

◦ The *low-dimension social modeling hypothesis:* In a nutshell, *macroscopic characteristics of the bipartite graph generated from activity schedules can be described by a small number of parameters*.
◦ The *cultural similarity hypothesis:* The macroscopic characteristics of urban environments show a significant degree of correlation (at least for large cities in the United States).

The two hypotheses were stated in an unpublished manuscript (Barrett et al., 2004), with a number of measurements to support them. However, the results in (Barrett et al., 2004) do not have significant implications on the definition of synthetic activity generators as few modeling guidelines are provided.

Our long-term research objective is to complete the random-graph based approach to a full-fledged generative model of activities, incorporating temporal and activity-type information (components missing in the preliminary model proposed in (Eubank et al., 2004a)). In this paper, we present information supporting this goal:

◦ First, we show how to generalize the ACL model to a bipartite random graph model that incorporates multi-label information. The set of labels can then be chosen to encode information about both activity types and their start- and end-times.
◦ We then show how the set of labels can be inferred from real survey data by clustering.
◦ Finally, we give evidence for the low-dimensional nature of activity data. In particular, we show how different demographic communities can be inferred from activity data.

[1]EpiSims is a large-scale individual-based epidemiology simulation developed at Los Alamos National Laboratory (Barrett et al., 2005), based on detailed census, land-use, and population-mobility sample data.

## RANDOM NETWORK MODELS OF ACTIVITY DATA AND THEIR DRAWBACKS

The results in (Eubank et al., 2004a) are based on modeling activities using *bipartite graphs (networks):*

**Definition 1** *A* bipartite network $G = (V, E)$ *is a network such that its vertex set* $V$ *can be partitioned into nonempty subsets* $V = V_1 \cup V_2$ *such that every edge in the edge set* $E$ *of* $G$ *has one endpoint in* $V_1$ *and one endpoint in* $V_2$*.*

Activities can be naturally modeled by bipartite graphs $G = (P, L, E)$ where $P$ is the set of *people*, $L$ is the set of *locations*, and for every person $p \in P$ and any activity $a$ of $P$ performed at location $l \in L$ we add an edge $(p, l)$ to graph $G$. The edge can be endowed with additional information, such as the start- and end-times of the activity, the activity type, etc.

The results in (Eubank et al., 2004a) draw on existing literature in random graph theory, adapted to bipartite graphs. A critical measure to capture is *degree distribution*. This measure differentiates epidemiology approaches based on random graphs from the more common *SIR models*, which assume very simple structure of the social network, given by uniform or block mixing. Being able to capture arbitrary degree distributions gives rise to the most natural model of this type, the so-called *configuration model*. It is specified by a distribution of degrees $(d_1, \ldots, d_n)$. Sampling from this model is reasonably easy: for every index $i$ we create $d_i$ copies of a node, connected to all nodes arising from a different index. One then considers a random perfect matching in this graph, and contract the copies of each node. With constant probability this gives rise to a simple graph (one without duplicated edges).

Since the generation and analysis of random samples from the configuration model is reasonably complicated, the paper (Eubank et al., 2004a) used a "relaxed" version of this model, the so-called Aiello-Chung-Lu random graph model, more precisely its adaptation to bipartite networks:

**Definition 2** *Let* $D_1 = (d_{1,1}, \ldots, d_{1,p})$ *and* $D_2 = (d_{2,1}, \ldots, d_{2,l})$ *be two sequences of integers with equal sum. A sample of the ACL bipartite graph with degree distributions* $D_1$ *and* $D_2$ *is a bipartite graph with* $p + l$ *vertices labeled* $\{P_1, \ldots, P_p\}$ *and* $\{L_1, \ldots, L_l\}$*. Any two nodes* $P_i$ *and* $L_j$ *are connected independently at random with probability* $p_{i,j} = \frac{d_{1,i} \cdot d_{2,j}}{\sigma}$*, where* $\sigma$ *is the proportionality factor* $\sigma = \sum_{i,j} d_{1,i} \cdot d_{2,j}$ *(such that resulting probabilities add up to one).*

In (Eubank et al., 2006) it was shown that a number of structural characteristics of the people-location network in EpiSims can be captured to a reasonable degree of accuracy by a random ACL graph with exponential distribution of degrees on the people side and a power law degree on the locations side.

Nevertheless, a random model such as the one outlined above has a number of drawbacks:

- The model only captures activity properties on a timescale of 24 hours (or whatever time scale is used for recording activities). It does not incorporate information on activity types and their start- and end-times, and thus cannot be used for activity generation.

- It does not capture *community structure*, as displayed in real networks. Indeed, the probability of an edge (in an ACL random graph) depends only on the degree of the two nodes. In contrast, in real urban environments the activities that people undertake are highly clustered, reflecting both *geographic* and *demographic* locality, and this is potentially important for many problems (e.g. disease propagation).

## Multi-labeled Networks and the Temporal Disaggregation of Activity Data

In this section we give a generalization of the ACL random graph that is compatible with the task of activity modeling. The basis for the generalization is the following observation: suppose that instead of edges of a single type we "color" each edge with a symbol chosen from a fixed set of *labels*. The coloring is such that no more than one edge adjacent to a person node is colored with a given label. Then we can define, for every person or location node $n$ and every label $\mu$, the *degree of node $n$ with respect to label $\mu$*, as the number of edges in graph $G$ that are adjacent to node $n$ and labeled $\mu$. By the convention we imposed that the degree of any person node with respect to a label $\mu$ is either zero or one. In contrast, there is no such restriction on location nodes.

Suppose now that we define the set of labels, and we color the activity graph such that:

- *Labels refine activity types.* That is, for any label $l$ there exists an unique activity type $A$ (e.g., home or work) such that all edges labeled $l$ correspond to activities of type $A$.
- *Labels record temporal information.* That is, all activities labeled $l$ are "clustered" with respect to activity start- and end-times. These activities correspond to approximately similar time periods.

One can now reduce the problem of activity generation to one of generating random bipartite graphs with similar label degree distributions. Formally we want to generate random samples from the model specified as follows:

**Definition 3** *Let $W = \{l_1, \ldots, l_k\}$ be a set of labels with $k$ elements, and let $D_1 = (\overline{d_{1,1}}, \ldots, \overline{d_{1,p}})$ and $D_2 = (\overline{d_{2,1}}, \ldots, \overline{d_{2,l}})$ be two sets of $k$-tuples of integers with equal sum.*

*A* multi-labeled bipartite graph with generalized degree distributions $D_1, D_2$ *is a bipartite graph $V = (V_1, V_2, E)$ with edges labeled with labels in $W$ such that for all $j \in 1 \ldots k$*

- *For all $i = 1, \ldots, p$, the number of edges adjacent to the $i$th element of $V_1$ that are colored with label $l_j$ is $d_{1,j}$.*
- *For all $i = 1, \ldots, l$, the number of edges adjacent to the $i$th element of $V_2$ that are colored with label $l_j$ is $d_{2,j}$.*

One can easily see the model in the previous definition as a generalization of the configuration model: to each vertex add a number of colored stubs, with their number specified by the proper degree. Create then, separately for each color, a random matching between stubs on the left and right-hand side of the graph.

The kind of labels we are going to use will aim to reflect the additional information available in the activity data that is not captured by the degree information. For instance a label could correspond to activity type “work without interruption, from approximately 9 to 5.” In general, however, we will have to use more complicated encodings in cases where several activities are highly correlated. For instance, “work” interrupted by “lunch” activity will show in our data as two work activities with different time intervals taking place (in most cases) at the same location. It is, therefore, useful to introduce a single label that will represent both work before and after lunch.

The set of activities of a single person respects temporal disjointness: at most one activity can take place at any one time. This constraint will be reflected by a corresponding constraint on the labels.

Of course the generalization of the configuration model described above still suffers from a number of drawbacks:

- ∘ It is impractical to completely specify the set of label degrees. Rather, we would like to model the degree distributions.
- ∘ The model still does not incorporate community structure.
- ∘ It is not clear that a set of labels can be inferred from the data.

In the next sections we address these issues. First, we show that labels can often be inferred from temporal clustering of activities. We present preliminary results on identifying clusters of activity patterns.

## STRUCTURAL ANALYSIS OF ACTIVITY DATA IN REAL URBAN ENVIRONMENTS

We now present the results of a preliminary investigation of activity data arising from three surveys: a national-level household survey, as well as two household survey for the urban areas of Chicago and Houston.

The National Household Transportation Survey (NHTS) is a nationwide survey of travel patterns taken in 2001 (U.S. Department of Transportation, 2001). The table DAYPUB, which records trips for each person on a particular day, was converted from trips to activities by looking at the intervals between trips. All activities were selected except for those persons with flags indicating that they were out of town, and persons who had missing time information for their activities.

There were $761{,}811$ activity records in total. In all cases we eliminated any activity that crossed the maximum time, which was $1{,}740$ minutes (29 hours) from the start of the survey. We binned the start and end time activity pairs into a table of 30-minute intervals, truncating intermediate times to the previous $30$ minutes. We normalized the data, dividing by the sum of the entries in the table to create a probability density graph, and plotted it with $24$ color breaks ranging from $2 \times 10^{-15}$ to $2 \times 10^{-3}$. We added contour lines at $1.25 \times 10^{-3}$, $2.5 \times 10^{-3}$, and $5 \times 10^{-3}$, labeling them as $1$, $2$, and $4$ respectively. These levels are arbitrary; they help to show the shape of the probability density surface.

The Chicago data is from the 1990 Chicago Area Transportation Study (CATS) (Ghislandi, 1990). The activity records were taken from the Trip files table, converted to activities as in the

NHTS table above. There were 189,253 activity records in total. We binned, normalized, and plotted the data as in the NHTS case above. The primary difference is that for this survey the time was counted from 4:00 a.m. We added back four hours in the plot to show comparable results.

The Houston data is from a 1994 Household Travel Survey by the Houston / Galveston Area Council (Houston-Galveston Area Council, 2001). The activity records were taken from the activity table ACTIVITY.ORG. There were 29,045 activity records in total. The 541 work-related activities, the 338 other activities, and the 196 college activities were rejected for being too small a sample size. We binned, normalized, and plotted the data as in the NHTS case above.

The activity types employed in the three surveys do not completely overlap. Table 1 presents the breakout of activities per activity type for each of the three survey. An "NR" (Not Recorded) entry signifies that an activity type with that name was not recorded in the study.

We will now present probability density graphs for start- and end-times of four different activities: home, work, education, and serve passenger. As we will see, the graphs will all support the notion of a cultural similarity. College activities were not explicitly recorded in the CATS data; they were instead added to school activities. In order to compare the three sets of survey data, we therefore merged the school and college activities of the NHTS and Houston surveys, respectively. Once aggregated, these two activity types are classified as "education."

TABLE 1: Activity counts in sample data

| Activity type | NTHS | Chicago | Houston |
|---|---|---|---|
| Work | 56,843 | 28,235 | 3,308 |
| Work-related trips | 14,508 | 6,781 | 541 |
| Home | 220,413 | 90,168 | 8,321 |
| School | 14,739 | 4,526 | 1,418 |
| College | 2,757 | NR | 196 |
| Retail | 80,348 | 16,341 | 2,298 |
| Serve passenger | 68,398 | 10,759 | 2,200 |
| Visit | 26,893 | NR | NR |
| Services | NR | NR | 1,326 |
| Medical | 9,021 | NR | NR |
| Recreation | 41,496 | 7,700 | 2,911 |
| Banking | 16,443 | 2,983 | NR |
| Meal | 36,410 | 15,200 | NR |
| Daycare | 2,926 | NR | NR |
| Other | 2,926 | 13,520 | 338 |
| TOTAL | 761,811 | 189,253 | 29,045 |

## Clustering of Home Activity Data

Figure 6 shows the probability density graphs of the home activity start- and end-times for the NHTS, Chicago, and Houston surveys. The three graphs display the following common features:

- ○ There is a distinct cluster at the very left of each graph, which simply reflects that people are at home at the beginning of the survey (i.e., in the early morning). It is also seen how most people leave their home between 6–9 a.m.
- ○ There are two more clusters, corresponding to lunch, which takes place around noon, and evening activities, which peaks some time between 5–6 p.m. As expected, the lunch cluster is closer to the diagonal, as this indicates a shorter activity (generally less than an hour).

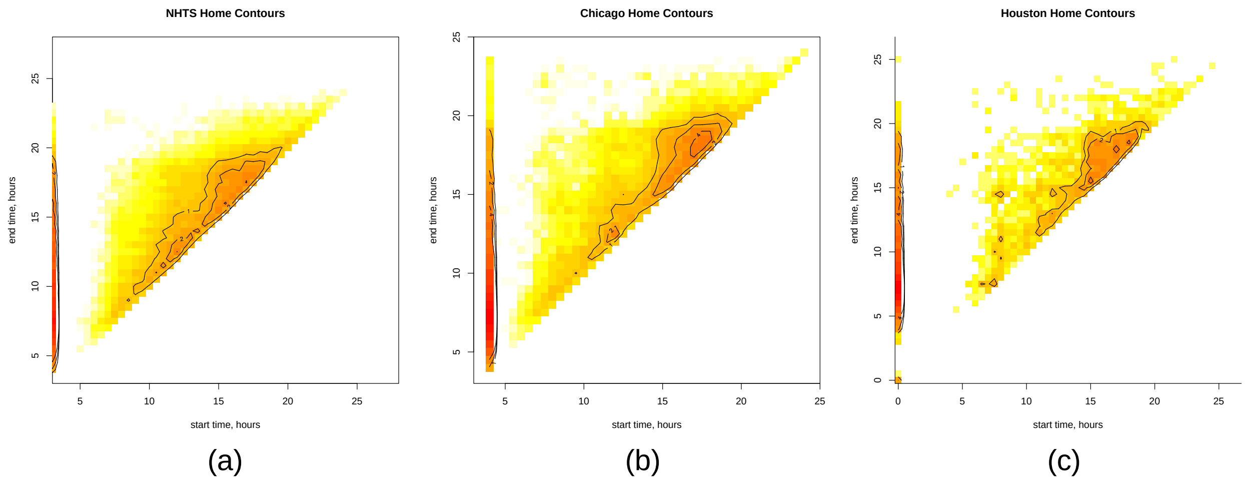


FIGURE 1: Probability densities of the home activity start- and end-times for (a) NHTS, (b) Chicago, and (c) Houston

## Clustering of Work Activity Data

Figure 2 shows the probability density graphs of the work activity start- and end-times for the NHTS, Chicago, and Houston surveys. The three graphs display the following common features:

- ○ The most dominant cluster is farthest away from the diagonal, and this corresponds to an uninterrupted workday, starting between 6–9 a.m. People in this cluster were generally recording a single work activity throughout the day.
- ○ By instead considering people who (in general) recorded two work activities throughout the day, we can locate the two clusters closer to the diagonal. Like the first cluster, these clusters are also very distinct, and they correspond to work before and after lunch, respectively. It can be seen how the start-time of the first cluster (before lunch work) and the end-time of the second cluster (after lunch work) naturally peak at the same time as the start- and end-times of the uninterrupted work activity. The first work activity ends around noon and the second starts around 1 p.m.
- ○ One can also see some smaller clusters that correspond to shift workers who start their day in the afternoon and work until the evening. (Some shift work activities for Houston are apparently cut off as they appear at the very top of the graph.)

Let us provide two more remarks concerning the work activity data. The Houston survey is smaller, and this fact explains why the Houston graph shows less dispersion than the other two graphs. Finally, we conclude that work makes a good candidate for an *anchor activity* (for adults), i.e., once the work activity pattern of a given individual has been established, one can readily sample the start- and end-times of his/her remaining activities (e.g., a person who interrupts his/her work for a lunch break could potentially go home for lunch).

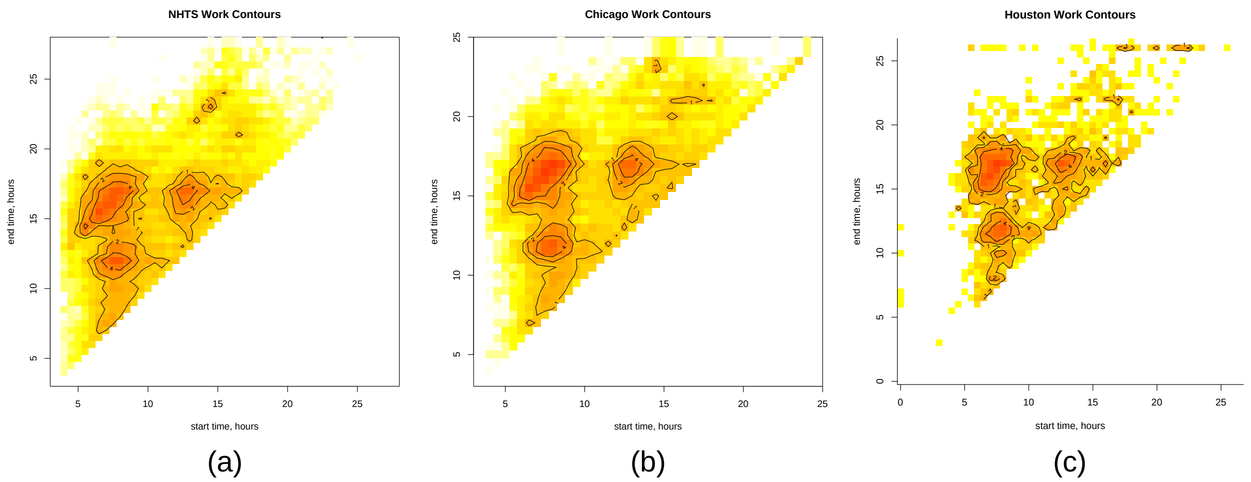


FIGURE 2: Probability densities of the work activity start- and end-times for (a) NHTS, (b) Chicago, and (c) Houston

## Clustering of Education Activity Data

Figure 3 shows the probability density graphs of the education activity start- and end-times for the NHTS, Chicago, and Houston surveys. The three graphs display the following common features:

- ○ The education activities appear to be somewhat more fragmented than work activities (for example, see the Chicago data). We do not attribute this behavior to a smaller sample size, but instead hypothesize that students engage in various other activities between classes.
- ○ The three clusters that was seen for work activities can also be identified here—although not as easily: uninterrupted education that starts in the morning and ends in the afternoon, as well as a education before and after lunch (see the NHTS graph).
- ○ One can also see a cluster corresponding to evening classes. This cluster is particularly evident in the Chicago graph, which perhaps could be explained by the rich educational choice of the Chicago metropolitan area (in contrast to the NHTS data in which this effect is somewhat neutralized by rural areas).
- ○ Finally, we observe that there are virtually no education activities before 7 a.m. in the morning, as expected.

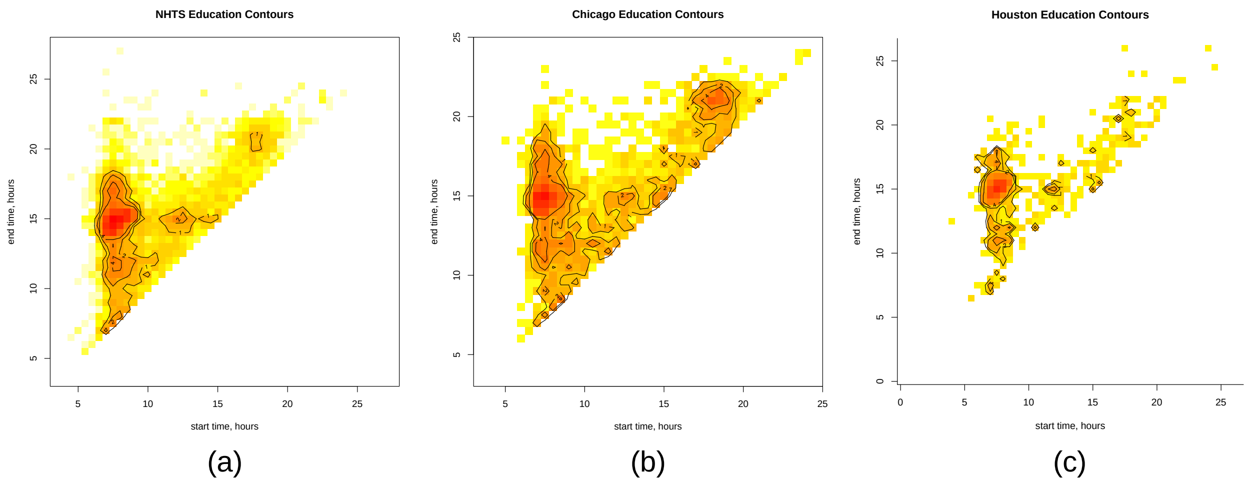


FIGURE 3: Probability densities of the education activity start- and end-times for (a) NHTS, (b) Chicago, and (c) Houston

## Clustering of Serve Passenger Activity Data

Figure 4 shows the probability density graphs of the serve passenger activity start- and end-times for the NHTS, Chicago, and Houston surveys. The three graphs display the following common features:

- ◦ The serve passenger activity type is more or less evenly distributed throughout the day from around 7 a.m. until around 9 p.m. Moreover, the activities tend to be short (the cluster is concentrated close to the diagonal of the graph).
- ◦ One could identify two short activity dips (most apparent in the Chicago data): the first around 10 a.m. and the second around 1 p.m. At these points in time, most people have generally arrived at their work, either in the morning, or after lunch, and the demand for the serve passenger activity naturally drops for a while.

## CLUSTERING AND IDENTIFICATION OF ACTIVITY PATTERNS FROM DATA

The previous section has provided evidence for the cultural similarity of two urban areas with respect to one national survey. In this section we provide evidence for the existence of natural community structure. In this extended abstract we do not address the full problem of community detection based on concepts such as modularity. Rather, our goal is to cluster activity sets based on common patterns.

The data-set we used for this purpose was a subset of the Chicago data-set, as used in the

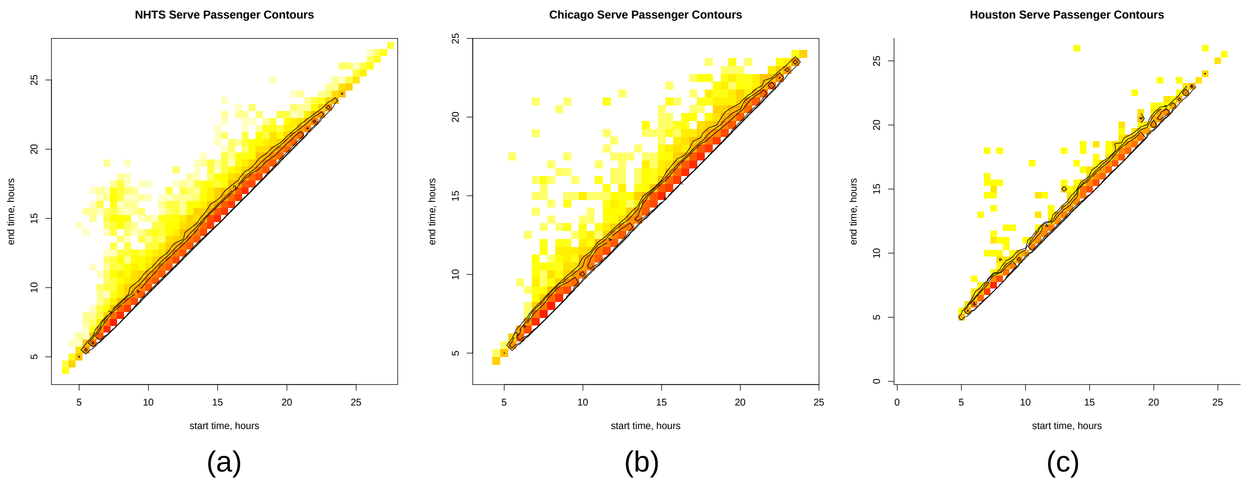


FIGURE 4: Probability densities of the serve passenger activity start- and end-times for (a) NHTS, (b) Chicago, and (c) Houston

simulations Transims and Episims. A sample of $0.1\%$ of the individuals in the simulation was used, resulting in $4{,}365$ individuals. Table 2 displays the percentage of people performing the given activity type at least once.

TABLE 2: Percentage of people performing a specific activity

| Home | Work | School | Retail | Other | Serve p. | College |
|---|---|---|---|---|---|---|
| 99.8 | 45.8 | 20.3 | 27.7 | 50.5 | 11.9 | 4.1 |

Next we performed a Principal Component Analysis (PCA) on the sets of individual activities. We create a matrix, whose rows are people and whose columns correspond to activity types. The entries of the matrix are zero, except for the cases when the given person performs the corresponding activity, in which case it is equal to one. PCA tries to find a space generated by the PC's (vectors) such that the distance of the observations (in this case, the $4{,}365$ vectors of $0$s and $1$s) to the space generated by the PC's is minimized. It does this by iteratively seeks a subspace such that the projection onto that subspace has maximal variance. The method can often reveal clusters in the data. The result of the PCA, presented in Table 3, show that clearly the first three PC's are enough to explain a significant amount ($76$–$84\%$) of the variance in the input data. Table 4 presents the coefficients in the linear combination of the $0/1$ activity variables that specify the PC's, e.g. the equation of $PC_1$ is:

$$-0.759 \cdot (\text{Home}) - 0.378 \cdot (\text{Work}) - 0.144 \cdot (\text{School}) - 0.235 \cdot (\text{Retail})$$
$$-0.44 \cdot (\text{Other}) - 0.103 \cdot (\text{Serve passenger}) - 0.027 \cdot (\text{College}).$$

The important features in this equation and similar ones are the magnitude of the coefficients and quantities with a negative sign. In $PC_1$ the most important variables are "Home," "Work," and "Other." This suggests that most of the variance in the activity data is given by the presence or absence of one of these three activity types. For $PC_2$ the important variables are "Work," "Other," and "School." Also, observe that "Work" has opposite sign to "School" and "Other." So the variance along $PC_2$ is capturing the group of people that work, but neither go to school nor do "other" activities, or those that are in the opposite situation. Similarly, the variance along $PC_3$ captures people going to school that do not do retail activities and vice-versa, people that do retail but do not go to school. The results of the PCA are also displayed in Figure 5, where the projection of activity points is presented with respect to each pair of PC's.

We have also performed PCA on the *total time* spent doing the activities. That is, for each person we keep track of how much time he/she spent doing each of the activities. This amount becomes the input to a matrix indexed by people (for the rows) and activity types (for the columns). The relative importance of the PC's in explaining variance in the data is presented in Table 5. It is easy to see that most variability in the data is explained by the first four PC's.

TABLE 3: Importance of PCA components for activity data

| Importance of components: | $PC_1$ | $PC_2$ | $PC_3$ | $PC_4$ | $PC_5$ | $PC_6$ | $PC_7$ |
|---|---|---|---|---|---|---|---|
| Standard deviation | 1.288 | 0.578 | 0.462 | 0.4319 | 0.3193 | 0.2612 | 0.1935 |
| Proportion of variance | 0.638 | 0.128 | 0.082 | 0.0717 | 0.0392 | 0.0262 | 0.0144 |
| Cumulative proportion | 0.638 | 0.766 | 0.848 | 0.9202 | 0.9594 | 0.9856 | 1.0000 |

TABLE 4: PCA coefficients for activity data

| | $PC_1$ | $PC_2$ | $PC_3$ | $PC_4$ | $PC_5$ | $PC_6$ | $PC_7$ |
|---|---|---|---|---|---|---|---|
| Home | $-0.759$ | $0.035$ | $-0.230$ | $0.220$ | $-0.056$ | $-0.540$ | $0.164$ |
| Work | $-0.378$ | $0.774$ | $0.060$ | $-0.165$ | $-0.042$ | $0.466$ | $-0.091$ |
| School | $-0.144$ | $-0.298$ | $-0.677$ | $0.319$ | $0.037$ | $0.553$ | $-0.150$ |
| Retail | $-0.235$ | $-0.212$ | $0.664$ | $0.611$ | $-0.082$ | $0.273$ | $-0.064$ |
| Other | $-0.440$ | $-0.515$ | $0.192$ | $-0.670$ | $-0.068$ | $0.219$ | $-0.058$ |
| Serve passenger | $-0.103$ | $-0.008$ | $0.083$ | $-0.002$ | $0.991$ | $0.009$ | $0.010$ |
| College | $-0.030$ | $0.024$ | $0.006$ | $0.003$ | $0.009$ | $-0.252$ | $-0.967$ |

The coefficients of the PCA are listed in Table 6. In this case the most important variables for $PC_1$ are "Home" and "Work," and then "Other" and "School" (but much less than in the PCA for activities). The second PC is most influenced by the "Work," "Other" and "School" variables. Note the opposite signs: people working spend less time doing "other" activities and/or at school. $PC_3$, on the other hand, depicts people that do "other" activities and work.

The percentage of time people spent in each activity is presented in Table 7. Not surprisingly, the time spent at home and at work are most important. The "other" activities category, on the other hand, does not seem to be so important in the PCA.

In Figure 6 (b) we analyze more closely the relation between the first three PC's. Looking at the $PC_1$ vs. the $PC_2$ plot unveils two clusters of points along the wedges of the triangle. They

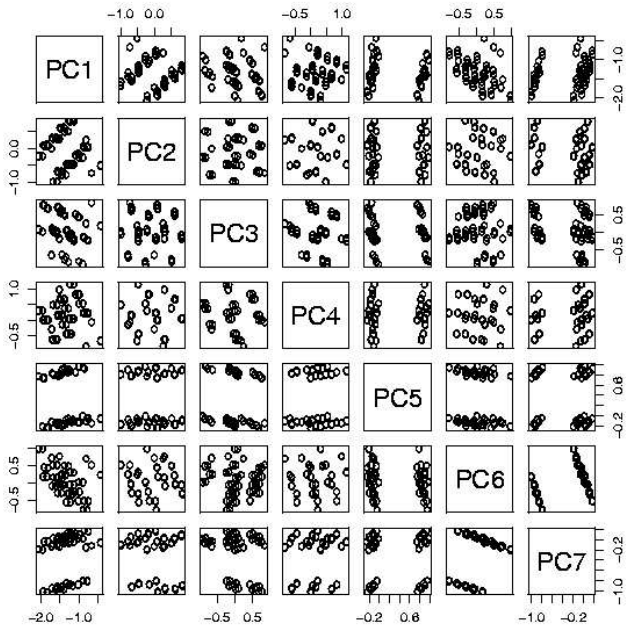


FIGURE 5: PCA for activity data

TABLE 5: Relative contribution of PC's to explaining variance in duration data

| | $PC_1$ | $PC_2$ | $PC_3$ | $PC_4$ | $PC_5$ | $PC_6$ | $PC_7$ |
|---|---|---|---|---|---|---|---|
| Standard deviation | 19.70 | 4.98 | 4.12 | 3.36 | 1.45 | 0.91771 | 0.66118 |
| Proportion of variance | 0.87 | 0.06 | 0.04 | 0.03 | 0.01 | 0.00189 | 0.00098 |
| Cumulative proportion | 0.87 | 0.93 | 0.97 | 0.99 | 0.997 | 0.99902 | 1.00000 |

mostly correspond to a distinction into two distinct group of people according to the time spent working. Indeed, for $PC_2$, work is the only activity with a negative coefficient. People with a strong negative value of $PC_2$ are those spending a significant amount of time in work activities. In contrast, people with a positive value in the $PC_2$ component spend more time in the school and "other" activities.

We next present results on the independence and correlation of activities. Table 8 gives the total number of people that perform both of a given pair of activities. The diagonal counts people that perform the given activity at least once. Table 9 presents the corresponding joint probabilities of the main activity categories. Observe that activity home is independent of all other activities (see also the correlation below). For example, the probability of working and doing retailing is $0.17$. The probability of working is $0.46$.

We now compute the conditional probabilities, given by $p(i|j) = p(i,j)/p(j,j)$. The results are presented in Table 10. So for example the probability of working, given that the person goes retailing is $0.999$. Or probability of retailing, giving that one works, is $0.2$. A further measure

TABLE 6: PCA coefficients for duration data

| | $PC_1$ | $PC_2$ | $PC_3$ | $PC_4$ | $PC_5$ | $PC_6$ | $PC_7$ |
|---|---|---|---|---|---|---|---|
| Home | $-0.981$ | $0.091$ | $0.148$ | $0.080$ | $0.023$ | $0.010$ | $0.003$ |
| Work | $-0.165$ | $-0.853$ | $-0.431$ | $-0.241$ | $-0.024$ | $-0.008$ | $0.000$ |
| School | $-0.062$ | $0.332$ | $-0.109$ | $-0.935$ | $-0.007$ | $-0.015$ | $-0.002$ |
| Retail | $-0.018$ | $0.022$ | $0.012$ | $0.016$ | $-0.999$ | $-0.017$ | $0.001$ |
| Other | $-0.076$ | $0.391$ | $-0.883$ | $0.247$ | $0.003$ | $-0.005$ | $0.000$ |
| Serve passenger | $-0.003$ | $0.000$ | $0.001$ | $0.002$ | $-0.001$ | $0.014$ | $-1.000$ |
| College | $-0.007$ | $0.000$ | $0.011$ | $0.015$ | $0.017$ | $-1.000$ | $-0.014$ |

TABLE 7: Percentage of time spent in various activities

| Home | Work | School | Retail | Other | Serve p. | College |
|---|---|---|---|---|---|---|
| 66.4 | 13.0 | 5.6 | 1.5 | 7.4 | 0.3 | 0.5 |

of activity independence is given by the amount $p(i|j) - p(i,i)$. A value close to zero in Table 11 signifies independence.

A different indicator of independence, the statistical correlation between activities, is presented in Table 12. Home activity is independent of every other activity, and college seems to be the activity less correlated with all the others. Also observe that the correlations between activities is low, somewhere between $-0.4$ and $0.1$.

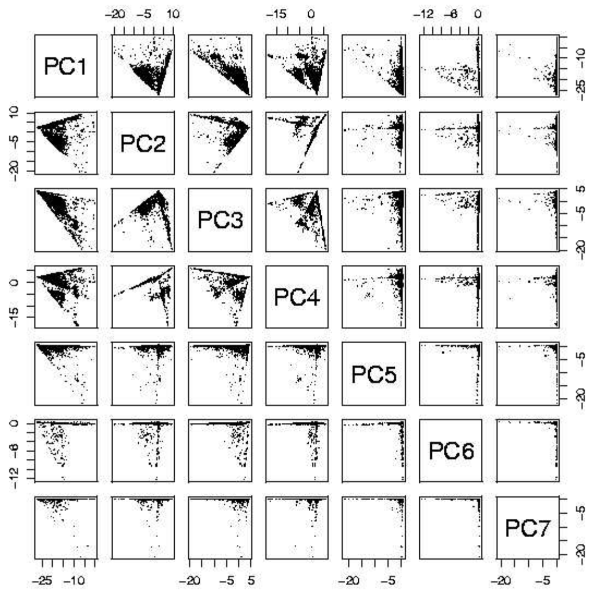
The conclusion of the preliminary investigations of this section suggest that clustering activity patterns is likely to be significant in a full-fledged model. Ignoring such clustering destroys all correlations between activities.

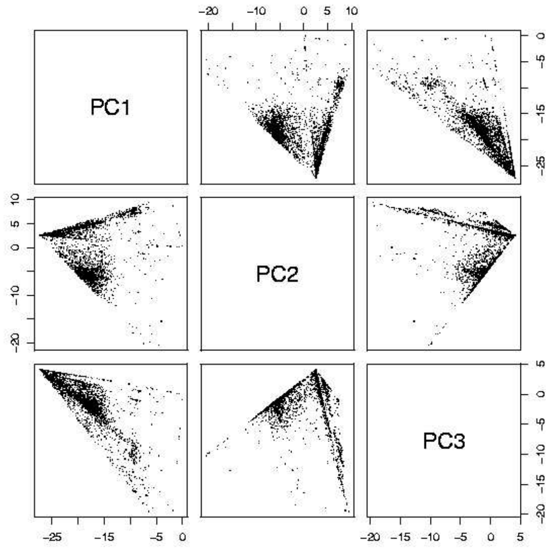
## CONCLUSIONS

We have outlined an approach that can lead in principle to a synthetic model of activities in real data that extends the approach in (Eubank et al., 2004a). Significant work remains to be done. For instance, the clustering analysis in the previous section is most natural on the set of "labels," the refinements of activity types we highlighted. Such an analysis (combined with community inference methods) would hopefully result in a hierarchical random model that can represent locality. The existence of fast generation algorithms for our network models (similar to the work in (Eubank et al., 2004b)) is an interesting algorithmic problem. Finally, the accuracy of random models of activity data should be verified more thouroughly.

## ACKNOWLEDGMENTS

This work has been supported by the Department of Energy under contract W-705-ENG-36 and by the Department of Homeland Security under the National Infrastructure Simulation and Analysis Center (NISAC) program.





(a) (b)

FIGURE 6: (a) Clustering by pairs of PCA components, (b) The first three PCA components

TABLE 8: Joint activity counts in sample data

| | Home | Work | School | Retail | Other | Serve p. | College |
|---|---|---|---|---|---|---|---|
| Home | 4356 | 2000 | 886 | 1206 | 2195 | 520 | 179 |
| Work | 0 | 2001 | 56 | 400 | 757 | 260 | 88 |
| School | 0 | 0 | 886 | 123 | 424 | 75 | 0 |
| Retail | 0 | 0 | 0 | 1207 | 715 | 192 | 38 |
| Other | 0 | 0 | 0 | 0 | 2204 | 319 | 69 |
| Serve passenger | 0 | 0 | 0 | 0 | 0 | 520 | 24 |
| College | 0 | 0 | 0 | 0 | 0 | 0 | 179 |

## REFERENCES

Barrett et al. (2004). A tale of two cities. (unpublished manuscript).

Barrett, C. L., Eubank, S. G., and Smith, J. P. (2005). If smallpox strikes Portland ... *Scientific American*.

Eubank, S., Guclu, H., Kumar, V. S. A., Marathe, M., Srinivasan, A., Toroczkai, Z., and Wang, N. (2004a). Modeling disease outbreaks in realistic urban social networks. *Nature*, 429:180–184.

Eubank, S., Kumar, V. S. A., Marathe, M., Srinivasan, A., and Wang, N. (2004b). Structural and algorithmic aspects of massive social networks. In *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 711–720.

Eubank, S., Kumar, V. S. A., Marathe, M. V., Srinivasan, A., and Wang, N. (2006). Structure

TABLE 9: Joint probabilities of main activity types

| | Home | Work | School | Retail | Other | Serve p. | College |
|---|---|---|---|---|---|---|---|
| Home | 1.000 | 0.459 | 0.203 | 0.277 | 0.504 | 0.119 | 0.041 |
| Work | 0.459 | 0.459 | 0.013 | 0.092 | 0.174 | 0.060 | 0.020 |
| School | 0.203 | 0.013 | 0.203 | 0.028 | 0.097 | 0.017 | 0.000 |
| Retail | 0.277 | 0.092 | 0.028 | 0.277 | 0.164 | 0.044 | 0.009 |
| Other | 0.504 | 0.174 | 0.097 | 0.164 | 0.506 | 0.073 | 0.016 |
| Serve passenger | 0.119 | 0.060 | 0.017 | 0.044 | 0.073 | 0.119 | 0.006 |
| College | 0.041 | 0.020 | 0.000 | 0.009 | 0.016 | 0.006 | 0.041 |

TABLE 10: Conditional probabilities of main activity types

| | Home | Work | School | Retail | Other | Serve p. | College |
|---|---|---|---|---|---|---|---|
| Home | 1.000 | 1.000 | 1.000 | 0.999 | 0.996 | 1.000 | 1.000 |
| Work | 0.459 | 1.000 | 0.063 | 0.331 | 0.343 | 0.500 | 0.492 |
| School | 0.203 | 0.028 | 1.000 | 0.102 | 0.192 | 0.144 | 0.000 |
| Retail | 0.277 | 0.200 | 0.139 | 1.000 | 0.324 | 0.369 | 0.212 |
| Other | 0.504 | 0.378 | 0.479 | 0.592 | 1.000 | 0.613 | 0.385 |
| Serve passenger | 0.119 | 0.130 | 0.085 | 0.159 | 0.145 | 1.000 | 0.134 |
| College | 0.041 | 0.044 | 0.000 | 0.031 | 0.031 | 0.046 | 1.000 |

of social contact networks and their impact on epidemics. In Abello, J. and Cormode, G., editors, *Discrete Models in Epidemiology*, volume 70 of *DIMACS-AMS Series in Discrete Mathematics and Theoretical Computer Science*. American Mathematical Society.

Ghislandi, A. C. (1990). CATS 1990 household travel survey, technical documentation for the household, person, and trip files. Technical report, Chicago Area Transportation Study Information Services Division, Chicago, IL.

Newman, M., Barabasi, A. L., and Watts, D., editors (2006). *The Structure and Dynamics of Networks*. Princeton University Press.

Houston-Galveston Area Council (2001). Regional travel models. 1995 model validation and documentation report. Technical report, http://www.h-gac.com/HGAC/home/Default.htm.

U.S. Department of Transportation (2001). The national household travel survey (NHTS). Technical report, Center for Transportation Analysis, Oak Ridge National Laboratory, http://nhts.ornl.gov/2001/index.shtml.

Timmermans, H., editor (2005). *Progress in Activity-Based Analysis*. Elsevier.

TABLE 11: Degree of dependence between main activity types

| | Home | Work | School | Retail | Other | Serve p. | College |
|---|---|---|---|---|---|---|---|
| Home | 0.000 | 0.000 | 0.000 | −0.001 | −0.004 | 0.000 | 0.000 |
| Work | 0.000 | 0.541 | −0.396 | −0.128 | −0.116 | 0.041 | 0.032 |
| School | 0.000 | −0.175 | 0.797 | −0.101 | −0.011 | −0.059 | −0.203 |
| Retail | 0.000 | −0.077 | −0.138 | 0.723 | 0.047 | 0.092 | −0.065 |
| Other | −0.002 | −0.128 | −0.027 | 0.086 | 0.494 | 0.107 | −0.120 |
| Serve passenger | 0.000 | 0.011 | −0.035 | 0.040 | 0.025 | 0.881 | 0.015 |
| College | 0.000 | 0.003 | −0.041 | −0.010 | −0.010 | 0.005 | 0.959 |

TABLE 12: Statistical correlation of main activity types

| | Home | Work | School | Retail | Other | Serve p. | College |
|---|---|---|---|---|---|---|---|
| Home | 0 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |
| Work | 0 | 1.000 | −0.402 | −0.159 | −0.235 | 0.030 | 0.013 |
| School | 0 | −0.402 | 1.000 | -0.156 | -0.028 | -0.054 | −0.105 |
| Retail | 0 | −0.159 | −0.156 | 1.000 | 0.107 | 0.076 | −0.030 |
| Other | 0 | −0.235 | −0.028 | 0.107 | 1.000 | 0.079 | −0.050 |
| Serve passenger | 0 | 0.030 | −0.054 | 0.076 | 0.079 | 1.000 | 0.009 |
| College | 0 | 0.013 | −0.105 | −0.030 | −0.050 | 0.009 | 1.000 |

# ORGANIZATIONAL CASE STUDY: THEORY AND MATHEMATICAL SPECIFICATIONS FOR AN AGENT BASED MODEL (ABM)

W.F. LAWLESS,[*] Paine College, Augusta, GA
J.C. WOOD, U.S. Army, Fort Gordon, GA
S. EVERETT and W. KENNEDY, Naval Research Laboratory, Washington, DC

## ABSTRACT

This case study found that a military medical department research center (MDRC) with access to advanced information technology was struggling to determine the quality of the residents it trains and to measure their scholarly productivity. Yet snapshots in time and inevitable researcher biases restrict case studies to hindsight rather than proactive sources of organizational solutions. Case studies guided by theory, however, have illuminated and tested many of the organizational principles that have been discovered. Unlike simple Newtonian mechanics, interactions among organizational members are interdependent with the interviews that a case study collects to establish a base line. Consequently, case study measurements collapse organizational interactions, losing enough information to elude a science of the fundamental interaction. But first principles can be discovered if the uncertainty left from the collapse of interactions can be predicted and exploited for key interdependent variables: planning and execution; or resources and time.

## INTRODUCTION

Organizational theory has failed to produce predictable (Pfeffer and Fong, 2005) or replicable results (Weick and Quinn, 1999). Traditional organizational theory, generally based on methodological individualism (MI; e.g., game theory; in Nowak and Sigmund, 2004), assumes that information from individuals is stable and accessible (Baumeister, 1995), making an organization into a rational aggregation of the contributions from its individual members. In what we believe is related, many agent-based models (ABMs) are based on MI with the same assumption about rational aggregation, but discounting the value of prediction as a consequence: “the value of a [computational] model is not prediction but insight” (Bankes, 2006). Defining rationality as “normative consistency,” Shafir and LeBoeuf (2002) concluded that neither average humans nor experts make consistent choices, preferences, or justifications, undercutting the traditional model of rationality. But to successfully operate an autonomous computational organization in the field, a rational process of prediction is necessary.

In contrast to MI, we have had limited success by importing the quantum uncertainty relations as first suggested by Bohr (1955) and Heisenberg (1958) to address interdependent uncertainties in human social interaction (Lawless et al., 2000) and to predict decision-making among human organizations in the field (Lawless et al., 2005) and more recently in the laboratory (Lawless et al., 2006b). With our limited success, we have begun to develop metrics based on the “measurement paradox” (Lawless and Grayson, 2004).

[*] *Corresponding authors’ addresses*: W.F. Lawless, Paine College, 1235 15th Street, Augusta, GA; email: lawlessw@mail.paine.edu; J.C. Wood LTC, U.S. Army, Fort Gordon, GA; email: Joseph.C.Wood@usarmy.mil.

The paradox indicates that measuring an interaction or organization collapses the existence of its interdependent information into strictly classical information that cannot be re-aggregated to reconstruct the organization (Levine and Moreland, 1998; 2004), nor apparently even for the individual—despite more than 30 years of research, no better than a weak link has been confirmed between self-esteem and actual performance at school or in the workplace (Baumeister et al., 2005). Surprisingly, the measurement paradox suggests that the collapse of interdependent information can be exploited to favor one of two interdependent states in our mathematical model of interdependence to produce predictable outcomes under certain rather extreme conditions, such as the difference between consensus (CR) versus majority rule (MR) decision processes in organizations (Lawless et al., 2005): We have predicted and found that CR leads to less concrete decisions less welcomed by an organization's customers, but at lower energy expenditures that take longer to process; in contrast, MR leads to more practical decisions more welcomed by customers, but with more conflict and energy expended that quicken decisions.

The relationship between decision processes and organizations is itself complex, especially for CR. The purpose of CR is to convert the neutrals in a group into active individual participants (Bradbury et al., 2003). However, the process in CR that suspends criticism of beliefs no matter how bizarre lends itself to being hijacked: "The requirement for consensus in the European Council often holds policy-making hostage to national interests in areas which Council should decide by a qualified majority." (WP, 2001, p. 29). Organizations are primarily hierarchical and governed by a single leader under command decision-making (CDM); the link to CR becomes more obvious under the control of multiple leaders (e.g., the crisis at Unilever prior to 2005; the current crisis at Europe's aerospace EADS group), however, single leaders using intimidation or even violence can convert an organization or system into a quasi CR process that stifles criticism; e.g., Germany's response to Hitler's "Night of the Long Knives", in 1934 (Benz, 2006, p. 54). Counterintuitively, when competition can be managed to preclude conflict, we have found that the most robust consensuses are derived from competition (i.e., MR); more learning occurs under competition (Dietz et al., 2003); and the more competitive is a team, the greater the cooperation among its members.

Since our laboratory studies, we have further exploited the paradox to propose the first mathematical set of interdependent metrics designed to measure the real-time performance for a system of military forecasters in the field (Lawless et al., 2006a). We recently revised and extended these metrics to analyze the reorganization of the Management Information Service Center (MISC) at a major university in Europe to further establish organizational principles that were then used to reverse model terrorist organizations (Lawless et al., 2006b). Mindful that a case study reflects a static snapshot in time, exposing our results to confirmation bias (Eagly and Chaiken, 1993) which we countered with a theoretical foundation directed at four interdependent variables (in Figure 1, planning and execution; energy and time), we found that MISC and its university had been operating without a structured business model (BM). The lack of a focused BM for the university had led to a disorganized assemblage of faculty, staff and students that discouraged innovation, promoted administrative malfeasance, impeded student progress and faculty research, and significantly reduced opportunities for MISC and its university. We concluded that a loose aggregation in the limit approaches a CR process in that less information is processed by the organization than some of its members but on an ad hoc basis, consequently precluding organizational learning and change in response to environmental perturbations (Dietz et al., 2003).

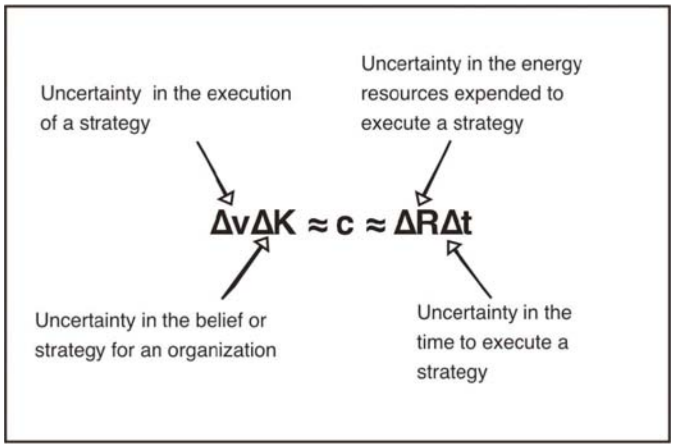

Uncertainty in the execution of a strategy
Uncertainty in the energy resources expended to execute a strategy
ΔvΔK ≈ c ≈ ΔRΔt
Uncertainty in the belief or strategy for an organization
Uncertainty in the time to execute a strategy

**FIGURE 1** The measurement problem was derived from case studies of the Department of Energy's (DOE) program to cleanup the widespread waste and environmental contamination at its sites (Lawless and Grayson, 2004; in the above equations, "*c*" is an unknown constant). Here it was applied to Arcelor Steel's response to Mittal Steel's hostile merger bid for Arcelor in 2006: Strategy: How focused was Arcelor's strategy to protect its resources; operationally, how widespread was the consensus among its members and stockholders in support of its strategy? Execution: How motivated or effective were the members-stockholders to execute Arcelor's plan; operationally, how many supporters were collected per unit of time as a result of the strategy? Resources: How effective was the strategy (plan or algorithm) in growing Arcelor by gathering new organizational resources (increasing free energy to increase choices); secondarily, how efficient was the strategy at saving existing resources (contracting by reducing wastes to increase predictability)? Time: How time consuming was implementing the strategy; secondarily, how quickly was feedback captured by Arcelor's leadership to refine or tune its plan; alternatively, how opportunistic was the plan?

## OVERVIEW OF MDRC

In 2006, a new director assumed command of a military Medical Department Research Center (MDRC) at a regional military medical hospital. (Note: Fictional names have been used in this case study.) MDRC supports clinical and basic research for its staff and all hospital personnel including family practitioners, internal medicine, general and orthopaedic surgery, and dentistry (endodontics, peridontics, oral maxillofacial, orthodontics, and nurse anaesthesiology). In addition to providing basic research support for the hospital, MDRC is responsible to teach the fundamentals of experimental research to the hospital's medical residents (categorical residents working within a specialty, residents rotating among the hospital's different specialties, and transitional interns; in JWO, 2005, p. 4), provide continuing education for more experienced care providers, and train dentists in research methods. One of the goals of the training by MDRC is to help the hospital's Graduate Medical Education (GME) candidates become certified by their respective American specialty boards, in what is becoming a matriculation requirement for many training programs. GME trainees are supervised by about 150 teaching staff members who are all board certified in their respective specialties (JWO, 2005, p. 4).

Over the remaining two years of his administration, the Chief, MDRC, wants to establish Metrics of Effectiveness (MOE) to measure his organization's success and to craft a plan with MOE's to improve the performance of scholarly activities (i.e., plan or Business Model, BM). Some resident trainees start their research rotation prepared with a line of investigation derived from their own interests, collaboration with peers, or previous mentors. But if they are not presently working in research or have a research interest, a mentor is assigned to them by MDRC staff. One problem with using MOE's is that much of the research proposed in the MDRC protocols has lasted or may last for a number of years before scholarly products can be published, whereas other research protocols may last under one year, giving the less complex Protocols an advantage in the generation of scholarly products. As the complexity of a Protocol increases, the time necessary to complete a program of research also increases. The Chief wants to increase the complexity of protocols but remain able to measure the impact of complexity on scholarly productivity. (For the entire case study along with a full list of all of the recommendations, see Lawless et al., 2006c).

## BRIEF SUMMARY OF THE RESULTS AND RECOMMENDATIONS FOR MDRC

### Planning uncertainty for the organization's Business Model (BM, or $\Delta K$)

In the MDRC Chief's opinion, attracting quality residents will require an increase in the quality of publications and in receiving help from other Division Chiefs, plus having a stronger program. Then better graduates will begin to attract other quality candidates, adding credibility to the BM.

A focused, concrete plan of action (BM) should be designed by MDRC to gain wide support among its staff, mentors, and trainees (possibly circulated in draft, but higher Command's support is crucial). The MDRC Chief observed that the support provided by his organization is satisfactory from all accounts, but that it should be better organized and more focused (INC, 2006, p. 3-4). The revised BM ($\Delta$BM) should permit everyone to work under the same roof (INC, 2006, p. 7), gain more extramural funds, improve scholarly productivity, and keep education as the primary goal of MDRC (INC, 2006, p. 5). As part of an innovation plan, new resources or funds must be sought (INC, 2006, p. 5; $-\Delta A$) from State, Federal and industrial sources. If successful, the plan should produce a qualitative shift among mentors and GME trainees under the new system; trainees will also learn new professional techniques that should help them to find better jobs afterwards.

Currently, as the complexity of a protocol increases, the time necessary to complete a program of research also increases, implying CR and a lack of command decision-making

(CDM) control. The new plan must focus organizational resources to produce high quality research executed under a sense of competitive urgency. The goal should not be to seek complexity but instead publishable research that helps to drive more and more competitive scholarly productivity and the search for extramural funds. Complexity should be a byproduct of the plan.

## Planning Execution ($\Delta v$)

The goal of executing the BM should be to increase the number of customers per period (new grant funds; new trainees; and new recognitions of quality).

The lack of focus at MDRC has primarily arisen from an internal lack of competition among its organization's members and also its customers (the trainees) in the execution of its current BM. However, a lack of focus again suggests that an implicit CR exists to block the execution of a revised BM. Resistance to implementing a new BM can be anticipated ($exp(-\Delta A / <A>)$), implying that barriers must be anticipated and overcome, as well as an average rise in activation ($<A>$) that should be exploited to increase the rate of execution.

With support from his fellow Division Chiefs, colleagues, and staff ($n_D$), eventually the barriers that arise can be overcome among new mentors and new trainees ($n_N$) around which a new culture should be encouraged to become established and grow.

The MDRC and hospital staff should be educated to understand the need for regular professional training to improve research performance, especially the quality and the quantity of scholarly productivity; and the need for new information channels to distribute technical information about research opportunities. Numerous messages about the change ($v_L$) should be given to trainees in seminars provided by MDRC and designed to revise the culture to match the new BM ($\sigma_{RL}$)

## Resource uncertainty (free energy, or $-\Delta A$)

The goal of the plan of action (BM) is to maximize the resources (free energy) available to MDRC to execute its BM in the minimum of time. Such a BM, however, will likely reduce innovation, but practices instituted to seek innovation can offset this shortfall (e.g., including in the BM a strategy to continually seek new partners to obtain extramural grants, along with MOE's of grant progress)

One measure of progress is the level of teamwork across the organization (MDRC staff, mentors and trainees) to increase the competitiveness of MDRC's performance in completing its protocols as an indirect measure of the overall effort by its research teams; however, teamwork must not be simply commanded, but encouraged as a part of the competitive process and demonstrated to work.

As innovation increases from the gathering and expenditure of resources (free energy, $-\Delta A$) derived from the discovery of new resources (new trainees and new faculty become more attracted by new skills and the initiation of new industrial-state-federal projects) and a reduction of costs (less waste), planning complexity increases correspondingly as the ability to direct these freed funds to a random exploration of new projects for MDRC and the hospital with success determined by a reduced effort (practice effects; stochastic resonance) in discovering new sources of free energy.

## Time Uncertainty

The goal is to reduce the time to execute a BM, gain new resources, and discover new opportunities.

As the available resources are increasingly directed by MDRC to the completion of existing projects with the ultimate goal of freeing resources (free energy, $-\Delta A$) for new projects, the average time to complete and execute existing projects should decrease. At the same time, if and only if an innovation circuit has been established, the time to innovate should decrease correspondingly as new opportunities arise.

As new opportunities become available and exploited by MDRC, they will provide new opportunities to trainees, adding incentives that should further improve the quality of future trainee candidates.

## SUMMARY

The present case study of MDRC indicated that fragmentation among its processes and researchers had reduced control over its future plans. Fragmentation in an organization is associated with increased innovation at the individual level (e.g., Benz, 2006), but at the expense of the organization. Enforcing cooperation, however, can be counterproductive unless it is managed by organizational members and leaders together (Gürek et al., 2006). Successful organizations are constructed by Command Decision-Making (CDM) into becoming entangled as centers of cooperation (Lawless et al., 2000; Lawless et al., 2006a) that execute quickly a focused BM to gain wide support, new resources with minimal waste, but also that tend to reduce innovation from marginalizing the available knowledge as a consequence of making the organization, in this case MDRC, more competitive in its marketplace. This reduced innovativeness can be countered with a BM of practices to increase innovation.

The MISC case study also served to mathematically extend our model of organizations with improved specifications estimated for a nonlinear agent-based model. In this new case study, we continued to develop theory, tested it in a case study of a military Medical Department Research Center (MDRC), and, based on our recommendations to MDRC, refined our proposed model of an ABM computational organization from the results of theory and this latest case study (see Table 1). For the next stage of our research, we plan to pursue a case study of a department at a national military research laboratory (MRL) in its attempts to secure more extramural research funds.

**TABLE 1** Specifications for an ABM of Figure 1 as derived from the MDRC Case Study

| | CR | MR |
|---|---|---|
| Beliefs, $\Delta K$ | • By definition, consensus is achieved: $CR_{time} = \sum_i^N \Delta K_i = min$ $CR_{agents\text{-}org} = \sum_i^N \Delta K_i = max$ • More risk perceptions • Algorithm steps = *f(N)* • Probability of new *K* is low | • By definition, conflict is initiated: $MR_{time} = \sum_i^N \Delta K_i = max$ $MR_{agents\text{-}org} = \sum_i^N \Delta K_i = min$ • Fewer risk perceptions • Algorithm steps = *f(D=N-M)* • Probability of new *K* is high |
| Execution, $\Delta v$ | • Less execution steps per unit time • Produces *N* customers (fewer) • Least customer satisfaction. | • Most execution steps per unit time • Produces *N* customers (most) • Greatest customer satisfaction. |
| Energy, $\Delta E$ | • Least *E* saved • Entropy high • Least practical steps per unit of time • Most agents at $E_0$ (resonance), fewer at $E_1$ (drivers), fewest at $E_2$. | • Most *E* saved • Entropy low • Most practical steps per unit of time • Fewest agents at $E_0$, most at $E_1$ (attention), fewer at $E_2$ (drivers) |
| Time, $\Delta t$ | • More time required to solve a problem (exponential time) • Time = *f(N)* | • Less time required to solve a problem (polynomial time) • Time = *f(D)* |

## REFERENCES

Bankes, S.C. (2006, 6/23), Luncheon Invited Speaker: “Robust inference in computational social science, NAACSOS-06, Notre Dame.

Baumeister, R. F., Campbell, J.D., Krueger, J.I., and Vohs, K.D. (2005, January). Exploding the self-esteem myth. *Scientific American*.

Benz, W. (2006), A concise history of the Third Reich. Dunlap, T. (Tr.). UCLA Press: Berkeley.

Bohr, N. (1955). Science and the unity of knowledge. *The unity of knowledge*. L. Leary. New York, Doubleday: 44-62.

Bradbury, J., A. Branch, K.M., and Malone, E.L. (2003). An evaluation of DOE-EM Public Participation Programs (PNNL-14200). Richland, WA, Pacific Northwest National Laboratory.

Conzelmann, G., Boyd, R. Cirillo, V. Koritarov, C.M. Macal, M.J. North, P.R. Thimmapuram, and T. Veselka (2004). Analyzing the Potential for Market Power Using an Agent-Based Modeling Approach: Results of a Detailed U.S. Power Market Simulation. International Conference on Computing, Communication and Control Technologies, Austin, TX.

Dietz, T., Ostrom, E., and Stern, P.C. (2003). “The struggle to govern the commons.” *Science* **302** 1907.

DWA (2004), MDRC Clinical Investigation Program, RCS MED-300 (R).

Eagly, A.H. and Chaiken, S. (1993), *The psychology of attitudes*, Harcourt: Fort Worth.

Gürek, Ö., Irlenbusch, B., and Rockenbach, B. (2006). “The competitive advantage of sanctioning institutions.” *Science* **312**: 108-111 (see accompanying Perspective: Henrich, J., 60-61).

Heisenberg, W. (1958/1999). Language and reality in modern physics. *Physics and philosophy. The revolution in modern science*. New York, Prometheus Books, Chapter 10, pp. 167-186.

INC (2006), Interview Notes dated June 2, 2006 and subsequent informal communications.

JWO (2005), MDRC Clinical Investigation Program, RCS MED-300 (R).

Kelley, H.H. (1992). Lewin, situations, and interdependence. *J Social Issues*, 47, 211-233.

Lawless, W.F. (2006). Medical Department Research Center (MDRC) case study. Unpublished source material.

Lawless, W.F., Castelao and Abubucker (2000), Conflict as a heurestic in the development of an interaction mechanics, *Conflicting agents: Conflict management in multi-agent systems*. C. Tessier, L. and Muller, H.J. (Eds.), Boston, Kluwer. 1: pp. 279-302.

Lawless, W. F., and Grayson, J.M. (2004). A quantum perturbation model (QPM) of knowledge and organizational mergers. *Agent Mediated Knowledge Management*. L. van Elst, and V. Dignum. Berlin, Springer (pp. 143-161).

Lawless, W. F., Bergman, M., and Feltovich, N. (2005). “Consensus-seeking versus truth-seeking.” *ASCE Practice Periodical of Hazardous, Toxic, and Radioactive Waste Management* **9(1)**: 59-70.

Lawless, W. F., Bergman, M., and Feltovich, N. (2006a). A physics of organizational uncertainty. Perturbations, measurement and computational agents. *Computational Economics: A Perspective from Computational Intelligence*. Shu-Heng Chen, Lakhmi Jain, and Chung-Ching Tai. Hershey, PA, Idea Books, 268-289.

Lawless, W.F., Bergman, M., Louca, J. and Kriegel, N.N. (2006b, forthcoming). A quantum metric of organizational performance: Terrorism and counterterrorism. Computational and Mathematical Organizational Theory.

Lawless, W.F., Wood, Joseph, Everett, Stephanie and Kennedy, William (2006c, submitted to a journal for review), Theory driven case study of metrics for a military medical research center.

Levine, J.M. and Moreland, R.L. (1998), Small groups, In Gilbert et al., *Handbook of Social Psychology*, pp. 415-469, McGraw.

Levine, J. M., Moreland, R.L. (2004). “Collaboration: The social context of theory development.” *Personality and Social Psychological Review* **8(2):** 164-172.

Pfeffer, J., and Fong, C.T. (2005). “Building organization theory from first principles: The self-enhancement motive and understanding power and influence.” *Organization Science* 16: 372-388.

Shafir, E., and LeBoeuf, R.A. (2002). “Rationality.” *Annual Review of Psychology* **53:** 491-517.

Weick, K. E., and Quinn, R.E. (1999). “Organizational change and development.” *Annual Review of Psychology* **50:** 361-386.

Wendt, A. (1999). *Social theory of international politics*. Cambridge University Press.

WP (2001). White Paper. European governance (COM (2001) 428 final; Brussels, 25.7.2001). Brussels.

Saturday, September 23, 2006

## Social Simulation Applications

# National Security

# SENDING AGENTS TO WAR

M. T. K. KOEHLER,[*] MITRE Corporation, McLean, VA
P. S. BARRY, MITRE Corporation, McLean, VA
T. E. MEYER, MITRE Corporation, McLean, VA

## ABSTRACT

As agent-based modeling has grown in popularity in the academic community as a research tool for the social sciences, it has also gained popularity with the governmental sector. However, the needs of the governmental sector are quite different than the academic community. With systems to acquire or large-scale public policy decisions to be made there is a strong push to know the simulations backing a particular course of action are "right." In this regard customer or sponsor buy-in is very important. Buy-in implies a certain level of familiarity and comfort with the process that was used to create the recommendations. This paper will examine the use of agent-based modeling in the US Department of Defense. It will focus on military verification and validation, (V&V) of agent-based models, particularly for their application within a military context.

**Keywords:** Agent Based, Combat, Human Behavior, Verification, Validation

*"What is not surrounded by uncertainty cannot be the truth."*
~ Richard Feynman

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

## INTRODUCTION

In 1981 Francis Kapper wrote, "The most appropriate and valid objectives for using war games and simulations within the DoD [Department of Defense] context are to: better understand complex phenomena, identify problems, evaluate alternatives, gain new insights, and broaden one's perspectives. The least valid or appropriate objectives for using war games and simulations are to predict combat/crisis outcomes or control broad and highly complex programs" (as quoted in Hartley 1997).

Simulations within the DoD have been used for years as training aids (e.g., flight simulators[i]), to understand the effects of new tactics or techniques (constructive simulations such as the Joint Training Confederation[ii]) or for experimentation such as exploring how to defeat time sensitive targets[iii]. However, as the extent of military operations expands into non-kinetic arenas such as stability and support operations, the traditional closed form detailed simulations are proving inadequate for training, experimentation and analysis.

To address this shortfall, analysts are beginning to use agent-based modeling (ABM) and simulation more frequently in the DoD. Publications devoted to this practice can now be found

[*] *Corresponding author address:* Matthew Koehler, MITRE Corporation, H305, 7515 Colshire Drive, McLean, VA, 22102; *email:* mkoehler@mitre.org.

(e.g. Ilachinski 2004, Horne and Johnson 2002, 2003) and more studies using ABMs are described in the proceedings of the Military Operations Research Society (www.mors.org). However, there is a consistent barrier to widespread use of agent based modeling, that of traditional verification and validation (V&V). Modeling in the DoD traditionally undergoes a rigorous V&V cycle based upon a requirements document and the "objective reality" being modeled. For example, producing a simulation of a missile is based on physics and to the extent the physics involved is known and understood the simulation can be made increasingly more accurate. Even combat has been seen as a set of equations (in this case Lanchester equiations).

However, even physical, equation based systems can begin to move beyond our ability to model them fully. Weather, for example, is well understood and most of the underlying physical principles are known and measured. Even with this weather models are not completely accurate and are better described as producing insight into weather patterns rather than predicting them. It is in this light the authors feel ABMs should be considered for use in the military context. ABMs offer insight in to the potential dynamics of a system or scenario that other modeling and simulation capabilities will not be able to capture. This is the case because ABMs are often used on problems as a technique of last resort, typically on open systems and ill-defined systems that cannot be easily formalized in a system of equations.

Therefore, we assert traditional, static V&V approaches to ABMs are inappropriate, particularly when the simulation is designed to represent human behavior. As discrete closed form models of human behavior have yet to be developed, expecting *a priori* definitions of human behavior is unrealistic. This is wholly different from the situation where physics and well understood phenomena can be modeled and then checked. By and large, ABMs are simulations of open, adaptive systems that should be used for insight. Using a model for insight rather than prediction engenders a different V&V process—not weaker or less rigorous but merely different.

## AN OVERVIEW OF MILITARY V&V

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). The Defense Modeling and Simulation Office (DMSO) defines these concepts as follows: "Verification: The process of determining that a model implementation and its associated data accurately represent the developer's conceptual description of specifications. Validation: The process of determining the degree to which a model and its associated data provide an accurate representation of the real world from the perspective of the intended uses of the model. [DoDI 5000.61]" (http://vva.dmso.mil/). DMSO continues this discussion by incorporating, almost as an aside, the interesting concept of Credibility: should the model or simulation be trusted? This issue of credibility will be discussed later.

As part of their mission DMSO produced and maintains the VV&A Recommended Practices Guidelines (www.vva.dmso.mil) that spell-out how the military approaches and understands the practice of VV&A. Since ABMs are often used to study human systems, the most applicable DMSO document from our perspective is: "Validation of Human Behavior Representations." This DMSO document describes the process of validating Human Behavior Representation (HBR) and the tools and processes involved there in. Within in this document DMSO defines HBR very broadly: "All HBRs model the behavior of people at some level." This is elaborated on by stating that HBRs may be a part of a human (e.g. their hands), a person, a group of people, organization, or group of organizations. In a general sense a HBR is simply some simulation entity that receives inputs and, using its internal states and knowledge base, creates outputs in an attempt to mimic some aspect of human behavior. In its most basic sense this is generally analogous to an agent in a typical ABM. DMSO contends that validating HBRs is difficult because of their inherent complexity driven by HBR adaptation/learning within a single run of the simulation, feedback within the HBR, heterogeneity among HBRs within the model, and the degrees of freedom associated with a HBR focused simulation.

DMSO sees validation of HBRs as requiring four general steps: develop a requirements document, define referents that can be used to determine accuracy, assess the capabilities of the HBRs, and finally compare the capabilities to the requirements to determine the fitness of the HBRs. These steps are usually accomplished via a Subject Matter Expert (SME) doing face validation of the HBRs. However, DMSO considers this the least reliable method of validation. This is due in part to the large parameter spaces associated with these models, as well as the highly complex and nonlinear relationship between inputs and outputs making generalization about performance from one set of input parameters to another problematic. Furthermore, SMEs can be biased when thinking about a simulation, may poorly understand how the simulation works, may only be familiar with a small part of the information necessary to validate the HBRs, etc. However, when a model employs learning and other well studied behavior/or performance modules SMEs can be instrumental in comparing the model instantiation to the current thinking in the field, and, therefore, make an assessment as to how well the HBRs captures it.

DMSO also stresses the importance of empirical data as a potential referent for V&V. However, this comparison is potentially nontrivial. Data may be anecdotal or incomplete. Data may be from poorly controlled experiments calling its reliability in to question. The available data may only be tangentially related to the HBR, thus, also limiting its utility. This is not an exhaustive list, and the issues have not been lost on the ABM community as will be mentioned below.

The aforementioned referents are used to establish six categories of correspondence: Domain, essentially face validation of the model by a SME; Sociological, the HBR behaves appropriately at the group or aggregate level; Psychological, the HBR behaves appropriately at the individual level; Physiological, the HBR contains the correct physiologic responses; Computational, the HBR corresponds to human computational limitations; and Physical, the HBR corresponds to the physical limitation of a human. The perfect HBR will have correspondence in all six categories, though DMSO concedes that no such HBR exists. In light of the non-attainability of perfection, correspondence in more categories is clearly better than in fewer. Furthermore, correspondence will be easier to demonstrate in categories like the physical limitation of humans than in the psychological functioning of humans.

## V&V IN THE AGENT-BASED MODELING COMMUNITY

The issue of validation of ABMs is not unique to the military context, and the larger ABM community also is dealing with it. Consequently, there are many different tools and ideas regarding V&V. In the interest of space we will only discuss a few to highlight the general concepts. In particular we will consider how the V&V of ABMs centers on SME evaluation, comparing the model to data and/or to other models, or intelligently sampling the parameter space of the model to better understand its behavior.

In ABM-based Economics Fagiolo et al (2006) characterize the difficulty of the V&V of ABMs in this way: "…an important aspect, recognized by AB [agent based] modelers themselves, concerns the perceived lack of robustness in AB modeling. This threatens the AB research enterprise as a whole." These authors continue by discussing four general features of the ABM community that they believe do a very good job of characterizing the reasons why validation continues to be an issue. First, there is no core set of theories or models that can be applied to various research areas. Of course, that is not to say that there are not common principles of ABM (heterogeneity, explicit spatial qualities, non-ergodicity, etc.) but these do not constitute an underlying theory. Second, the idiosyncrasies contained within models makes cross model comparisons very difficult—even when they purport to be modeling the same phenomena. Third, there is a lack of standardized tools and methods for creating and analyzing ABMs. Finally, the relationship between dynamic ABMs and static empirical data is nontrivial, which results in difficulty in determining how well the ABM represents or relates to the empirical data from which it is built (Fagiolo 2006).

Two of the authors have utilized SME face validation on a number of occasions, some of which are described in (Meyer 2005 and Koehler 2004). SMEs can play an enormously important role in the creation and V&V of a model. While face validation has its faults, it is often the best and perhaps only option.. We have often spent a great deal of time trying to explain the abstractions within our models to SMEs with varied success. This understanding is critical so they may better understand what they are trying to validate. Without this understanding the utility of their assessment is questionable—but it can be extremely difficult to explain fully an ABM to a SME who may have no ABM experience.

A more quantitative approach to validation is Active Nonlinear Tests (ANTs). ANTs is a methodology designed to "break" a simulation. In essence, a genetic program is created to make small changes to input parameters and then measure the associated change in output values. The fitness function for the ANTs relates small changes to the input parameters to how large a change was produced in output values (Miller 1998). The original application was not on an ABM but it is the authors' contention that the methodology is appropriate.

As all models are abstractions, the community has tried to ameliorate the effects of these assumptions by the use of multiple models. Docking is the process of aligning simulations. Docking has been proven to be a very good way to test for the effect of and presents of "hidden" or subtle assumptions inherent within all models. There are three levels of docking: identity, where the simulations produce identical results; distributional, where simulations produce statistically indistinguishable results; and relational, where the simulations "behave" in qualitatively equivalent manners but are statistically dissimilar (Axtell 1996).

Perhaps the most straightforward method of validation is to check the results against empirical data. Axtell (2005) articulated four types of model agreement: Level 0, refers to the simulation simply functioning as bug free code; Level 1 refers to macro-level qualitative agreement—the model displays aggregate behavior similar to the dynamics seen within the data; Level 2 is macro-level quantitative calibration; and finally, Level 3 is micro-level quantitative estimation. Axtell's taxonomy describes a metric for the relationship between the ABM in question and the data available to it for development and testing. This approach allows the validator to use the levels as a way of judging and describing the extent of V&V of an ABM.

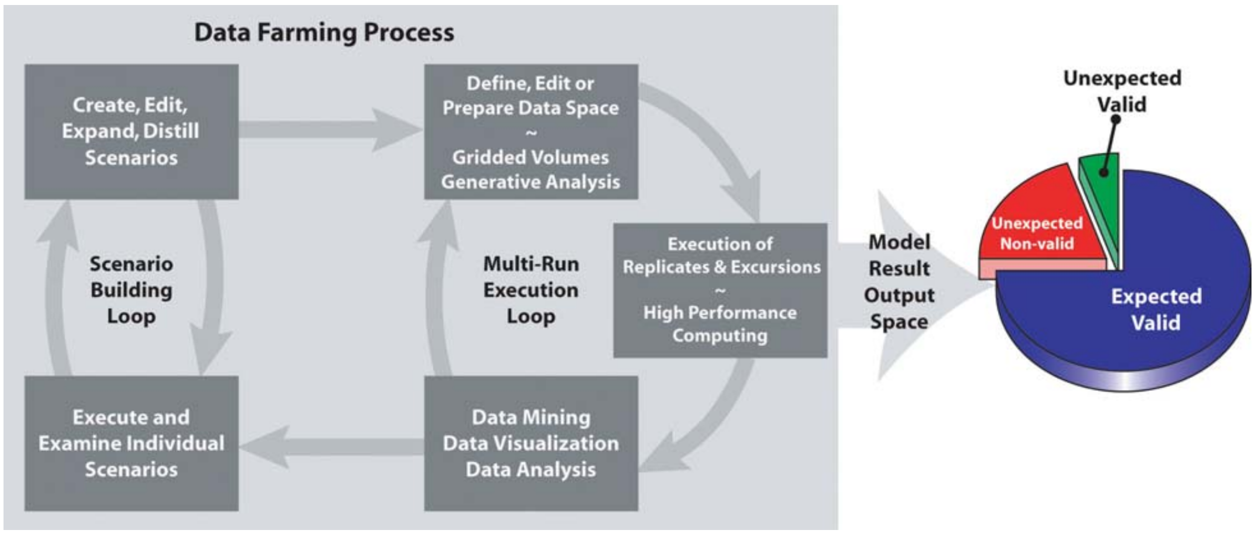

Data Farming Process
Create, Edit, Expand, Distill Scenarios
Define, Edit or Prepare Data Space ~ Gridded Volumes Generative Analysis
Scenario Building Loop
Multi-Run Execution Loop
Execution of Replicates & Excursions ~ High Performance Computing
Execute and Examine Individual Scenarios
Data Mining Data Visualization Data Analysis
Model Result Output Space
Unexpected Valid
Unexpected Non-valid
Expected Valid

**Figure 1: Data Farming Process and Insight**

A more integrated approach, though not unique to AMBs, is *Data Farming* (Brandstein and Horne, 1998, Horne and Meyer 2004). Data Farming is a process developed for Project Albert, a USMC program, aimed at supporting military decision-makers. It is the process of running models many times in a high-performance computing (HPC) environment varying initial conditions in order to: find outliers, examine the potential range of outcomes, and test the model across its parameter space. Figure 1 represents the Data Farming process as a set of loops: one to build and adjust the model, and another to examine the model results over the potential space of input variations.

Loops within data farming represent a collaborative process among subject matter experts, modelers, analysts, and decision-makers. These loops support an integrated process of verification and validation. While implementing the ABM the behaviors of agents and groups of agent may be tested in the HPC environment by varying input data over the potential valid input space. Statistical studies of all or portions of the model processes may be examined to determine the sensitivity of the agent behavior and model results across the parameter space.

## CROSS CHECK OF V&V METHODS

Both the military and ABM community see the necessity of some V&V for simulations, and both see this as difficult for models focusing on human behavior and decision-making. Furthermore, both communities see this as a very difficult endeavor. This section will briefly compare methods for V&V from each community.

Both DMSO and the ABM community see the utility of comparing models with each other. The problem is that for this methodology to have meaning in the military context the model(s) that are compared against the focus model must, themselves, be verified and validated. In the ABM case it may be difficult to find another validated ABM against which to compare. However, it is likely an ABM does not represent the first attempt to describe a system; the most obvious model to compare the ABM in question is the equation(s) that were used to describe the system originally. For example, if one is using an ABM to represent combat it is reasonable to ask if the ABM replicates the approved model for combat, most likely a model based on Lanchester equations (Sheldon 2002). This means that if one intends to validate by docking, one must choose the model(s) carefully. If the plan is to dock ABMs against unvalidated models such as other ABMS, the docking process may have minimal V&V benefit. It will, of course, help to demonstrate that the results one obtains are robust to some of the subtle assumptions inherent within an ABM creation environment.

One methodology espoused by DMSO is iterative V&V. This methodology may work quite well for physics-based simulations, but iterative development could be very problematic for ABMs. As agents are created via a code set that is then replicated for each agent when it is instantiated, iterative development may have many intermediary steps that produce nonsense when replicated across a population of agents. This means that validation on intermediate steps could be, at best, a waste of money and time. However, there is likely a useful middle ground. As an example consider the case of validating agent movement algorithms. It makes sense to validate the movement algorithms upon their completion. Validating intermediate movement algorithm would result in significant overhead with very little insight as incomplete movement algorithm may generate nonsense behaviors. Agent decision-making may be similar—incomplete decision-making rules may generate nonsense. However, decision-making and movement algorithms may be able to be V&V separately. It is this *segmented* rather than *iterative* approach that may have the most value when trying to limit developmental risk but also only V&V the model at logical points.

Axtell's levels of empirical relevance and DMSO's six domains of correspondence complement each other and can be used to create a lexicon to express the level or veracity of V&V that was achieved for a model. A model that corresponds to all of DMSO's domains will, necessarily, find itself in Axtell's level 3. Qualitative Sociological correspondence would place a model in level 1, while quantitative Sociological correspondence and qualitative Psychological, Physiological, Computational, and Physical correspondence should produce models of level 2. It may make sense to split this up and say that Axtell's level 0 requires at least Physical correspondence. For example, an ABM of shallow water obstacle clearing was created by incorporating physical data generated by experimentation. Data from the experiment was incorporated into the agent movement algorithms. This helped the argument for validation as empirical data existed to demonstrate that agent movement was correct (Paterson 2003).

Alfred Brandstein, one of the initial developers of the Data Farming paradigm, sees Data Farming as an important part of the validation of abstract models. The right portion of Figure 1 represents Brandstein's view of the output space of the Data Farming process and its utility in a validation process. Data Farming results fall into three categories: 1) output expected valid results; 2) output that is considered invalid; and 3) output that is unexpected but determined to be valid. One of the goals of Data Farming is to maximize expected valid output; minimize the invalid output; and gain insight from the unexpected, but valid output. The more success one has in this the more confidence one can have in the unexpected valid output and insights derived from it. This will increase the credibility of the simulation and increase the likelihood of obtaining accreditation for its use.

Data Farming also represents one method to achieve meaningful, efficient V&V of ABMs that include some aspect of human behavior representation and are not strictly physical. Within the Data Farming methodology one can view the simulation as a black box and "validate" it by comparing to measures against the real world…detailed understanding of the actual phenomena may not be necessary if the goal is to provide insight. Furthermore, given Data Farming use of HPC and intelligent experimental design (Lucas 2002, Sanchez 2005), one can efficiently sample vast parameter spaces to understand the functioning of the model and increase the likelihood of finding usable data for comparison.

## CONCLUSION

As with any cultural shift, we find ourselves in the midst of a painful transition. Judging the adequacy of ABMs by a uniform standard of V&V misses the point for which the models have been constructed. Simulations are tools to increase our understanding; some are precise, some are coarse. The appropriateness of a model or simulation is application dependent. As we have different modeling techniques for different problems, so must we have different V&V approaches. ABMs are used for different phenomena than closed form physical systems; therefore, the standards and approaches for V&V should necessarily be different.

The traditional view of V&V includes a detailed understanding of the inner workings of the simulation (verification) as well as predictability (validation). The concordance of the model with reality is largely dependent upon the depth of analytic understanding of the phenomena as well as the complexity of the phenomena being simulated. We contend that for many uses of ABMs we are not concerned with anything but the aggregate behavior of the simulation.

Military VV&A is built upon a foundation of physics-based models. As such, it has created a methodology suited to processing those types of models. As ABMs are frequently used for problems where there is no quantitative understanding of the inner workings and the physics involved, it is an untenable position to wait for this understanding prior to using the tool. DMSO has begun to acknowledge the growing importance of models that incorporate human behavior and agents in nontrivial ways. However, as in the ABM community, the military process to V&V these models has not yet been formalized. If the ABM community can formalize a process for V&V it is highly likely that it can have a great deal of influence in the eventual adoption of formal (and practical) methodologies within the military and the greater analytic community.

When creating such a formalized system it should be kept in mind, as Charles Macal pointed out (Macal 2005), the end result of model V&V is not a V&V'ed model but a model that has passed all of its V&V tests. In this way model validation becomes, in essence, a series of attempts to invalidate the model. Verification and Validation are methods for removing reasons not to use a model (Macal 2005). Furthermore, if one takes the position that ABMs are not used to predict but rather used to provide insight, it is unreasonable and inappropriate to pursue traditional military V&V techniques.

Appropriate V&V techniques should be applied with as much rigor as is feasible. Releasing the shackles of traditional V&V does not open the door for any ABM or other model to be used indiscriminately. In this light, specific simulations require individual approaches to V&V depending upon the intended use and the depth of phenomenological understanding. Employment of SMEs, Data Farming, and Docking to better understand the behavior of the simulation will be necessary. Most of all, the V&V process should be malleable to the ABM's intended use and be extremely cognizant of the limitations of its use and its extensibility.

Such a formalized process should include at a minimum:

1. A methodology to check the functioning of the agent code—was the model built correctly?

2. A methodology to systematically try to "break" the model and intelligently explore the parameter space associated with it. ANTs is offered as an option as is Data Farming.

3. Assuming the model passes step 2, it is suggested to employ a methodology to dock the model. Docking against at least one validated or widely accepted model is preferred. Docking against other validatd models aids in bringing to light hidden assumptions that may be lurking in the code.

## REFERENCES

Axtell, R., R. Axelrod, J. Epstein, M. Cohen, 1996. "Aligning simulation models: A case study and results, Computational & Mathematical Organization Theory," **1(**2), Feb.

Axtell, R., 2005. "Three Distinct Kinds of Empirically-Relevant Agent-Based Models," *Brookings Institute, 30 September.*

Brandstein, A. and Horne, G. 1998. Data Farming: A Meta-Technique for Research in the 21st Century. *Maneuver Warfare Science 1998*, G. Horne and M. Leonardi, editors. Marine Corps Combat Development Command Publication, Quantico, Virginia.

Fagiolo, G., 2006. "Empirical Validation of Agent Based Models: A Critical Survey," LEM

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

Horne, G. and Meyer, T. 2004. "Data Farming: Discovering Surprise." *Proceedings of the 2004 Winter Simulation Conference*.

Ilachinski, A., 2004. Artificial War: Multiagent-Based Simulation of Combat. World Scientific, Singapore.

Koehler, M, P. Barry, B. Widdowson, A. Forsyth., 2004.l "Case Study: Using Agents to Model Stability and Support Operations." *Proceedings of Agent 2004*. Argonne National Lab, Chicago, IL.

Lucas, T., Sanchez, S., Brown, L., and Vinyard, W. 2002. Better Designs for High-Dimensional Explorations of Distillations. *Maneuver Warfare Science 2002*. G. Horne and S. Johnson, editors. United States Marine Corps Project Albert. Quantico, Virginia.

Macal, C., 2005. Model Verification and Validation. Presentation to the Workshop on "Threat Anticipation: Social Science Methods and Models." The University of Chicago and Argonne National Laboratory. Chicago, IL.

Meyer, T., Koehler, M., Barry, P., and Tivnan, R. 2005. "How Simple is Simple Enough?: Military Modeling Case Studies," *Proceedings of Agent 2005*. Argonne National Lab, Chicago, IL.

Miller, J., 1996. "Active Nonlinear Tests (ANTs) of Complex Simulation Models," CMU working paper, 5 February 1996. http://zia.hss.cmu.edu/econ/misc/wp.html.

Paterson, Capt R., and E. Bitinas, 2003. Modeling Obstical Reduction with the Pythagoras Agent-based Distillation. *Maneuver Warfare Science 2003, G. Horne and S. Johnson, editors*. United States Marine Corps Project Albert. Quantico, Virginia.

Sanchez, S. M. and P. J. Sanchez, 2005. "Very large fractional factorials and central composite designs," ACM Transactions on Modeling and Computer Simulation **15**(4): 362-377.

Seldon, B, 2002. Comparing the Results of a Nonlinear Agent-Based Distillation to Lanchester's Linear Law. *Maneuver Warfare Science 2002*. United States Marine Corps Project Albert. Quantico, Virginia.

[i] http://www.airforce-technology.com/contractors/training/link/

[ii] http://www.peostri.army.mil/PRODUCTS/JTC/

[iii] http://www.dodgamecommunity.com/modules.php?op=modload&name=News&file=article&sid=81

# SIMULATING TERRORISM IN A COMMUNITY

M. W. DREWEK,* Michigan Tech, Houghton, MI
W. M. BULLEIT, Michigan Tech, Houghton, MI

## ABSTRACT

Prediction of where terrorists are most likely to strike is an issue that concerns planners, law enforcement and government agencies at various levels, and engineers who must design facilities of all kinds. The ultimate goal of our research is to be able to estimate the probability of attack for various types of facilities in a population center so that resources can be allocated for hardening or otherwise protecting those facilities; the hypothesis is that agent-based modeling could provide the basis for such prediction. This paper describes a two-phase, resource-based, agent model, with endogenous terrorism and security; examines the validity of the simulated terrorist activities; and provides a social psychological basis for the generation and behavior of the terrorist agent. Using incubation and week-to-week community simulation, an experiment was conducted using a four-resource, quasi-realistic community environment. Results lead to strong evidence suggesting that terrorist attack magnitudes are lognormally distributed; time between attacks are exponentially distributed; and the attack magnitude is scale invariant and independent of the time between attacks.

**Keywords:** Agent-based modeling, artificial societies, simulation, terrorism

## INTRODUCTION

Prediction of where terrorists are most likely to strike concerns planners, law enforcement and government agencies at various levels, and engineers who must design facilities of all kinds. Agent modeling of civil violence has been performed in the past (Epstein 2002). The present work is an effort to use *simulation-based engineering science* techniques (NSF 2006), specifically agent-based modeling, to examine the interaction of civilians, terrorists, and security to determine the types of facilities that are most susceptible to attack. The ultimate goal of our research is to be able to estimate the attack probability for various types of facilities in a community so resources can be allocated for hardening or otherwise protecting those facilities.

A two-phase, resource-based, agent model, an extension of the type used by Epstein and Axtell (1996), has been developed to simulate community terrorist activity. The environment where this community evolves consists of a rectangular grid on which lie a number of renewable resources. Civilian agents collect, store, and expend resources, moving about and interacting on this community environment. Endogenous terrorist agents are formed from within the civilian agent population using tag-mediated cultural interaction, similar to that used by Axelrod (1997). The terrorist agents conduct surveillance and commit terrorist attacks. When a terrorist attack occurs, fear is generated in the area subjected to the attack; over time this fear spreads out to surrounding areas. Searching for resources, civilian agents attempt to balance the rewards of

* *Corresponding author address:* Matthew W. Drewek, Department of Civil and Environ. Engrg., Michigan Tech, 1400 Townsend Dr., Houghton, MI 49931-1295; email: mwdrewek@mtu.edu.

visiting the nodes with their fear of those nodes and their innate nervousness. Security agents search for and arrest terrorist agents in areas of locally high population and are introduced onto or removed from the environment according to civilian demand.

Results from an earlier version of the model show that the location of attacks is affected by the level of security; higher levels of security shift many of the attacks away from the areas of high resources (Bulleit and Drewek 2005a). Further work (Bulleit and Drewek 2005b), led to an extension of a technique employed by Cederman (2003a), where the simulation process was divided into two parts: (1) the *incubator*, in which the community evolved to a certain point, and (2) the *simulator*, where the day-to-day community simulation was performed. In this work, an extension of Bulleit and Drewek (2005b), incubation and week-to-week simulation are used in conjunction with a quasi-realistic community environment. The objective of this paper is to use the model to quantitatively investigate terrorist activity within a community and provide a social psychological basis for a terrorist agent's behavior.

## MODEL DESCRIPTION

Figure 1 shows the environment, representing a community, town, or city, and consists of a "walled" 50 by 50 rectangular grid on which lie a number of *piles* of resources. For this study, the environment consists of nine resource piles, representing four different renewable resources, each representing general needs in a community. Resource 1 represents locales in the community where people work, and resource 2, where people live. Resources 3 and 4 represent two types of commercial districts (supplying commerce, entertainment, etc.) in the community.

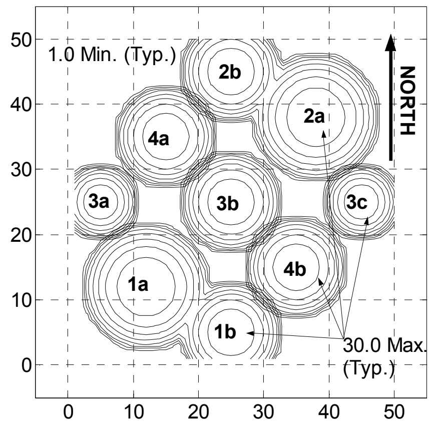


**Figure 1**. Community Environment

Revised resource distributions required re-calibration of the agent resource metabolisms from those in Bulleit and Drewek (2005b). Each agent is assigned an initial metabolism for each resource on the environment from a uniform distribution with range 1.5 to 4.5 (U(1.5, 4.5)) for resources 1 and 2 and from U(1.25, 2.75) for resources 3 and 4. The resource metabolisms are proportional to the amount of time the average person in the community spends pursuing what the resources represent. For example, considering resource 1, the average person in this community spends 8 hours per day working; likewise, for resource 2, the average person spends 8 hours at home. The activities and needs satisfied by resources 3 and 4 will account for the remaining 8 hours per day. Therefore, subdividing any particular period of time, resources 1, 2, 3, and 4 will account for 1/3, 1/3, 1/6, and 1/6 of that time period, respectively.

Previously, civilian and terrorist agents could only alter their agent type on time steps when cultural interactions occurred; it was determined that this led to excessive terrorist activity, since moderate agents were becoming and remaining terrorist agents for time durations long enough to conduct attacks. Therefore, the model was altered to determine agent type at each time step. For similar reasons, the surveillance data for terrorist agents reverting back to civilian status is zeroed out. These changes reduced simulated terrorist activity and increased sensitivity to cultural interaction parameters, i.e., the base probability for radical change, $P_b$, and the probability for isolated change, $P_{ic}$ (Bulleit and Drewek 2005b). For this work, both $P_b$ and $P_{ic}$ are equal to 0.03, calibrated to ensure some level of terrorist activity within the community represented by the agent population and quasi-realistic environment.

## TERRORIST AGENTS: SOCIAL PSYCHOLOGICAL BASIS

Terrorist agents evolve from the civilian agent population using a tag mediated process based on the approach used by Axelrod (1997). Each agent is assigned a tag at the beginning of the simulation (randomly) or at birth (inherited from parents). The tag consists of a string of five integers ranging from 0-9. As the agents move about their environment they interact with other agents. Using the approach in Axelrod (1997) alone, conversion through interaction, the agent tags will quickly become homogeneous. Radical and isolated changes were introduced (Bulleit and Drewek 2005b), providing variability and preventing tag convergence. Conversion and radical change, requiring agent-to-agent interaction, is an inter-personal process; isolated change is an intra-personal process (Markus and Wurf 1987). When an agent becomes isolated, has a cultural tag dissimilar to neighboring agents, the agent will have little chance of interaction, which will indirectly provide the agent with a cognition of its environment, analogous to attribution theory in social psychology (Harvey and Weary 1984).

The integers making up the cultural tag do not represent specific cultural traits; they represent generic propensities towards radical behavior. Each tag integer represents a *feature*. Axelrod (1997) used the value of the tag integer categorically; here the tag integer represents a measure of extremism, leading to ties with current terrorism social psychology research (Arena and Arrigo 2005). For example, for a particular agent, if a tag integer is 4 or 5, then moderate tendencies are inherent. If a tag integer is either 0 or 9, then extremist tendencies are inherent. The individual features are analogous to "the five organizing concepts from structural symbolic interactionism": "symbols, the definition of the situation, roles, socialization and role-taking, and emergence of the self" (Arena and Arrigo 2005, p. 488). Combining the tag mediated cultural interaction with the meaning of the individual features, provides an "understanding of how terrorist identities are created, embraced, and maintained, as well as how they influence the behavior of members of the extremist subculture" (Arena and Arrigo 2005, p. 485).

An agent becomes a terrorist based its *cultural identity*, the sum of the five integers in its tag. The cultural identity represents a measure of extremism. Agents are more likely to become terrorists if their individual tag integers indicate extremism and collectively have focused tendencies (integers consistently falling either left or right of moderate). The probability that the agent becomes a terrorist is determined using a U-shaped symmetrical polynomial function, see Figure 2, that passes through 1.0 at a sum of 0, 0.0 at a sum of 22.5, and through 1.0 again at 45. Thus, there is some probability that any agent can become a terrorist, but the probability is greatest near the end points of the sum of the tag integers. Relating back to structural symbolic interactionism: "roles can be understood as the means by which people are connected to the social systems . . . roles provide an underlying framework where interaction can occur and where shared meaning within this framework can exist" (Arena and Arrigo 2005, p. 493). Hence an agent's cultural identity represents "goals (desired states of the world), standards (ideas about how people should act), and preferences (likes and dislikes)" (Johns and Silverman 2001), specifically, as related to terrorist-type behavior.

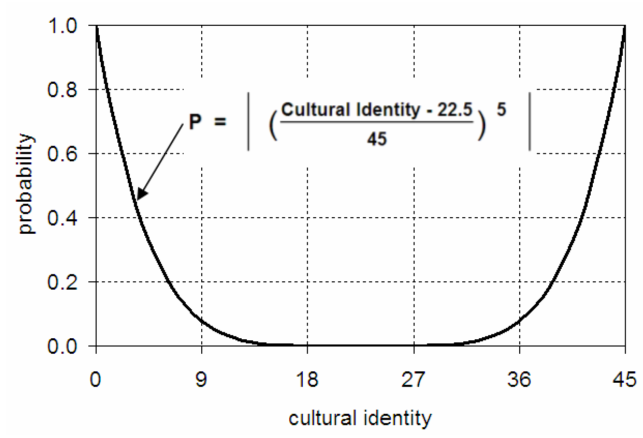


**Figure 2**. Probability of Becoming a Terrorist

After a civilian agent has searched for and collected resources and satisfied metabolisms during a time step, whether it has procreated and culturally interacted or not, its cultural identity is used to determine the probability of becoming a terrorist (if already a terrorist, the probability of remaining one). This probability is then used to determine the agent's type (whether civilian or terrorist) for the next time step; hence, the decision rules (Johns and Silverman 2001) that define agent behavior are determined. After an agent becomes a terrorist, it remains an *inactive* terrorist until its age and wealth each reach specific values, only then becoming *active*, and remaining active so long as its wealth remains above a specified value (Bulleit and Drewek 2005b). An agent's movement between inactive and active terrorist behavior, and vice versa, manifests self preservation behavior, the agent wanting to survive long enough to accomplish its goal, the future terrorist attack (Johns and Silverman 2001).

An active terrorist agent stops looking for resources and begins to examine the wealth on the von Neumann neighborhood of nodes within its vision and moves to the node with the largest surrounding wealth. Even though the terrorist stops looking for resources, it continues to collect resources at the nodes it is visiting. The *target data* consists of the wealth of the agents present at the time of the search plus the moving average of the agent wealth over the recent past (referred to as *historical nodal wealth*) on each node in the von Neumann neighborhood. The terrorist agent thus equally weighs pure destructiveness of current wealth (Conrad et al. 2004, p. 27) with symbolic destructiveness (Arena and Arrigo 2005); terrorists do not desire to strike just high wealth locations, but also locations where wealth passes through.

The active terrorist agent keeps track of the mean and standard deviation of the largest target data values seen in its travels since becoming a terrorist, collectively referred to as *surveillance data*. When it finds a node that has target data greater than the mean of the surveillance data plus some number of standard deviations *and* the coefficient of variation of its surveillance data is less than a specified value *and* assuming that the terrorist agent has collected target data from at least five sites, it becomes a suicide bomber and explodes, destroying wealth on the Moore neighborhood. The nodes attacked that are not in the target data represent the terrorist agent's incomplete information about the current situation and collateral damage. The terrorist agent is focused purely on destroying wealth; Joseph Conrad's fictional *The Secret Agent*, which became popular in both the government and media in the immediate post-9/11 era because of its "uncanny insights into the individuals and their society," supports this simplifying assertion (Conrad et al. 2004, pp. xiii, 27). The local information available to the terrorist agent, though incomplete, ensures that an attack will occur in an area of relatively consistent high wealth. Thus, terrorist agents exhibit satisficing behavior (Simon 1957).

## RESULTS AND DISCUSSION

The process described above was implemented using MatLab (MathWorks 2002). With parameters described above and in Bulleit and Drewek (2005b), the 1,200-year incubation period (where one time step equals one year) was initialized with 500 agents. Through incubation an agent population acclimated to living on the environment was generated. After 1,200 years, the post-incubation conditions were recorded. Four post-incubation conditions were generated, with 693, 632, 537, and 664 non-security agent populations. Each post-incubation condition was then used as the initial conditions for six 40-year community simulations (where one year equals 52 time steps). Figure 3 shows the non-security agent population for the 24 simulations.

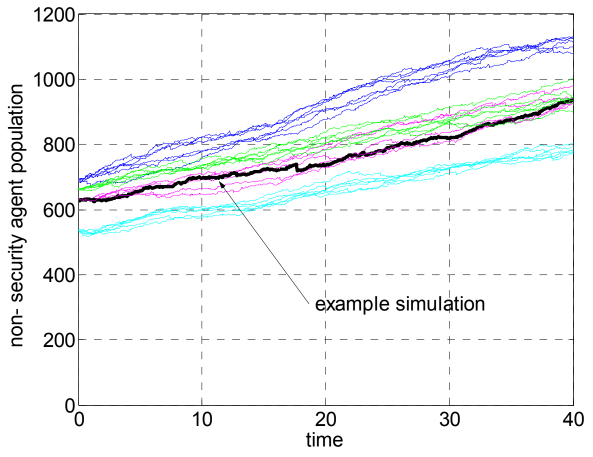


**Figure 3**. Simulation Population Histories

One simulation was selected as an example for further discussion. The population history for this example, beginning with 632 non-security agents and ending with 935, is highlighted in Figure 3. Figure 4 shows the terrorist activity (attack magnitude), as well as the security level (security population as a percentage of the non-security population) time history. Five terrorist attacks occurred during the 40-year simulation: attacks occurred at times (in years) 1.269, 12.808, 17.654, 17.712, and 18.769; attack damage magnitudes, in terms of the total agent generalized wealth (Bulleit and Drewek 2005b) destroyed in each attack, were 111.0, 176.1, 185.5, 192.0, and 156.5, respectively. The security level was less than 1% entering the simulation. When the first attack occurred on 3b (Figure 1), the security level increased briefly to more than 2.5%, but declined to the pre-attack level. The second attack had little effect, because of the 10+ year period between attacks and the 1b location. Before this attack fear had dispersed across the environment. When the attack occurred, newly generated fear was high in magnitude but concentrated in an area with relatively low population density, and therefore, relatively few agents were affected. The third and fourth attacks occurred close together, were both relatively large, resulted in security increasing to almost 7.5%, and caused a noticeable population drop (Figure 3). The third attack occurred near the southern edge of 3b; the fourth attack occurred on 2a, just opposite 3b. The final attack occurred on 3b, little more than one year later, and caused the security level to exceed 7.5%, before dropping to a stable level near 4%.

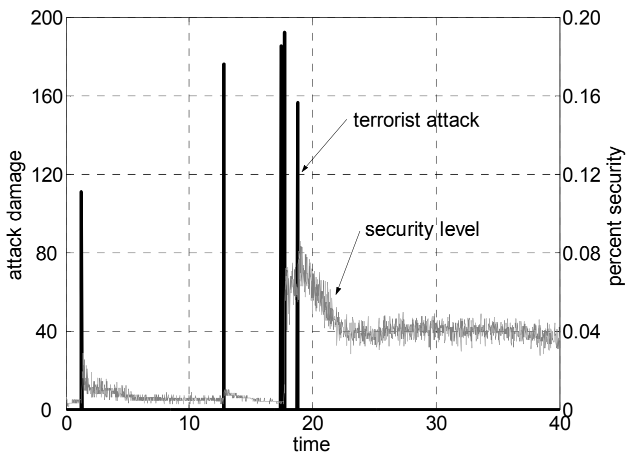


**Figure 4**. Example Terrorism and Security Behavior

The transition between incubation and simulation is not seamless. Agent demographics, as well as the security level and behavior, remain stable between incubation and community simulation, but in the beginning of each community simulation, the behavior of agents seeking resources change for a short time. To maintain stable generalized wealth, environmental resource concentrations are factored by 1/52 when beginning the community simulation, causing the agents to try to *re-equalize* their generalized resources (Bulleit and Drewek 2005b). The resulting change in behavior is noticeable but temporary; in general, stabilization occurred within two years. During the twenty-four 40-year simulations, 153 terrorist attacks occurred, for an occurrence rate of 0.1594 attacks per year. Twenty-one attacks occurred during the first two years of the simulations, for an occurrence rate of 0.4375 attacks per year. Such a significant change in the occurrence rate further corroborates the need for a stabilization period. Thus, the attacks occurring during the first two years will not be used further. Considering only the last 38 years of each simulation, a total of 132 attacks occurred, for a 0.1447 occurrence rate.

Figure 5 shows the areas subjected to terrorist attacks. Comparing the community environment (Figure 1) with the cumulative number of terrorist attacks (Figure 5) indicates a definite pattern for attack location. The region most affected by terrorism is 3b. Areas immediately north and south of 3b ('N' and 'S' travel routes) were also significantly impacted. Nodes on the edges of 1a, 1b, 2a, 2b, 4a, and 4b were repeatedly hit; regions in the interior of these resource concentrations were hit in isolated attacks, as were the 'E' and 'W' travel routes. Areas 3a, 3b, and the hinterlands suffered no attacks.

To determine if there is a relationship between occurrence time and attack magnitude, a line was fitted through the data. The estimate for the slope of this fitted line, $b_1$, was -0.10. A two-sided t-test was performed to determine whether or not the actual slope equals zero (Ayyub and McCuen 2003). Using the mean square error to estimate the standard deviation of $b_1$, it was found that the calculated absolute test statistic equaled 0.50. Therefore, with 95% confidence (with the critical value equaling 1.978), the actual slope could not be shown to be other than zero; i.e., attack damage could not be shown to vary over time. This conclusion is supported by

Clauset and Young (2005), where using actual data on terrorist attacks occurring from 1968-2004, global terrorism appeared to be scale invariant.

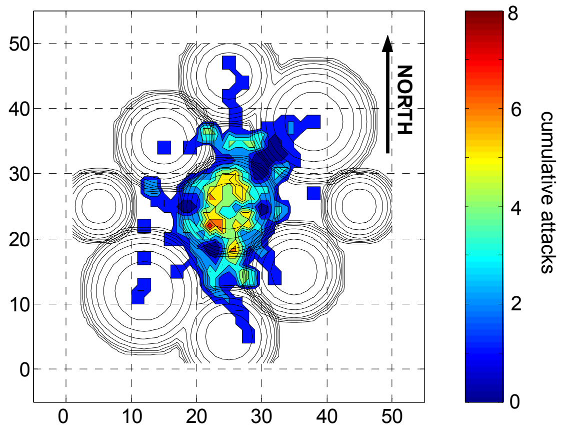


**Figure 5**. Cumulative Terrorist Attacks

To examine the relationship between the time between attacks and attack magnitude, the time between attacks were adjusted for the 2-year stabilization period. For the example simulation, the attack occurring at time 1.269 is eliminated and the other inter-arrival times are 10.808, 4.846, 0.058, and 1.057 years, in order of occurrence. A line was fit through the adjusted data for all the inter-arrival times; the estimate for the slope of this fitted line, $b_1$, was 0.08. A two-sided t-test was performed to determine whether or not the actual slope equals zero (Ayyub and McCuen 2003). Using the mean square error to estimate the standard deviation of $b_1$, the absolute test statistic was found to equal 0.19. Therefore, with 95% confidence, the actual slope could not be shown to be other than zero; attack damage could not be shown to vary with the time between attacks.

The histogram of the time between attacks strongly indicated an exponential distribution. Consequently, the Chi-Squared Goodness-of-Fit Test (Ayyub and McCuen 2003) was conducted for an exponential distribution with an occurrence rate, $\lambda$, of 0.1447 attacks/year (the maximum likelihood estimate calculated using 132 terrorist attacks occurring during twenty-four 38-year simulations). The null hypothesis was that the time between attacks is exponentially distributed with $\lambda = 0.1447$ attacks/year. The alternate hypothesis was that the time between attacks cannot be represented as such. The Chi-Squared test statistic was calculated dividing the observed data into 12 categories; the observed counts in each of the 12 categories were greater than 5, thereby maintaining the effectiveness of the test (Ayyub and McCuen 2003). The test statistic was calculated as 8.32; with 10 degrees-of-freedom the critical values for the Chi-Squared statistic with 50% and 70% significance levels are 9.34 and 7.27, respectively. Hence, with at least a 50% significance level, the time between attacks cannot be shown to come from other than an exponential distribution with $\lambda = 0.1447$. The terrorist activity modeled appears to be a Poisson process (Ayyub and McCuen 2003).

A histogram of the attack damage magnitudes strongly indicated a lognormal distribution. The sample attack magnitude mean and standard deviation were 150.2 and 27.0, respectively. An analysis was conducted using probability plotting; the resulting coefficient of determination was 0.9898 and the Kolmogorov-Smirnov statistic was relatively small, 0.0485. Hence, the histogram and results from probability plotting suggest attack magnitudes come from a lognormal distribution.

Research conducted by Cederman (2003a), using an agent-based model of war and state formation, concluded that the magnitude of wars was power law distributed. Later, Clauset and Young (2005), using a data set of global terrorist attacks occurring from 1968-2004, indicated that the magnitude of terrorist attacks was power law distributed. However, work done by Cederman (2003b) concerning state size, and the present work on the magnitude of terrorist attacks, indicates lognormal distributions. Cederman (2003b) addressed this supposed discrepancy by highlighting the inherent limitations governing state size which prevent the formation of the power law distribution's fat tails. The range in terrorist attack magnitude occurring in our community is limited by the fixed-resource environment, terrorist agent's triggering mechanism, single-type terrorist attack, and non-cooperating terrorist agents. Therefore, the lognormal distribution of our community's attack magnitude is not unexpected.

## CONCLUSIONS

The following conclusions can be drawn from the model. The level of security present on the environment is sensitive to the time between attacks, magnitude, and location. The simulations required a stabilization period, after which data was collected. The attack magnitude was scale invariant and independent of the time between attacks. Strong evidence suggests that the time between attacks is exponentially distributed, and the attack magnitude is lognormally distributed. The model accounts for not only the agents' relative need for the various resources, but also for the spatial distribution of those resources. Attacks occur not only where resource concentrations are highest, but also in areas through which agents travel. The two-phase agent model, with endogenous terrorism and security, on a quasi-realistic environment, has been partially validated using two methods: (1) the terrorist agent model is supported by social psychology and terrorism theories and (2) the aggregate terrorist activity resulting from numerous community simulations favorably compares to conclusions drawn from other models and examination of real-world data.

## ACKNOWLEDGMENTS

We would like to thank the National Science Foundation Graduate Research Fellowship Program and the Michigan Technological University Department of Civil and Environmental Engineering whose support has made this project possible. Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the writers and do not necessarily reflect the view of the National Science Foundation.

## REFERENCES

Arena, M. P. and B. A. Arrigo, 2005, "Social Psychology, Terrorism, and Identity: A Preliminary Re-examination of Theory, Culture, Self, and Society," *Behav. Sci. Law*, 23:485-506.

Axelrod, R., 1997, *The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration*, Princeton, NJ: Princeton University Press.

Ayyub, B. M. and R. H. McCuen, 2003, *Probability, Statistics, and Reliability for Engineers and Scientists*, 2nd Edition, New York, NY: Chapman & Hall / CRC.

Bulleit, W. M. and M.W. Drewek, 2005a, "An Agent-Based Model of Terrorist Activity," *Proceedings of the North American Association for Computational Social and Organizational Science (NAACSOS) Conference*, Notre Dame, IN, June 26-28.

Bulleit, W. M. and M. W. Drewek, 2005b, "Simulating Initial Conditions in Agent-Based Modeling," *Proceedings of the Agent2005 Conference*, Chicago, IL, October 13-15; available at http://www.agent2005.anl.gov/proc2005.html

Cederman, L.-E., 2003a, "Modeling the Size of Wars: From Billiard Balls to Sandpiles," *American Political Science Review* 97(1):135-150.

Cederman, L.-E., 2003b, "Explaining State Sizes: A Geopolitical Model," *Proceedings of the Agent2003 Conference*, Chicago, IL, p. 388.

Clauset, A. and M. Young, 2005, "Scale Invariance in Global Terrorism," available at http://arxiv.org/abs/physics/0502014.

Conrad, J., P. L. Mallios, and R. D. Kaplan, 2004, *The Secret Agent: A Simple Tale*, 2004 Paperback Ed., NY: Modern Library (First Edition, 1907).

Epstein, J. M., 2002, "Modeling Civil Violence: An Agent-Based Computational Approach," in *Proceedings of the National Academy of Sciences*, Vol. 99, Suppl. 3.

Epstein, J. M. and R. Axtell, 1996, *Growing Artificial Societies: Social Science from the Bottom Up*, Washington, DC: The Brookings Institution.

Harvey, J. H. and G. Weary, 1984, "Current Issues in Attribution Theory and Research," *Ann. Rev. Psychol.*, 1984, 35:427-459.

Johns, M. and B. G. Silverman, 2001, "How Emotion and Personality Effect the Utility of Alternative Decisions: A Terrorist Target Selection Case Study," *10th Conference On Computer Generated Forces and Behavioral Representation*, SISO, May 2001.

Markus, H. and E. Wurf, 1987, "The Dynamic Self-Concept: A Social Psychological Perspective," *Ann. Rev. Psychol.*, 1987, 38:299-337.

The MathWorks, 2002, *MatLab Student Version with Simulink*, Version 6.5, R13, Natick, MA.

NSF, 2006, *A Report of the National Science Foundation Blue Ribbon Panel on Simulation-Based Engineering Science*, available at www.nsf.gov/pubs/reports/sbes_final_report.pdf

Simon, H. A., 1957, *Administrative Behavior*, 2nd Ed., New York, NY: Free Press.

# Transportation and Mobility

# USING REPAST TO DEVELOP A PROTOTYPE AGENT-BASED PEDESTRIAN EVACUATION MODEL

C.J.E. Castle, Centre for Advanced Spatial Analysis, University College London, UK*

## ABSTRACT

London's King's Cross St. Pancras underground station has been the unfortunate location of two major incidents within the last twenty years. A fire in November 1987 and the terrorist bombing in July 2006 both resulted in the loss of lives, and the injury of many people. The implementation of measures to mitigate or neutralise the effect of all possible future incidents at this site is unrealistic. The adoption of preparedness measures is crucial for the emergency services to limit the loss of life and property, and to improve the response phase of an incident. King's Cross underground station is currently being redeveloped, partly to mitigate the remaining few operational and safety issues identified after the 1987 fire, and also to allow for a predicted increase in passenger use. Despite these modifications and improvements, both the surrounding built environment and the station will necessarily remain complex structures. The local emergency services have several duties placed upon themselves in the event of a major incident at this site, and a computer based model capable of examining the effects of different incident assumptions or contingencies has been identified as a potential benefit to the local National Health Service resilience planning department.

The specific aim of this paper is to provide the reader with an overview of this research. To begin, the aims and deliverables are identified. In light of these, principles of pedestrian evacuation modelling are presented, highlighting a shift in approaches: from aggregate movement, to individual-level movement and behavioural models. The feasibility of using a proprietary pedestrian evacuation model to achieve the research goal is discussed. This is followed by an agenda for developing an agent-based pedestrian evacuation model using the Repast toolkit. This paper concludes with progress of the prototype model to date.

**Keywords:** Pedestrian evacuation modelling, Agent-Based Modelling, Repast toolkit, London's King's Cross St. Pancras underground station.

## INTRODUCTION

London's King's Cross St. Pancras underground station has been the unfortunate location of two major incidents within the last twenty years. On November 18th 1987, a fire within the station resulted in 31 fatalities, more than 60 severe injuries, and serious structural damage to the station (Hallén and Kulling, 1990). On Thursday July 7th 2005 one of four terrorist bombs to explode on London's public transport network occurred on a train leaving King's Cross St. Pancras underground station (BBC News, 2005). Twenty six people lost their lives and many more people were injured in this incident.

* Corresponding author address: Christian J.E. Castle, Centre for Advanced Spatial Analysis, University College London Department of Geography, Gower Street, London, WC1E 6BT, England, UK; Tel. +44(0)207 679 0527; Fax +44(0)207 679 0565; E-mail: c.castle@ucl.ac.uk; Website: http://www.casa.ucl.ac.uk/cjec/phd.

During a major incident such as the King’s Cross fire or the London bombings, people within an enclosure, the enclosure itself, and the immediate environment (i.e., at the street level) are affected in ways that considerably reduce the speed of incident response (Kwan and Lee, 2005). The complex internal structure of buildings and limited number of access points at street level render speedy escape and rescue particularly difficult. During any major incident a matter of minutes may lead to significant changes in the environment within which evacuees need to escape and rescue personnel have to operate. In this respect, the efficiency and effectiveness of the emergency services is critical, and directly related to their major incident management.

Preparedness is a key component of major incident management, which can play a significant factor in the event that emergency response efforts become necessary (Castle and Longley, 2005). Unfortunately, information from comparable incidents is usually unavailable and the way that some incident scenarios evolve is unknown. Nakanishi *et al.* (2003) state that robust incident preparedness plans should incorporate the results of available computer simulation models. These models are especially useful in estimating the impacts of incident characteristics and response strategies, on response and evacuation times. The principles advocated by Nakanishi *et al.* (2003) provide the wider context in which this thesis is set.

The remainder of this paper provides the reader with further information regarding this research. The following section identifies the specific aims and objectives, highlighting the impact of the King’s Cross redevelopment on pedestrian egress from the underground station. Consequently, an overview of principles relating to the modelling of pedestrian evacuation is presented, identifying a shift in modelling endeavours. The penultimate section of this paper discusses the feasibility of using a proprietary pedestrian evacuation model to fulfil the research goal. This is followed by an agenda for developing an agent-based pedestrian evacuation model using the Repast toolkit. This paper concludes with progress of the prototype model to date.

## RESEARCH AIMS AND DELIVERABLES

This research is a direct response to the Department of Health’s (DoH, 2004a; b) request for the National Health Service to assess, and where necessary, make adjustments to their plans for coping with their responsibilities during a major incident. In particular, resilience planners within the local NHS department (Camden Primary Care Trust, PCT – the research sponsors) have identified the need to evaluate resources after the completion of the King’s Cross redevelopment.

An estimated £4 billion of investment has been allocated to redevelop the King’s Cross area of London, which is set to become Europe’s largest integrated transport hub. The redevelopment started in 2000 as is due to be completed in 2015[†]. One of the four main development projects entails the upgrade of King’s Cross St. Pancras underground station, one of London’s oldest and busiest stations. In part, the upgrade will address several of the remaining operational and safety recommendations identified by the Fennel investigation into the 1987 King’s Cross fire (Fennell, 1988). In addition, the station must cope with future passenger demand. The station is part of a major interchange between two national rail stations (St. Pancras and King’s Cross, Figure 1), which contributes to heavy passenger congestion during peak periods. At present an estimated 65,000 passengers traverse the station during the morning peak (7:00-10:00am). By 2011, with domestic and international high-speed rail services operating from St Pancras, 92,000 passengers will pass through the

[†] King’s Cross underground station is due to be completed in 2009.

underground station during the morning peak alone, rising to 105,000 during the Olympics in 2012 (London Underground Limited, LUL, 2006).

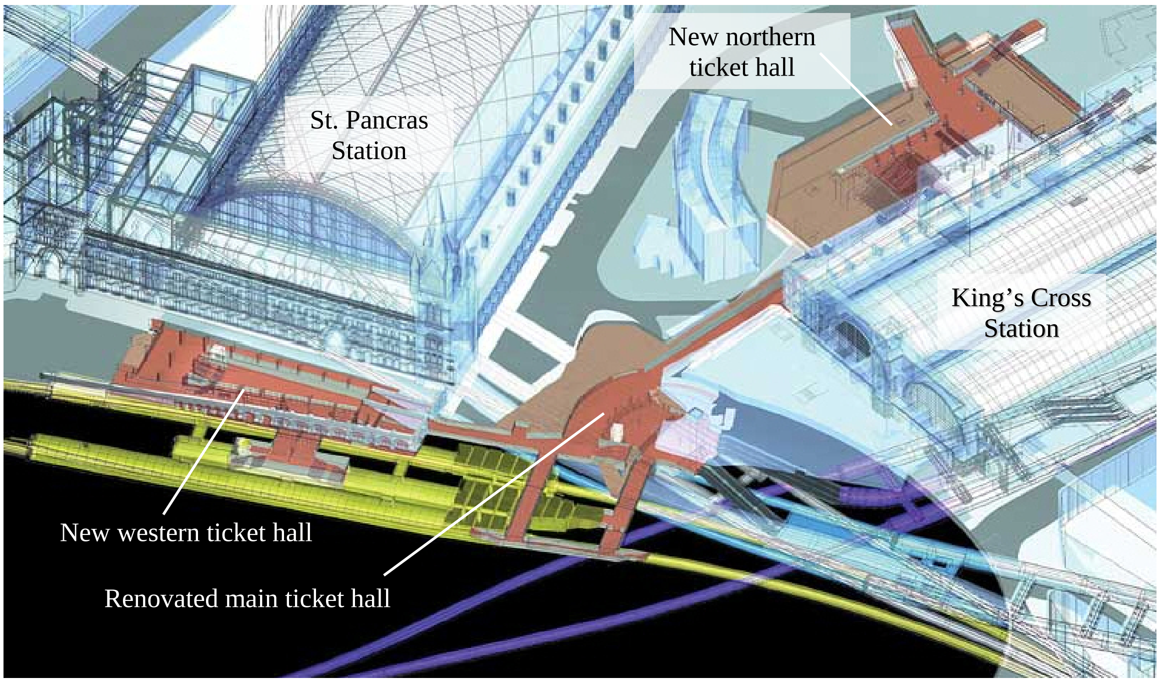


**FIGURE 1:** Renovated King's Cross St. Pancras underground main ticket hall, and new northern and western ticket halls (Allies and Morrison, 2004)

In light of the significant layout changes and increased passenger numbers, a main aim of this research is to develop a prototype pedestrian evacuation model for Camden PCT to evaluate their resources in the event of an incident within King's Cross St. Pancras underground station. In particular, Camden PCT are interested in the appraisal of pedestrian egress from the study area to determine the allocation and positioning of key emergency functions and facilities (e.g., ambulance loading point(s), casualty clearing station(s) to which the injured can be taken, etc), in the event of a future incident. Local emergency service administrators, medical and public health professionals, and other community policy makers and planners who must prepare for future incidents at King's Cross St. Pancras underground station have also been actively been involved in determining deliverables through the research advisory panel (see http://www.casa.ucl.ac.uk/kxsdsses/advisory.htm).

## PEDESTRIA EVACUATION MODELLING

The evacuation of occupants from a building during an emergency can be evaluated in several ways. Firstly, a building can be assessed by a full-scale evacuation demonstration, involving an exercise using a representative target population within the structure. However, this approach can pose ethical, practical, and financial limitations, which undermine its viability. Alternatively, before the construction of a building has started, an architect can design the layout based on prescriptive building codes; designed to accept or reject a proposed design on the basis of its compliance. Gwynne and Galea (1997) note, that in order to fully assess the potential evacuation efficiency of an enclosure, it is essential to address the configurational, environmental, behavioural, and procedural aspects of the evacuation process.

Configurational considerations are those generally covered by traditional building codes and involve building layout, number of exits, exit width, etc. In the event of an incident, environmental aspects need to be considered. These include, potential effects on building occupants as a result of heat, toxic and irritant gases, travel speed due to smoke density, and way-finding capabilities. Procedural aspects relate to staff actions, level of occupant evacuation training, occupant prior knowledge of the enclosure, emergency signage, etc. Finally, and what Gwynne and Galea (1997) perceive as the most important consideration, is the likely behavioural responses of occupants (e.g., initial response to evacuation, travel speeds, family/group interactions, etc.).

The aforementioned traditional methods of building design and evaluation fail to address these issues in a quantitative manner, preferring to rely almost totally on judgement and prescriptive rules. Prescriptive rules tend to have a dependency on configurational considerations (e.g., travel-distance and exit width), which can prove too restrictive. Moreover, these traditional methods are insensitive to human behaviour or varying emergency scenarios. Computer based models offer the potential to evaluate the evacuation of occupants from a building during an emergency; overcoming these limitations.

## Modelling Human Movement and Behaviour

In the building industry, research into quantifying and modelling human movement and behaviour has been underway for at least 30 years. This work has progressed down two routes: movement of people under emergency and non-emergency conditions. Early work was concerned with movement of people under non-emergency conditions (e.g., Fruin, 1971, and Predtechenskii and Milinskii, 1978). Their research mainly focused on the movement capabilities of people in crowded areas and stairs, and led to the development of initial pedestrian movement models such as PEDROUTE/PAXPORT. Attempts to simulate emergency pedestrian movement are somewhat more recent. One of the first publications was produced by Stahl (1978), which focused on modelling emergency egress during fires. Similarly, initial models only considered movement, although more recent models incorporate behaviour as well. Galea (2004) has subsequently classified the development of evacuation models into five generations: 1) manual calculations; 2) computer based flow or hydraulic models; 3) ball bearing models; 4) rule based models; and, 5) models sensitive to local conditions (i.e., attempting to reproduce behaviours based on the conditions experienced and information available.

Models classified within each of these pedestrian modelling generations adopted a different approach. Each approach can be distinguished by the way the enclosure, population, and behaviour of the population are represented. This plethora of approaches has led to the development of more than thirty models, designed for simulating the evacuation of pedestrians from buildings alone[‡]. Consequently, a review was conducted to determine the feasibility of using a proprietary model for achieving the aims and deliverables of this research.

## Simulating the Evacuation of Pedestrians: Proprietary Models

Based on the advice of Nelson and Mowrer (2002), and the findings of the feasibility review, there were several reasons why the use of a proprietary model was considered undesirable to the research sponsor. For instance, many proprietary models only provide limited information as to how the model works. Essentially black box, accompanying

[‡] Excluding models designed for aviation or maritime evacuation scenarios.

literature also provides little or no evidence to support the validity of the results they produce. Furthermore, the design of these models, to some extent, reflects the purpose for which they were originally intended, the nature of the model developer (e.g., engineer, physical scientist, psychologist, architect), and the computer power available to the developer at the time. Additionally, accessibility to some of these proprietary models is limited to a consultancy basis only. Invariably, in these circumstances, the client is limited to results and analysis published in the final report. Moreover, they are unable to explore results further, or reuse the model to explore a specific scenario at a later date. In light of these limitations, the development of a pedestrian evacuation model was identified as a viable option for this research.

### Developing an Agent-Based Pedestrian Evacuation Model with Repast

Amongst other developments, the evolution of pedestrian evacuation modelling demonstrates a noticeable transition from aggregate to individual-level modelling. In part, increased computer power and storage capacity has only made individual-level modelling practicable in recent times. Essential to the progression of individual-level modelling has been the development of automata approaches, which have been at the forefront of computer modelling research (see Benenson and Torrens, 2004). Two classes of automata tools, cellular automata (CA) and agent-based models (ABMs) have been particularly popular; their use has dominated the research literature. Both classes of tool offer significant development opportunities for pedestrian modelling (see Castle and Crooks, in press). Of particular interest to this research is the agent-based approach, which is capable of representing attributes of individual agents (or groups) and their behaviour within a given system. For example, entities could be programmed to have varying degrees of prior knowledge regarding the building layout (e.g., commuters vs. tourists), or different mobility speeds (e.g., children, adults, mobility impaired, etc). Agents may interact with each other and / or with the environment, of potentially varying conditions (e.g., a smoke filled enclosure). A computer model also allows for a model to be re-run many times, altering parameters to evaluate different situations. Helbing *et al.*, (2000) have developed a model which illustrates these possibilities. The model simulates panicking pedestrians evacuating an enclosure, representing every evacuee as thinking and reacting individuals rather than an identical particle, which generates some interesting outcomes. Moreover, in the context of pedestrian evacuation modelling, the agent-based paradigm will permit the development of a model that is adaptive (i.e., pedestrians can be sensitive to local conditions and have the ability to react to information and options available to them).

The recursive porous agent simulation toolkit (Repast) has been identified as a suitable medium for the development of the agent-based pedestrian evacuation model within this research. One of several reasons for choosing Repast was due to its abstraction of key conceptual requirements involving agent-based models. The toolkit provides suitable functionality to easily create, run, display, and collect data from an ABM. In addition, Repast provides template components. For example, representational elements are already provided, such as the environment in which agents interact (e.g., grid, torus, network, etc.). This functionality allows modellers to spend more time developing the specifics of their model (e.g., agent interactions, behaviours, etc.) rather than setting up the basics of a simulation (e.g., scheduling events to occur, developing a visual display, etc.).

## CURRENT STATE OF RESEARCH

At the time of writing, the prototype model is still in its early stages of development. The model is being developed iteratively, in accordance with general advice regarding the development of ABMs (e.g., Gilbert and Troitzsch, 2005). At present a baseline model has been completed in which pedestrians traverse the underground station using a cost surface as a means of determining the shortest path to available exits. The enclosure layout and cost surface are imported as .ascii files (ESRI format); represented as two-dimensional grids. Figure 2 is a screen capture of pedestrians evacuating the Piccadilly line underground platform at King's Cross St. Pancras underground station.

Subsequent versions will build upon this model, developing more sophisticated pedestrian attributes, behaviours, and interaction rules. In this sense, the final model will evolve analogously to the generations of pedestrian evacuation models identified earlier. At present, the baseline model would be classified as a third generation, or 'ball-bearing' model. Individuals are represented, but only the movement of pedestrians is considered. The model lacks behaviour associated with decision making processes. Subsequent versions will become more advanced. For instance, a rich diversity of behaviours will be incorporated, and pedestrians will be adaptive (i.e., sensitive to conditions around them, reacting to information and options available to them). Further iterations of the model will be posted on the research project website (http://www.casa.ucl.ac.uk/kxsdsses), as and when they have been developed.

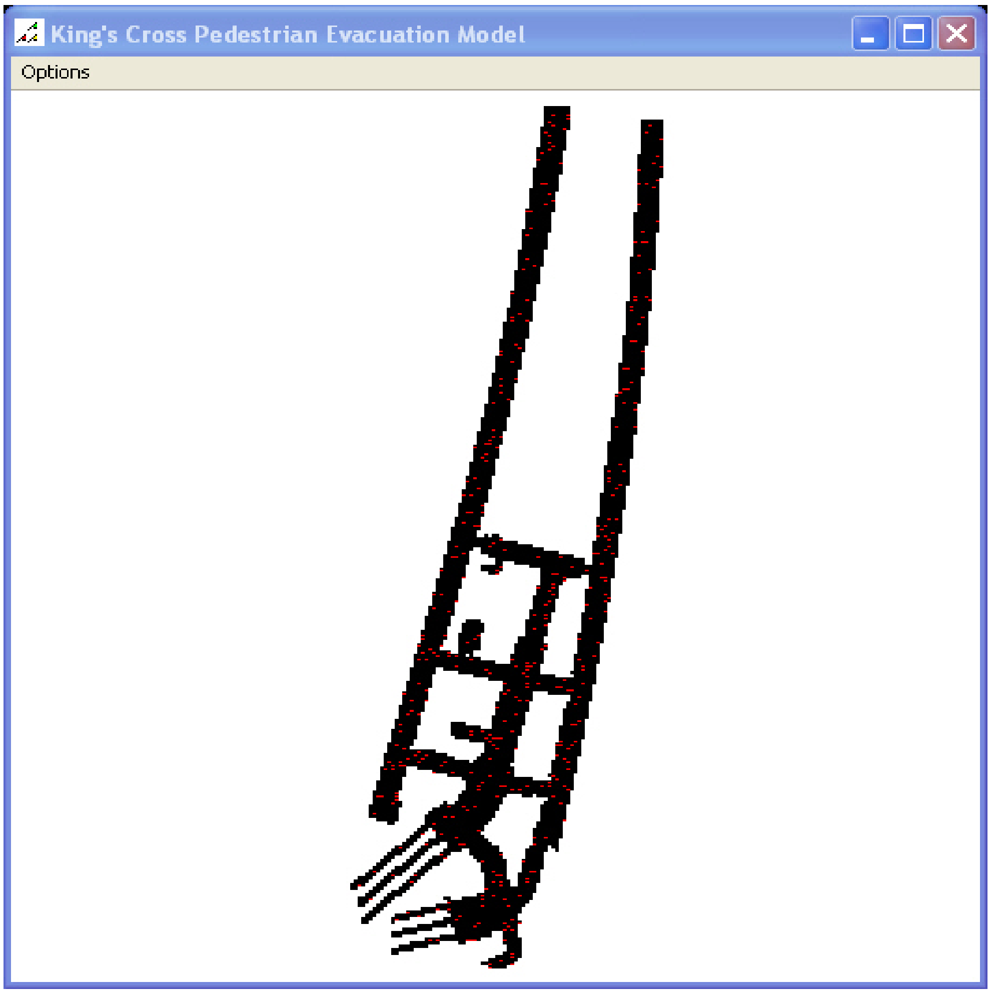


**FIGURE 2:** Pedestrian evacuation model of the Piccadilly line platform at King's Cross St. Pancras underground station.

## CONCLUSIONS

This paper has provided the reader with information regarding the progress of this research to date. The aims and objectives were identified; specifically the evaluation of pedestrian egress from King's Cross St. Pancras underground station. After the redevelopment passenger numbers will have increased and the layout of the enclosure will have significantly altered. Consequently, an overview of principles relating to the modelling of pedestrian evacuation was presented, highlighting a shift in modelling strategies: from aggregate movement, to individual-level movement and behavioural models. The penultimate section of this paper discussed the feasibility of using a proprietary pedestrian evacuation model to achieve the research goal. However, this was deemed as an undesirable option by the research sponsor. The development of agent-based pedestrian evacuation model using the Repast toolkit was chosen as a viable alternative. The paper concluded with a progress update of the prototype pedestrian evacuation model so far.

## AKNOWLEDGMENTS

I gratefully acknowledge the financial support of the Economic and Social Research Council (ESRC, Collaborative Award in Science and Engineering award PTA-033-2004-00034; http://www.esrc.ac.uk), and Camden Primary Care Trust (PCT; http://www.camdenpct.nhs.uk).

## AUTHOR BIOGRAPHY

Christian Castle is a second year Ph.D. student at the Centre for Advanced Spatial Analysis (CASA, University College London) under the supervision of Professor Paul Longley. His Ph.D. is entitled 'A GIS-Based Spatial Decision Support System (SDSS) for Emergency Services: London's King's Cross Redevelopment'. Some of his main research interests include pedestrian evacuation modelling, agent-based modelling, and GIS for emergency evacuation management.

## REFERENCES

Allies and Morrison (2004), King's Cross St. Pancras Underground Station Main Ticket Hall, Available at http://www.alliesandmorrison.co.uk [Accessed on March 4, 2004].

BBC News (2005), In Depth, Available at http://news.bbc.co.uk/1/hi/in_depth/uk/2005/london_explosions/default.stm [Accessed on September 20, 2005].

Benenson, I., and P. Torrens (2004), Geosimulation: Automata-Based Modelling of Urban Phenomena, John Wiley & Sons, London.

Castle, C.J.E., and A.T. Crooks (in press), Agents, Agent-Based Modelling, Toolkits and Repast: A Primer, Centre for Advanced Spatial Analysis (University College London): Working Paper, London.

Castle, C.J.E., and P.A. Longley (2005), "A GIS-Based Spatial Decision Support System for Emergency Services: London's King's Cross St. Pancras Underground Station," in P. van Oosterom, S. Zlatanova, and E.M. Fendel (eds.), Geo-Information for Disaster Management, Springer, Berlin.

DoH (2004a), Beyond a Major Incident, Department of Health, Available at http://www.dh.gov.uk/assetRoot/04/09/82/58/04098258.pdf.

DoH (2004b), Planning for Major Incidents: the NHS Guidance, Department of Health, Available at http://www.dh.gov.uk/assetRoot/04/07/36/32/04073632.pdf.

Fennell, D. (1988), Investigation into the King's Cross Underground Fire, HMSO, London.

Fruin, J.J. (1971), Pedestrian Planning and Design, Metropolitan Association of Urban Designers and Environmental Planners, New York.

Galea, E.R., and S. Gwynne (2004), Principles and Practice of Evacuation Modelling (6th Edition), Fire Safety Engineering Group - University of Greenwich, London.

Gilbert, N., and K.G. Troitzsch (2005), Simulation for the Social Scientist, Open University Press, UK.

Gwynne, S., and E.R. Galea (1997), A Review of the Methodologies Used in the Computer Simulation of Evacuation from the Built Environment, Fire Safety Engineering Group - University of Greenwich, London.

Hallén, B., and P. Kulling (1990), The Fire at the King's Cross Underground Station: November 18, 1987, Socialstyrelsens, Stockholm.

Helbing, D., I.J. Farkas, and T. Vicsek (2000), "Simulating Dynamical Features of Escape Panic," Nature, 407: 487-490.

Kwan, M.-P., and J. Lee (2005), "Emergency Response after 9/11: The Potential of Real-Time 3D GIS for Quick Emergency Response in Micro-Spatial Environments," Computers, Environment, and Urban Systems, 29: 93-113.

LUL (2006), King's Cross, Available at http://www.tfl.gov.uk/tube/company/projects/kings-cross/default.asp [Accessed on August 3, 2004].

Nakanishi, Y., K. Kim, Y. Ulusoy, and A. Bata (2003), Assessing Emergency Preparedness of Transit Agencies: A Focus on Performance Indicators, Annual Transportation Research Board, USA.

Nelson, H.E., and F.E. Mowrer (2002), "Emergency Movement," in P.J. Dinenno (ed.) SFPE Handbook of Fire Protection Engineering (3rd Edition), National Fire Protection Association, Quincy, MA, USA, pp. (3-367)-(3-380).

Predtechenskii, V.M., and A.I. Milinskii (1978), Planning for Foot Traffic Flow in Buildings, Amerind, New Delhi.

Stahl, F.I. (1978), "Preliminary Findings Concerning the Validity of 'BFIRES': A Comparison of Simulated with Actual Fire Events," in B.M. Levin and R.L. Paulsen (eds.), Second International Seminar on Human Behaviour in Fire Emergencies: October 29 - November 1, National Bureau of Standards.

## Social Ecology

# SANTA FE ON FIRE: AGENT-BASED MODELING OF WILDFIRE EVACUATION DYNAMICS

JOSHUA THORP*, RedfishGroup, Santa Fe, NM
STEPHEN GUERIN, RedfishGroup, Santa Fe, NM
FRANK WIMBERLY, RedfishGroup, Santa Fe, NM
MICHELLE ROSSBACH, City of Santa Fe Fire Department
OWEN DENSMORE, RedfishGroup, Santa Fe, NM
MICHAEL AGAR, Ethknoworks, Santa Fe, NM
DOUGLAS ROBERTS, RTI International, Nambe, NM

## ABSTRACT:

This paper reviews a recent RedfishGroup project for the City of Santa Fe Fire and Police Department integrating wildfire simulation and agent-based modeling of traffic dynamics. Visualizations of the evacuation dynamics were used to communicate with citizens dangers faced during evacuations and to explore when shelter-in-place is a reasonable alternative to evacuation. Additional uses of the model were for use in "table top" exercises conducted by multi-agency emergency response teams and for identification of intersections and road segments expected to have high congestion. The paper reviews the modeling approach and the ecology of software tools used on the project. ESRI ArcView was used for preparing GIS data sources of housing, street centerline data, elevation maps and geo-referenced aerial infrared imaging for wildfire fuels modeling. Netlogo was used for early prototyping of both cellular automata-based wildfire models and for network-based traffic models. FARSITE was later integrated for wildfire simulation. Open-source Blender3D was a central tool for this project used for both agent-based modeling and 3D visualization. The traffic agent-based model of 50,000 cars on the Santa Fe road network was written directly in Blender using Python. The traffic visualization was combined with wildfire simulation output from FARSITE and rendered in 3D in Blender. Simulation results were deployed for community use in Google Earth and Quicktime visualizations generated by Blender.

**Keywords:** Agent-based modeling, wildfire evacuation, emergency planning, Santa Fe,,traffic simulation, wildfire modeling, NetLogo, Python, Blender

## BACKGROUND, GOALS AND APPROACH

The City of Santa Fe, New Mexico faces a wildland fire threat as do many other communities in the United States. In the City there are significant numbers of homes nestled in wild vegetation along narrow roads. Often there is only one ingress/egress to those neighborhoods. Fire Department officials began wondering in 2003 about whether residents would be able to evacuate their homes safely in the event of a fast-moving wildfire, and they began teaching citizens to be ready to evacuate or shelter-in-place depending on the circumstances.

*Corresponding author address:* Joshua B. Thorp, RedfishGroup, 624 Agua Fria Street, Santa Fe, NM 87501; email: josh@redfish.com

In Santa Fe, City officials initiated a collaboration with agent-based modelers and visualization scientists at RedfishGroup and with fire scientists at Anchor Point Group of Boulder, CO to develop models of the interaction between massive but not unrealistic wildfires and traffic as it would exist in an evacuation (the 48,000 acre Cerro Grande Fire, which destroyed over 200 homes in Los Alamos, was visible from Santa Fe). The goals are not only disaster response planning (Figure 1) but also education of citizens and first responders. City emergency response professionals hope that the ability to visualize a fast-moving fire and its accompanying smoke and their dramatic effect on traffic will serve as an effective educational tool and as a means to illuminate the controversy.

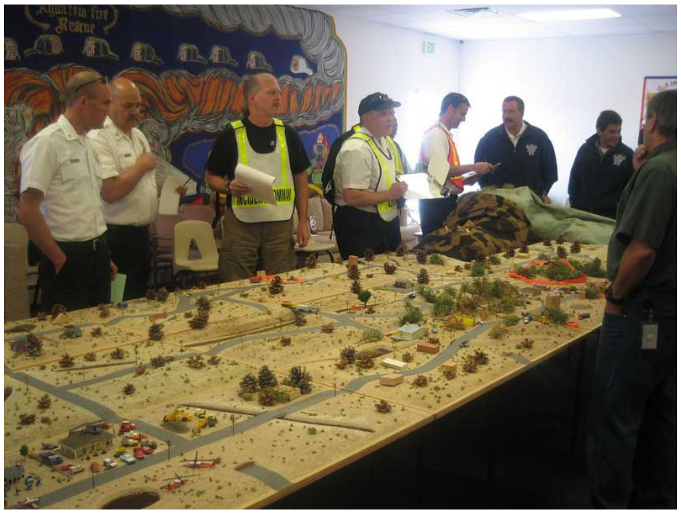


**Figure 1:** Agent-based models and visualizations were designed to augment existing "table top" exercises used by multi-agency emergency planners.

The local effort has not been without controversy, however; many wildland firefighters (and police officers) in the area insist that all citizens will be able to evacuate in all circumstances. The controversy over "sheltering-in-place" during wildfires exists nationwide and is described more in-depth elsewhere ("Emerging Issues", 2004). As a quick point, shelter-in-place is not a recommended option in many communities because of the type of wildfires expected and the prevalence of certain materials in construction (eg shake-shingle roofs). Some experts within the Santa Fe Fire Department and elsewhere, however, believe that due to local conditions, shelter-in-place is a viable alternative worth communicating to the citizens in Santa Fe. Local adobe and stucco construction with flat tar and gravel roofs are estimated to withstand direct flame for about 25 minutes while the type of wildfire expected (crown fire moving through pinon-juniper fuels) is expected to pass within 7 minutes. It is argued that if the evacuation routes are congested, a family may be safer in their house than stuck on the road in their car.

## THE APPROACH

The three components of the work are: Fire Modeling, Traffic Modeling, and Visualization. The architecture that combines these will be described below.

### Fire Modeling

In early explorations we developed a cellular automata (CA) approach to modeling the spread of fire over a terrain. In this approach we divided the area into a grid of small squares. When a fire ignites in a cell, the probability of a neighboring cell igniting depends on the wind, the elevation difference and the fuel available. Simply speaking, a cell is more likely to ignite if it is downwind and uphill from an already burning cell.

As the project advanced, however, more realism and credibility of the fire model were needed and we acquired Farsite for the fire modeling component of our work. Farsite is widely used in the Forest Service and others with an interest in wildfire science. It is part of a suite of programs, including FlamMap and Behave, that are based on first principles of fire physics. One of the pivotal papers in the area was published by Rothermel in the early 1970's (Rothermel 1972) and all of these programs are based on the physics described there. He created a model that combined conservation of energy and empirical measurements of fire spread in wind tunnel experiments.

Sullivan and Knight (2004) point out that CA models of "bushfires" (wildfires in Australia) have been less successful than hoped. According to them, models based on nearest neighbor ignition rules are too naïve. They say that empirical fire spread models such as Rothermel's are not true CA models since they are calculated on a raster or lattice. They propose a hybrid approach which they call cellular automata/semi-physical models.

The main input to a Farsite run is a landscape file. This is a file which combines elevation, slope, aspect, fuel and canopy data. In addition, a Farsite run take in to account weather data including wind, temperature and humidity. For most wildfire simulations in this project, we used extreme conditions: 98 degrees F, 40 MPH winds and 10% relative humidity. These values are the norm in April-May when wildfire risk is at its highest in Santa Fe.

Initial landscape files were constructed from digital elevation maps (Figure 2), landcover and canopy data (Figure 3) that we obtained from the USGS web site. We used ArcGIS 9 to assemble the aspect and slope data from the elevation data. In parallel with our project, the city of Santa Fe contracted with Anchor Point Group to develop a landscape file for its wildland/urban wildfire studies. Their work included analyzing the fuels in the area and characterizing the ignitability of the local structures. The area of interest was flown with aerial photography (6"/pixel resolution) in the spring of 2006 which included infrared imaging to better resolve fuel and canopy cover. We produced animations early in the project showing this imagery *texture-mapped* onto 3D topography (Figure 4).

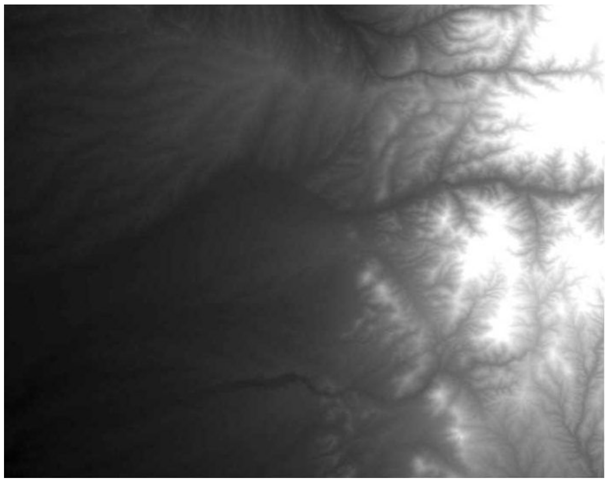


**Figure 2:** Digital Elevation Model of Santa Fe

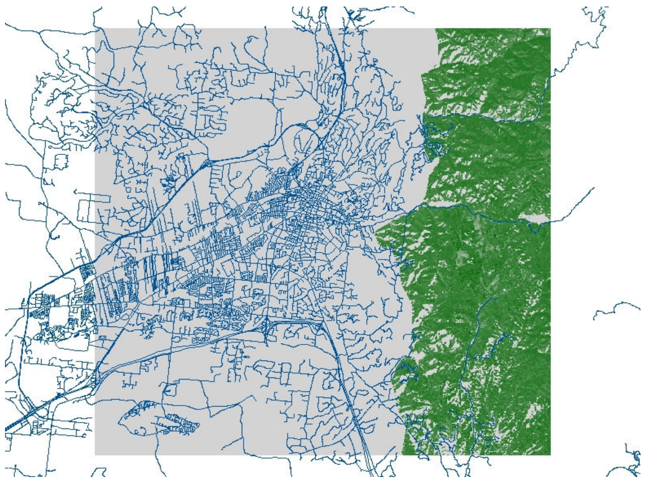


**Figure 3:** Canopy Map for Santa Fe

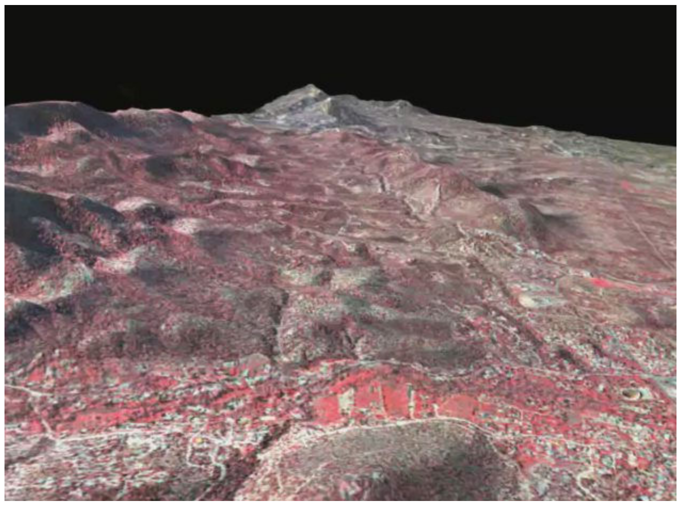


**Figure 4:** Infrared aerial photography *texture-mapped* to Santa Fe topography

Farsite has a variety of output options including the ability to create raster files describing the simulated fire. The types of raster files include, for example, time of arrival (when does the fire reach each point in the area), flame length, fireline intensity, spread direction and crown fire activity. These can be displayed in ArcGIS or, more importantly for our purposes, they are in a standard format which is easy for our visualization software to render.

After receiving the landscape file from Anchor Point, we "ignited" a number of simulated fires around Santa Fe in order to get simulations of the fire/traffic interactions in a number of kinds of city neighborhoods. We focused, however, on areas which we knew would be challenging to evacuate in the event of an aggressive crown fire. The body of research produced by Cova et al. (Cova 1997, 2002, 2003, 2005a, 2005b) provides good background and many valuable techniques for characterizing relative neighborhood evacuation risk.

## TECHNICAL DESCRIPTION OF THE WILDFIRE EVACUATION MODEL

The Wildfire Evacuation Model explores the interaction of two complex systems; a traffic model and a wildfire model. Combining two models with different time steps and architectures is a challenge in agent-based modeling, especially when the models are directly affecting each other. Cars blocking roads in an emergency evacuation could hinder fire crews preventing them from reaching the fire. This would cause the fire crew's attack on the fire to be hampered by the dynamics of the traffic model with a resulting change in fire model behavior as compared to an ideal fire attack (i.e. one with immediate fire crew presence). As the fires most likely to affect Santa Fe would be fast moving (up to 5 km/hour), we assume that only fires where the initial attack was unable to control the fire would rise to the level of evacuation. Thus this model can be considered the worst-case situation of no fire attack allowing the assumption that fire dynamics are independent of the traffic dynamics. This leaves the fire model free to be run independently of the traffic model and to serve as input to the traffic model. Below we detail the traffic model GIS inputs and traffic behavior including the wildfire model impact on the traffic model.

### Traffic Model

Initial explorations in modeling intersections was prototyped in NetLogo (Wilensky, 1999) with the road network modeled as a graph with nodes and edges (Figure 5). As an aside, our team continues to find NetLogo invaluable as a powerful rapid prototyping environment for agent-based models while it remains approachable to domain experts that don't normally identify themselves as coders.

As the number of cars in the traffic model increased up to 70,000 and the GIS components grew in importance, development was transitioned to Python (http://www.python.org) and visualizations in Blender3D (http://www.blender.org), an open-source 3D modeling and animation environment (Figure 6). Blender has integrated python scripting and was used as both view and control of the model. Input to the traffic model consists of many GIS data files. GIS data comes in raster and shape (vector) file formats. In our model we use both type. These files were imported into the Python implementation of the traffic model using the Geospatial Data Abstraction Library (GDAL) (http://www.remotesensing.org/gdal/) for raster type and the Python bindings to the Shapefile C Library (shapelib) (http://shapelib.maptools.org/) for shapefile type data. These format types are just like formats one will have dealt with in paint programs such as Adobe's Photoshop (raster) and Illustrator (vector). In simple terms raster files consist of a grid of pixels with a certain width and height, with each pixel having some value (such as elevation). A shapefile has a vector-based format that consists of descriptions of shapes such as lines, curves made up of lines, called poly-lines, and polygons.

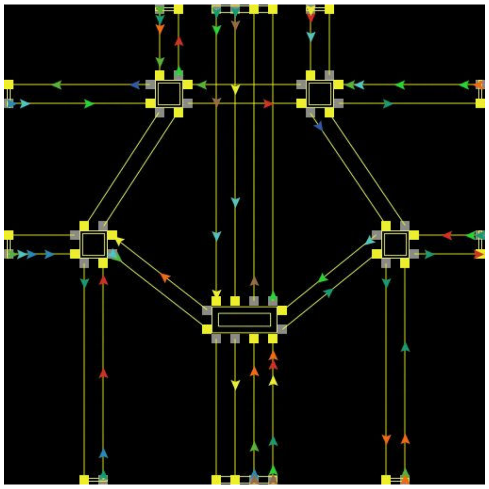


**Figure 5:** Traffic prototypes in NetLogo

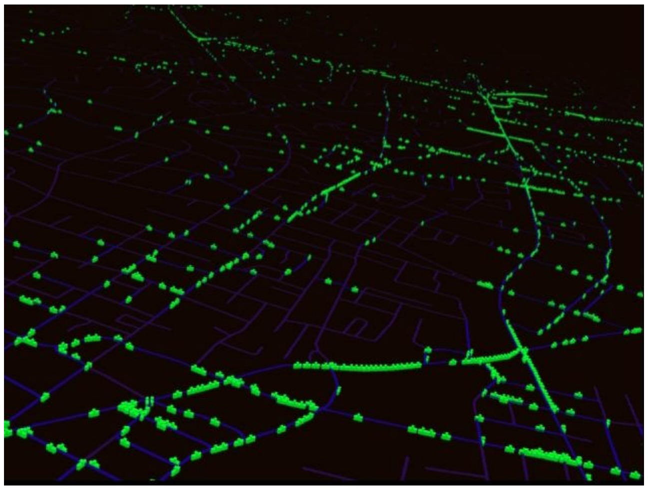


**Figure 6:** Full-scale traffic models implemented in Python/Blender

Agent-based models typically consist of agents interacting with each other in an environment. In general an environment may consist of cells in a grid with a topology of four or eight neighbors or a network (graph) topology. The traffic model's environment is constrained to the topology of the streets of Santa Fe as input in a shapefile. The shapefile consists of shapes called poly-lines -- shapes that are made up of a number of line segments -- and is a description of the 'center line' of the streets of Santa Fe. From this information a road network is built. Roads consist of multiple lanes proceeding from a directional distance of zero at one end to the total length of the road at the other. Each road has at least one lane that has directionality and a distance of zero at one end of the road proceeding to the length of the road at the other. Depending on the number of lanes and whether the road is 'one-way', roads may have lanes beginning at both ends of the road. The cars exist on lanes that meet at intersections and can pass from one lane to another at intersections based on the connectivity of the intersection (Figure 7). Lanes that enter an intersection are connected to lanes that exit that intersection.

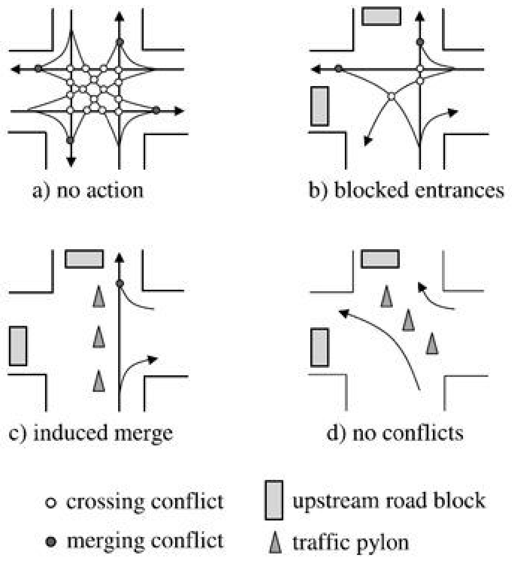


**Figure 7:** Modeling of intersections with merge and crossing conflicts based on Cova (2003)

Car agents are aware of their neighbors sharing a lane, a road, or an intersection. Cars are not aware of their absolute positions in space such as elevation or north or east location as they exist at a distance along a lane in this world of roads and intersections. Cars are aware of their neighbors in this space, avoid collisions, accelerate and decelerate, and turn at intersections following a mixture of local rules and seeking a destination.

Predetermined sets of origins and destinations are defined in the model. At the beginning of a model run cars are assigned a destination and an origin location based on the real locations of homes in Santa Fe. This is informed by the locations of actual Santa Fe homes from another GIS data file. Destination points may include an evacuation center or road that leads out of town and conceivably be extended to various other destinations, such as area hotels.

All intersections in the model are informed of the distance to destination points through an initial ‘flood-fill’ from these points. This technique is useful when there are only a handful of possible destinations; however, it is memory and CPU-intensive for a large number of points. This evacuation model is not so much concerned with the destination as with the dynamics of leaving the evacuated areas. Careful selection of evacuation points allows us to produce realistic sets of local routes out of the evacuated area. From that point cars, follow a set of local rules choosing particular routes at intersections according to the capacity of roads, their distance to the goal as determined by the flood fill, with aversion to changing roads too often, and certain amount of random noise that we refer to as the ‘tourist factor’ that serves to give drivers an incomplete knowledge of the roads. Added to the above is general aversion to driving on or near roads that are near active fires. Possible further work in the social modeling of the evacuees would be necessary to determine where in particular populations would drive to, including friends and family homes, hotels, and other locations outside of the evacuated area.

## Impact of Wildfire Model on Traffic Behavior

The fire model output from the Farsite wildfire model consists of eight raster files of which the “Time of Arrival” (TOA) and “Flame Length” (FML) files are currently used as input to the traffic model. The TOA file consists of a geographic area covered by a grid of pixels. Pixel values are either the fractional hour that fire first enters that pixel or “–1” if no fire ever entered the area. Shape files generated from simulations in Farsite were imported into Google Earth Pro (http://earth.google.com/) so that they could be shared with citizens of Santa Fe as .kmz layers. The Pro version of Google Earth is only required to generate the .kmz files from .shp files. The .kmz files are then distributable to end-users with the free version of Google Earth. See http://www.redfish.com/wildfire for more examples.

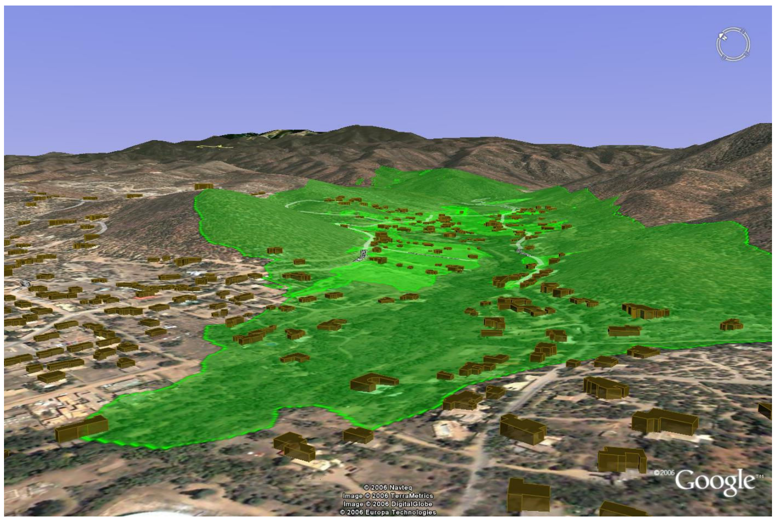

©2006 Google
© 2006 Navteq
Image © 2006 TerraMetrics
Image © 2006 DigitalGlobe
© 2006 Europa Technologies

**Figure 8:** Extent of wildfire in the first hour. GIS shapefiles imported into GoogleEarth

The traffic model takes this information as input and sorts fire points according to start of fire. As the traffic model reaches the time when a fire first appears that point is added to a set of current fires and roads that are near these fires are impacted as are roads that are located downwind of the fire. These points last for a certain time based on the estimate of duration of fire given the fuels at that location, before they are removed from the list. This information is made available to nearby roads, which can then be queried by cars traveling on these roads for deciding whether to turn at an intersection. Finding fire or smoke on a road, the car will avoid that road.

## Blender 3D

Visualization of the model is produced in the Blender environment. Blender is a sophisticated 3d modeling, animation, and rendering environment with Python scripting built in.

This allowed us to control the run of the model from within the Blender tool and render cars, fire, and Santa Fe elevation data from the area of interest Digital Elevation Model (DEM) in 3D animations (Figure 9).

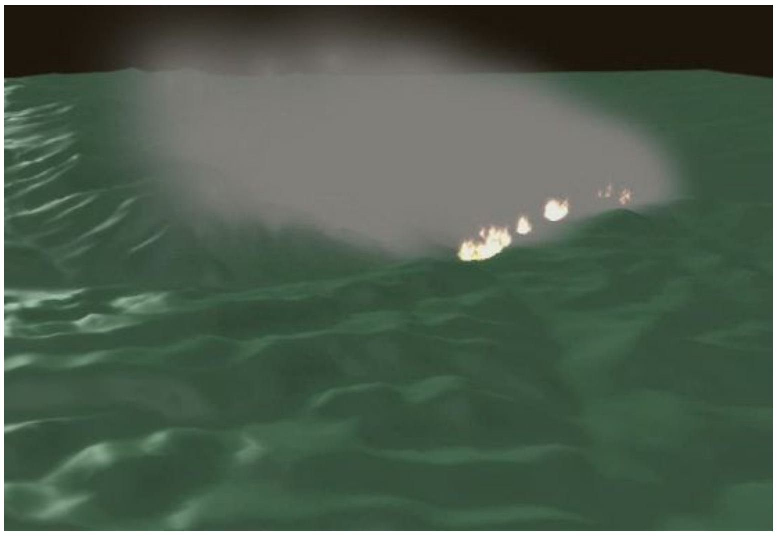


**Figure 9:** Rendering of smoke, Farsite *Time of Arrival* and *Flamelength* with Blender3D

Animations, reports, images and source code (released under LGPL) for this project can be found at http://www.redfish.com/wildfire

## RESULTS AND FUTURE WORK

At the beginning of this project, we anticipated that the primary value delivered would be the evaluation and optimization of traffic evacuation routes and plans. However, we discovered early on that existing evacuation plans in the city were relatively unsophisticated – eg, "go downhill and meet up at the local rodeo grounds, the planned emergency shelter". The cause for this was not due to an unsophisticated city management – In fact, we found the employees within the city to be among the most professional of any organization we encountered. The unsophisticated plans, we suspect, is more due to that fact wildfires can occur practically anywhere in a city. This is in contrast to something like hurricane evacuation planning, where planners know the threat will start from the coast and move inland and can publish planned evacuation routes. We now think the best you can do in wildfire planning is run different scenarios and look for common patterns in traffic congestion. For instance, which intersections consistently become bottlenecks and warrant police presence to direct traffic. Additional areas of research can be into more demand-based scheduling algorithms for traffic signals along the lines of Gershenson, 2005.

Additionally, a real value delivered can be to raise the awareness of the citizens to the possible contingencies. As the project progressed efforts focused more than planned on modeling the wildfire dynamics than the traffic dynamics so that we could get a better sense of the typical risk Santa Fe was facing. Visualizations of potential wildfire scenarios have been very effective at raising public awareness and they serve as backdrop to discussions involving hard decisions facing the community like fuel thinning projects.

Visualizations of the wildfire dynamics coupled with trapped cars on narrow roads helped frame the question of evacuation vs. shelter-in-place for citizens. Prior to viewing the visualizations, many citizens assumed they could outrun a fire. They did not realize fire-spotting could advance in front of them or how much of an issue smoke would be during evacuation. The project has helped people realize that they may be shelter-in-place may be a last-resort option that they may be forced to take. Preparing their house to survive a wildfire then becomes less about saving a structure and more an issue of potentially saving their family's lives.

Work is ongoing to further make the simulations accessible to emergency planners. We're currently working with Sandia Labs to develop *augmented-reality* tabletops that will allow us to project simulations down on to tables while allowing planners to manipulate real-world objects like fire trucks, fire attack aircraft, wind vectors, etc. on the simulation table. Cameras and RFID tags embedded in the objects allow for real-time feedback on object positions to the simulation. We expect this will provide a more collaborative environment unmediated by personal computer terminals.

## ACKNOWLEDGMENTS

RedfishGroup would like to thank employees of the City of Santa Fe that provided much time helping us understand the issues on this project. We would also like to thank the Santa Fe Institute for hosting a workshop where the team was able to meet with city planners and review the models. And lastly, we'd like to thank members of the FRIAMGroup in Santa Fe for their feedback and suggestions on this project.

## REFERENCES

Cova, T.J., and Church, R.L. (1997) Modelling community evacuation vulnerability using GIS. International Journal of Geographical Information Science, 11(8): 763-784

Cova, T.J., and Johnson, J.P. (2002) Microsimulation of neighborhood evacuations in the urban-wildland interface. Environment and Planning A, 34(12): 2211-2229

Cova, T.J., and Johnson, J.P. (2003) A network flow model for lane-based evacuation routing. Transportation Research Part A: Policy and Practice, 37(7): 579-604

Cova, T.J. (2005) Public safety in the urban-wildland interface: Should fire-prone communities have a maximum occupancy? Natural Hazards Review, 6(3): 99-108

Cova, T.J., Dennison, P.E., Kim, T.H., and Moritz, M.A. (2005) Setting wildfire evacuation trigger-points using fire spread modeling and GIS. Transactions in GIS, 9(4): 603-617

Emerging Issues in Wildland Fire Protection. When Wildfire Threatens: Should Residents Stay or Evacuate? Journal of the National Fire Protection Association. Wildland Fire Management Section. October, 2004.

Gershenson, C. (2005). Self-Organizing Traffic Lights. Complex Systems 16(1): 29-53

Lindroth, R. (2004). Community defense from wildfire, an international comparison. An applied research project submitted to the National Fire Academy as part of the July, 2004 Leading Community Risk Reduction course.

Oaks, D. (2000). Fight or Flight? Fire Chief. April 1, 2000

Rothermel, R. C. 1972, A Mathematical Model for Predicting Fire Spread in Wildland Fuels, Ogden, UT, U.S. Department of Agriculture, Forest Service, Intermountain Forest an Range Experiment Station Research Paper No INT-115.

Rothermel, R. C. 1972, A Mathematical Model for Predicting Fire Spread in Wildland Fuels, Ogden, UT, U.S. Department of Agriculture, Forest Service, Intermountain Forest an Range Experiment Station Research Paper No INT-115.

Sullivan, A. L., I. K. Knight, A Hybrid Cellular Automata/Semi-physical Model of Fire Growth, Proceedings of the 7th Asia-Pacific Conference on Complex Systems, Cairns, Australia, December 2004.

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

# STRAWBERRY FIELDS FOREVER? DEVELOPING A SOCIO-ECOLOGICAL SIMULATION GAME WITH “AGRO-DIVERSITY ver.2” FOR EXPLORING BIODIVERSITY MANAGEMENT ISSUES IN SUSTAINABLE AGRICULTURE

E.N. SPEELMAN, El Colegio de la Frontera Sur, Mexico

L.E. GARCIA-BARRIOS,[*] El Colegio de la Frontera Sur, Mexico

## ABSTRACT

Biodiversity loss associated to agricultural intensification is one of the most important issues on the global change agenda. Till recently, biodiversity within agricultural systems was perceived by mainstream science as a symptom and cause of economic inefficiency. Today it is increasingly recognized that biodiverse agriculture can be productive and play an important role in food security, in providing ecosystem services and in creating a wildlife friendly landscape matrix for biodiversity conservation. However, encouraging stakeholders towards maintaining or shifting to agrodiverse systems faces many social and ecological challenges: (1) in many cases, it is still unclear how much diversity is actually needed to provide basic inputs and services at the field level, and how specific ecological networks improve agro-ecosystem functions, (2) the relations between agricultural sustainability and agro-diversity management are not-easily grasped or demonstrated, as both are complex and long-term processes, (3) labor and/or financial constraints as well as transition costs can limit agrodiverse system adoption, (4) contrasting agrodiversity management strategies can coexist in the same landscape and influence each other both in positive and negative ways. Agent-based socio-ecological models –coupled with role-playing games- can be powerful tools to understand and address the issues, tradeoffs and conflicts associated with the adoption and sustainable management of agrodiversity. In this paper we briefly present a “small” Agent-Based Socio-Ecological Model (AGRODIVERSITY ver. 2 programmed in Netlogo 3.1) and explore the model’s current capacity to produce interesting scenarios for developing a role-playing game between virtual and/or flesh-and-blood stakeholders involved in agrodiverse system management.

**Keywords**: agro-diversity, sustainability, agent-based model, stakeholders games.

[*] *Corresponding author address*: Luis E. García-Barrios, El Colegio de la Frontera Sur (ECOSUR), Carretera Panamerica y Perifierico Sur, CP 29290, San Cristobal de las Casas, Chiapas, Mexico; e-mail: lgarcia@sclc.ecosur.mx.

## INTRODUCTION

Since the past century, industrial agriculture has come to depend excessively on high levels of fossil fuel and agrochemicals, and has dramatically reduced biological diversity at the field and landscape level. These trends are now extending to peasant agriculture, as market-oriented production and temporal or permanent rural migration is inducing small farmers to shift from biodiverse multipurpose agroecosystems to cash monocrops, and from labor-intensive to cash-intensive management practices. Such high agrochemical input monocrops have also proven to be costly, unhealthy for people and their environment, and ultimately unsuccessful due to resistance developed by pests (Altieri, 1999).

Pressure on peasant agricultural systems and their natural resources has consequently increased, threatening the continuation of these systems that are essential for the food security of households, communities and regions (García-Barrios and García-Barrios, 1992; Goodman and García-Barrios, 2004). As much as 1.5 billion people earn their livelihood from peasant agriculture (Chambers, 1993; Rosset, 2001).

There is a strong need for more sustainable alternative peasant systems. Research is starting to show that functional agro-diversity can be useful and even essential for the long-term sustainability of agriculture, of natural resource use and of biodiversity (Swift et al., 2004). Diversified agricultural systems perform essential eco-system services and can assist in in-situ conservation of (agricultural) biodiversity in creating a wildlife friendly agricultural matrix at the landscape level (Vandermeer and Perfecto, 2005). In Latin America many initiatives directed to improve current peasant systems are based on diversifying systems (Speelman et al. 2006).

However, encouraging stakeholders towards perceiving functional agro-diversity as an important and profitable resource faces many social and ecological challenges: (1) in many cases, it is still unclear how much diversity is actually needed to provide basic inputs and services at the field level, and how specific ecological networks improve agro-ecosystem functions, (2) concepts such as agricultural sustainability and agro-diversity management are not-easily grasped; they are complex and long-term processes, (3) functional agro-diversity is frequently a condition far from equilibrium, which requires human management to be sustained. Diverse systems can be managed in ways differing in economic and environmental costs. Outcomes and trade-offs involved in different management strategies can change and even shift over time.

Research is starting to produce modeling tools addressing sustainability issues in complex multi-stakeholder situations. Some are “big” models that aim at reproducing and predicting real data. They are powerful but usually data-hungry and difficult to parameterize. Others are “small” simulation models that focus more on getting right the multiscale interactions between agents (and the qualitative consequences of these interactions) rather than striving for agent-level details.

We have recently used or developed a number of simulation tools for creating awareness and increasing stakeholder and student understanding of issues involved in the sustainability of simple and complex agricultural systems (García-Barrios and Chakravarty, 2002; Speelman et al. 2006). We are linking some of these models (e.g. García-Barrios and

Pimm, 2006) to role-playing games in which sustainability tradeoffs and conflicts between stakeholders emerge in an agricultural landscape, and need to be resolved.

The joint use of multi-agent simulation and role-playing games is relatively new (Barreteau et al., 2003). This strategy triggers users to get more involved, interact and thereby reframe their strategies, interests and conflicts. Role-Playing games constitute a social laboratory in which human actions can be explored (Barreteau, 2003). Multi-Agent Simulation have been proposed to be well suited for capturing the complex biophysical and socioeconomic settings and long-term behaviors found in peasant NRMS in many developing countries (Berger et al., 2006).

In this paper we briefly present a “small” Agent-Based Socio-Ecological Model; *Agrodiversity ver. 2* (García-Barrios and Speelman, 2006) programmed in Netlogo 3.1 (Wilensky, 1999) and explore the model’s current capacity to produce interesting scenarios for developing a role-playing game between virtual and/or flesh-and-blood stakeholders involved in agrodiverse system management.

## MODEL SPECIFICATION

AGRODIVERSITY ver.2, features four ecological agents interacting in a virtual agroecosystem: (1) a perennial crop, that grows and produces fruit annually - in our story strawberries, (2) an insect population that grows by eating crop foliage and consequently reduces strawberry yield, (3) a spider population that grows by eating these insects, and (4) a perennial grass-like weed that is the host-plant for the spiders, but competes with the crop for space. The four species interact directly and indirectly through an ecological network involving predator-prey relations, commensalism, interspecific competition and intraspecific competition (Figure 1; intraspecific interaction are not shown).

Individual elements of each species population are spatially and explicitly modeled as Netlogo “turtles” in a virtual field that wraps around without borders. All individuals obey a set of probabilistic rules. Plants grow new branch/leaf modules in available empty space; insects move around and eat the crop while spiders sit on weeds and wait for insects to come around. Users can define prey nutritional value and predator voracity and energy loss. The energy balance of each insect and spider is monitored. Above an energy threshold, they reproduce. Below a minimal energy threshold they die.

Users set the initial density and spatial distribution of each species. At any given moment users can perform spatially explicit management: pruning weeds in contact with strawberry plants and insecticide spraying. Both have money and/or labor cost.

(+) Means a direct positive correlation between species abundance and (-) a negative one. By following the signs (both clockwise and counterclockwise), the systemic effects of each species on itself can be deduced. For example, the spider is benefited by the weed but does not directly harm it (commensalisms). Yet it does reduce the weed's growth rate indirectly by favoring the crop, which competes with the weed. By doing so, the spider increases its own food but slows down the growth of its shelter. All species indirectly exert both positive and negative effects on themselves. The consequences for coexistence are explained later in the results section.

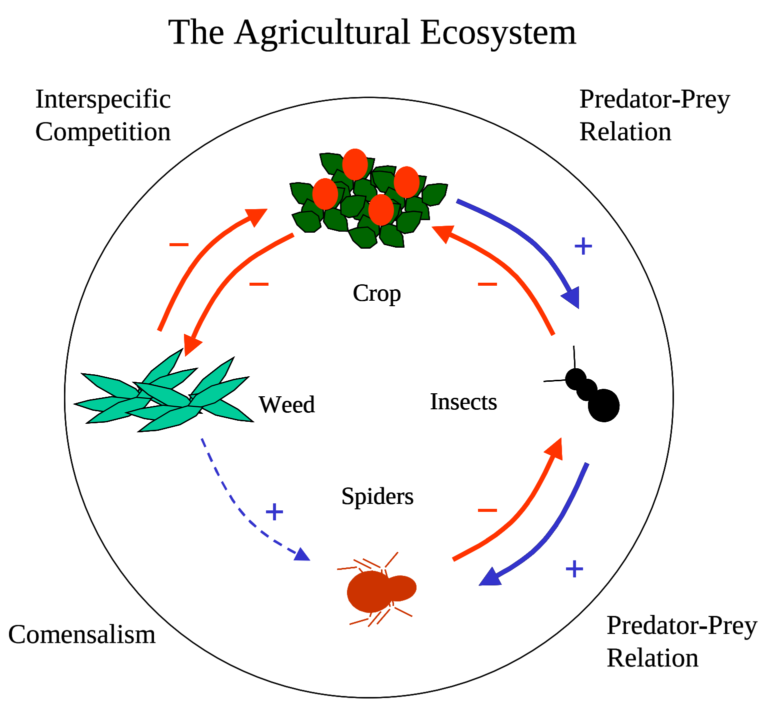
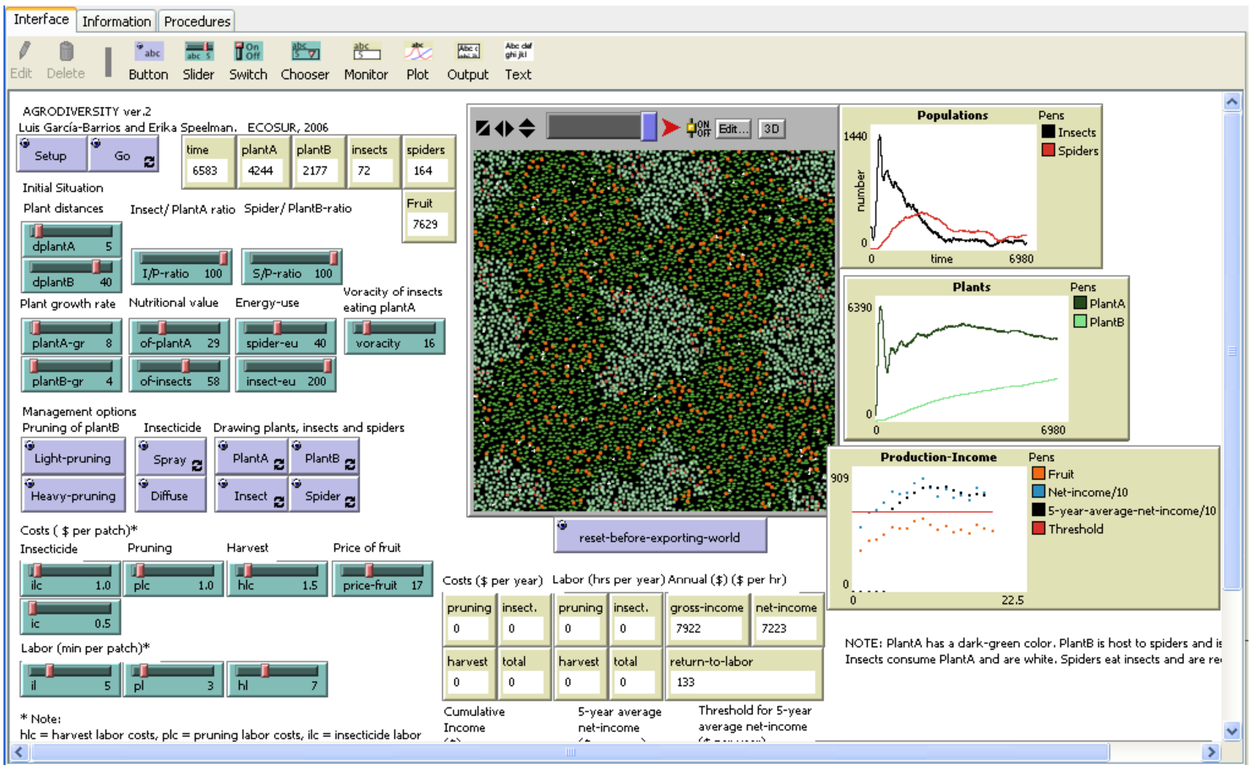


**FIGURE 1** Ecological interaction network of the four species that form the biodiverse agroecosystem.

AGRODIVERSITY ver.2 computes the income generated by the sales of strawberries and discounts labor and management costs made during the year. For simplicity, at this modeling stage the costs/benefit ratio of strawberry production is assumed to remain constant over decades. Annual socio-economic indicators such as gross-income, net-income, and return to labor are calculated. An economic threshold based on the 5-year moving average of net income is arbitrarily established to analyze the performance of the system. It sets the point at which the system is no longer economically viable and the farmer decides to abandon his strawberry field forever.

Figure 2 shows the model's interface. The latter provides a wide range of buttons and sliders that modify the ecological and socio-economic parameters of the system, as well as tools to perform management. The agent populations and the system's production and income are graphed over time. These graphs help the user to quickly see what happens in his strawberry field. The virtual field shows the crop and the weeds in two different shades of green. Users should keep in mind that insect and spider size has been exaggerated relative to plant modules. Otherwise, their movement and abundance would not be apparent.



AGRODIVERSITY ver.2
Luis García-Barrios and Erika Speelman. ECOSUR, 2006

time 6583
plantA 4244
plantB 2177
insects 72
spiders 164
Fruit 7629

Initial Situation
Plant distances — dplantA 5; dplantB 40
Insect/ PlantA ratio — I/P-ratio 100
Spider/ PlantB-ratio — S/P-ratio 100
Plant growth rate — plantA-gr 8; plantB-gr 4
Nutritional value — of-plantA 29; of-insects 58
Energy-use — spider-eu 40; insect-eu 200
Voracity of insects eating plantA — voracity 16

Management options
Pruning of plantB — Light-pruning; Heavy-pruning
Insecticide — Spray; Diffuse
Drawing plants, insects and spiders — PlantA; PlantB; Insect; Spider

Costs ( $ per patch)*
Insecticide — ilc 1.0; ic 0.5
Pruning — plc 1.0
Harvest — hlc 1.5
Price of fruit — price-fruit 17

Labor (min per patch)* — il 5; pl 3; hl 7

* Note:
hlc = harvest labor costs, plc = pruning labor costs, ilc = insecticide labor

Costs ($ per year) — pruning 0; insect. 0; harvest 0; total 0
Labor (hrs per year) — pruning 0; insect. 0; harvest 0; total 0
Annual ($) — gross-income 7922; net-income 7223
($ per hr) — return-to-labor 133

Cumulative Income
5-year average net-income
Threshold for 5-year average net-income

NOTE: PlantA has a dark-green color. PlantB is host to spiders and is
Insects consume PlantA and are white. Spiders eat insects and are re

**FIGURE 2** AGRODIVERSITY ver.2 User interface. The Virtual World screen, graphs, parameter rulers, buttons and monitors are all built-in Netlogo 3.1 features. The user can change a number of biological, management and economic parameters and follow the simulation of each agent in the virtual field, and the auxiliary graphs and monitors. The details of this interface are explained in the AGRODIVERSITY ver.2 tutorial (García-Barrios and Speelman, 2006).

Simulation results explained below show that insect/weed control is necessary for the ecological and economic sustainability of the AGRODIVERSITY agroecosystem. They also show that contrasting management strategies have different sustainability tradeoffs. Interestingly, they have the potential to generate conflict when put together in the same landscape. To explore these issues we developed a program upgrade that includes two neighboring strawberry fields managed by virtual farmers with contrasting management strategies. The connectivity between the fields is user defined. The virtual farmers have the ability to learn from their own and their neighbor's experiences and consequently change their management strategy at any stage.

## TESTING THE MODEL´S CAPACITY TO PRODUCE SCENARIOS RELATED TO RELEVANT AGRODIVERSITY MANAGEMENT ISSUES

Species in diverse systems must interact in such a way that a long-lasting community is self-organized, and species proportions conduce to an optimum crop production.

Conversely, A proper balance between the co-existing species (crop, host plant, pest and natural enemy populations) is necessary so that their ecological interactions allow the multispecies system to become self-organized and persist over time.

The first step was to calibrate the model for this purpose. In this model the simulated species interact and influence each other directly and indirectly through a complex ecological network. Figure 3 graphically shows a typical simulation scenario after calibration. All four populations fluctuate over time and change in population size of one species affects the other species. The predator-prey relations between crop and insects and between insects and spiders make all three species oscillate. The weed grows slower than the crop but has no predator Every time foraging insects reduce the crop modules, the weed profits and occupies more space. In the long run this steadily reduces the crop population and the fauna that depends on it. The equilibrium state of the system is therefore a mono-specific weed stand. Our simulation results showed that the system's ecology will always drive it to this non-productive weed community equilibrium in the absence of external weed and insect control.

The best scenario for the agrodiverse system with no external insect/weed control starts with a very low weed population. In the early stage of weed growth, this species fosters diversity and increases income. In a later stage, it has the opposite effect; it has expanded to an extent that it limits the growth of strawberry plants. The ecological system first self-organizes itself into a diverse community and later evolves into a monospecific weed field. The economic consequences is a hump-shaped income curve; eventually the system drop below the economic threshold (Figure 4). Although this unmanaged diverse system persists for a relatively long time, strictly speaking it is unsustainable.

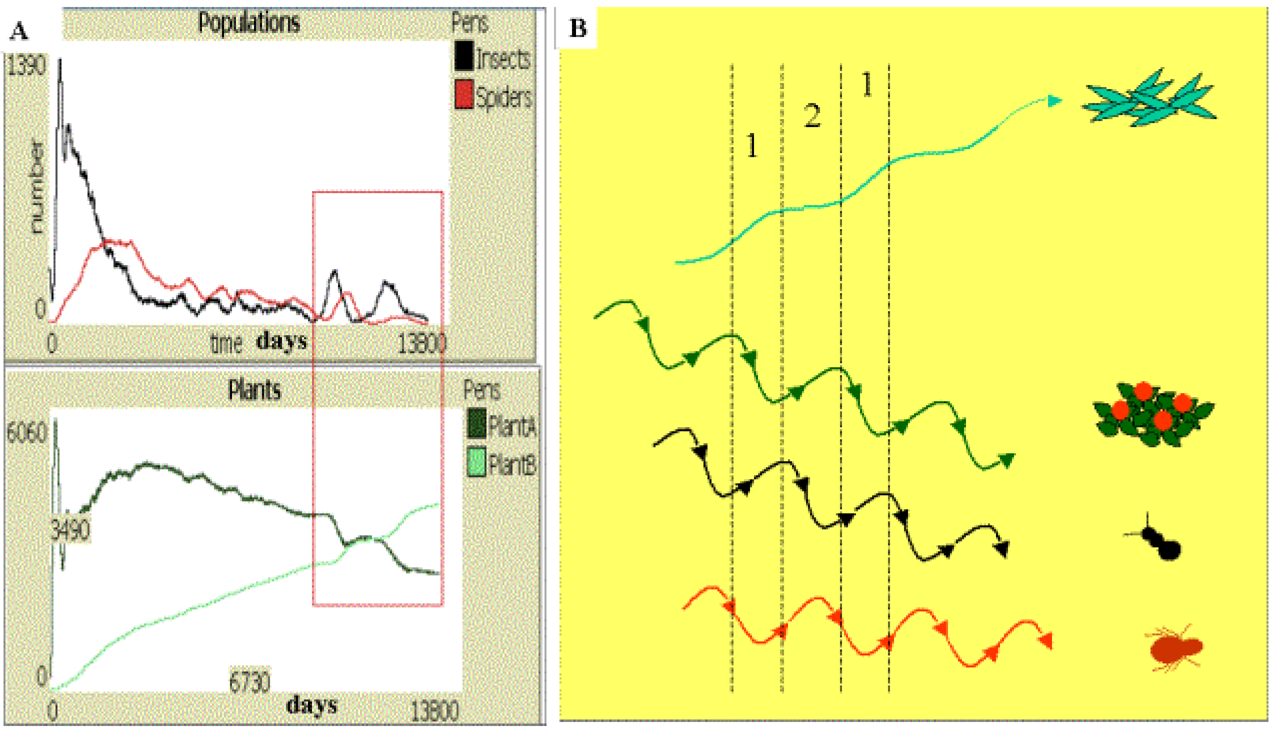


**FIGURE 3** Scenario exploration and network analysis show that the populations of spiders, insects and crop modules oscillate due to predator prey interactions. The red box in Figure 3A shows a scenario where this behavior is clearly manifested. Figure 3B schematically depicts how the periodic behaviors of all species are coupled.

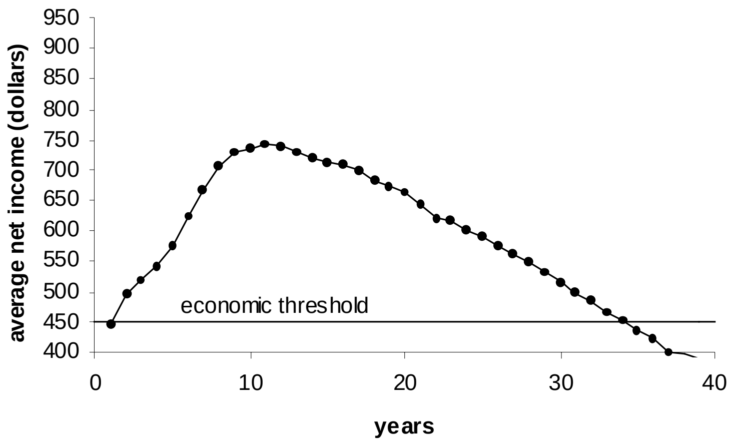


**FIGURE 4** Net income per strawberry field (5 year moving average) during 40 years in system with no insect/weed external control. The system first self organizes into a divers community and later into a monospecific weed stand. At year 34, the low crop population produces a net income below the economic threshold. The un-managed system is not sustainable.

During a workshop, MSc students explored with AGRODIVERSITY ver.2 the ecological features of the unmanaged system and were later challenged to develop an economic-viable sustainable system that would persist over a longer period of time. Students were asked first to try to reach their goal by only manipulating ecological parameters. The best scenarios approximated the hump-shaped unsustainable income curve. This exercise made students aware of the various interactions between the species and of the system's undesired equilibrium state. It made the case that economically viable diverse system are not necessarily in ecological equilibrium and require management for their continuation.

Students then explored the management options included in AGRODIVERSITY ver.2 A key condition for sustainability of the system is to limit weed growth. This can be done directly by weeding or indirectly by insecticide spraying. Interestingly, any of these practices needs to be carefully calibrated as they can produce paradoxical effects: too much weeding might produce a crop/insect boom that eventually affects the crop and allows weeds to expand even more. Similarly, too much spraying can kill the spiders, and eventually increase weeds.

Students explored the effects of both management options and their combinations. Most students concluded that performing regular weeding was the best management option for reaching a long-term productive system. They developed different strategies for the timing of management, using either a specific number of days or a specific number of weeds. The strength of role-playing games with agent-based models was confirmed during this workshop. Letting students take on the role of the farmer, intrigued them to fully understand the species interactions and the unexpected results of this complex dynamic system, as "their" strawberries and income were at stake (Figure 5). This assisted greatly in increasing their knowledge of some agro-diversity issues and of dynamical system behavior.

**FIGURE 5** Master's students explore scenarios in AGRODIVERSITY ver.2 and try to make all four species coexist in the benefit of the farmer, with and without management (Agroecology workshop, ECOSUR, 2006). We thank them for their enthusiastic collaboration.

After the workshop, we investigated the effects of insecticide-spraying strategies further to compare them to the best weeding management strategy (Figure 6). Weeding and spraying events were performed every time insects or weeds reached a specific number. Spraying effort was set to kill approximately 80% if the insects in the field in each spraying event. The income curve of the best insecticide strategy showed an initial high value but a steep downward trend. This strategy renders unsustainable as it finally drops below the economic threshold. In contrast, the initial income of the weeding strategy is low (at the threshold level), but it increases asymptotically to a high level that can be sustained.

These two distinct management strategies, weeding and insecticide spraying, present contrasting trade-offs between their short-term benefits and their long-term sustainability. Spraying insecticide is economically more favorable in the short term, but not on the long run. After an initial low income period, the weeding strategy attains a relatively high net average income fostered by functional coexistence of all four species.

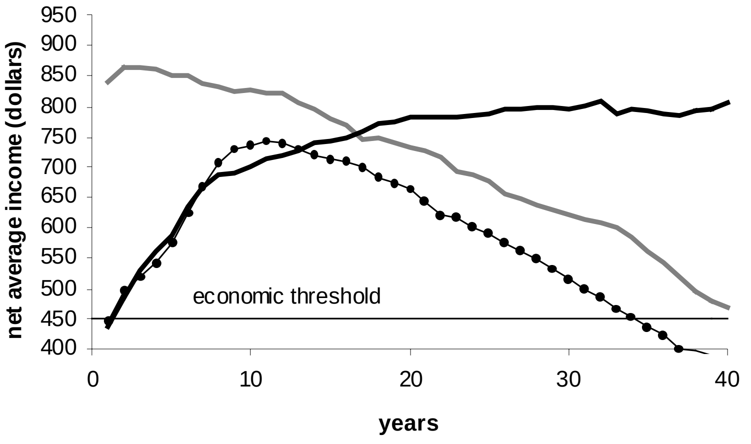


**FIGURE 6** Net income per strawberry field (5 year moving average) during 40 years in a system with no external insect/weed control, and with contrasting control strategies. The dotted curve refers to the system without management as shown in Figure 5. The black line consists of manually eliminating the weeds in direct contact with strawberries. The gray line consists of spraying insecticide.

The results of the "student-farmer" workshop and of our subsequent analysis motivated us to explore the potential of AGRODIVERSITY for developing a socio-ecological "strawberry-fields" game addressing issues of trade-offs, influences and conflicts between stakeholders. For this purpose we upgraded AGRODIVERSITY VER.2 to include the effects of decision-making processes by virtual farmers with neighboring fields.

Our virtual simple-minded farmers start off with one of the management strategies described above, but can learn from their neighbor's performance and change their management strategy at any time. The socio-economic context explains their initial strategies. The farmer practicing weeding (the WEEDER hereafter) is a farm-oriented producer. His monetary income is solely based on the sales of his strawberries. He is short of money and therefore he prefers a management strategy that allows him to use his own abundant family labor. His neighbor complements his income selling wage labor in the nearby city. He is in the opposite situation: he has more money, but lacks sufficient farm labor. Therefore he prefers to apply a more money-intensive strategy, namely to spray insecticide (The SPRAYER hereafter) (Figure 7).

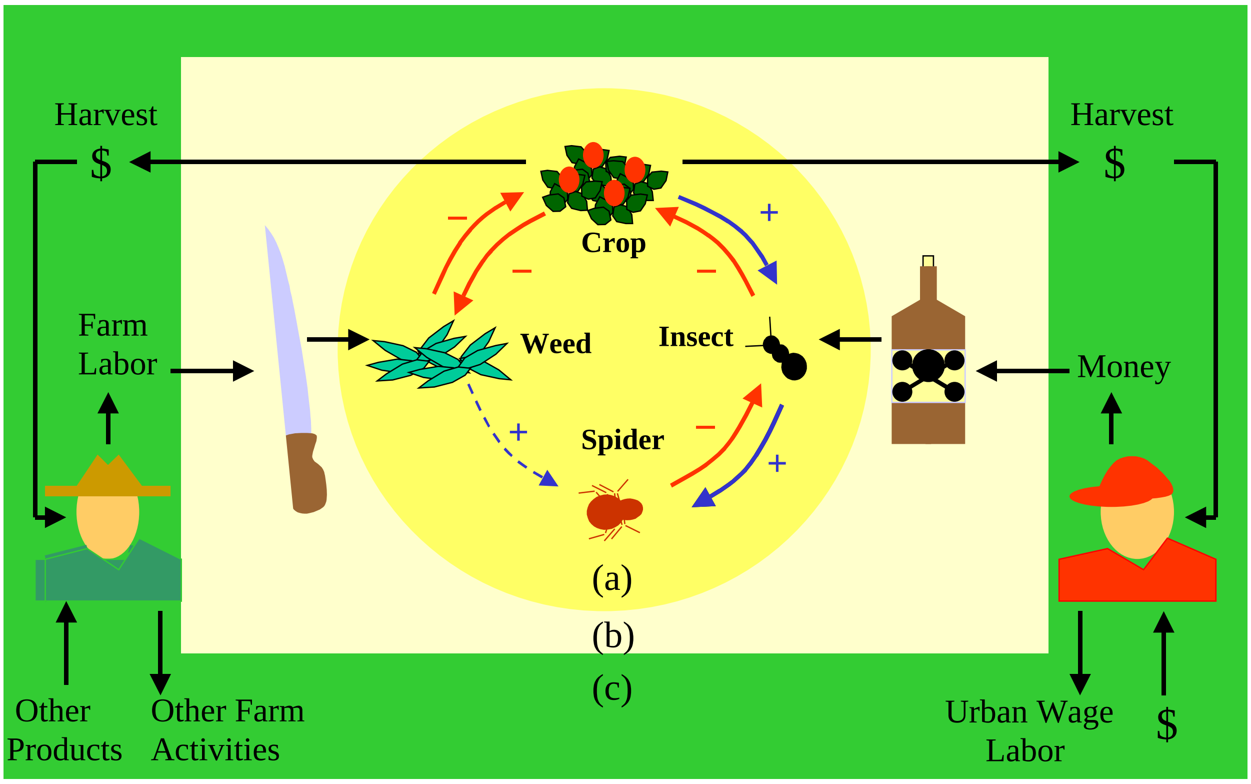


**FIGURE 7** A simple socio-ecological system for the strawberry fields game showing: (a) the ecological system components and interactions, (b) the pest management strategies and (c) the socio- economic drivers of pest management choices.

The upgraded version includes two strawberry fields (Figure 8). These can be parameterized with different levels of connectivity (border permeability to insects). Connectivity is the probability that an insect will cross to the other field when it reaches any border. When fields have 100% connectivity, they are assumed to be located at close distance from each other, whereas fields with 0% connectivity are assumed to be far apart. This feature allows us to explore how farmers' management strategies influence each other's fields and incomes.

Connectivity between fields with different management shows some surprising results (Figure 9). The WEEDER´s income is increased while the SPRAYER´s income is reduced by connectivity. Simply put, this happens because when the WEEDER´s insect population grows, individuals that migrate into the other field get sprayed at the expense of the SPRAYER while the WEEDER´s crop does better and reduces weed expansion.

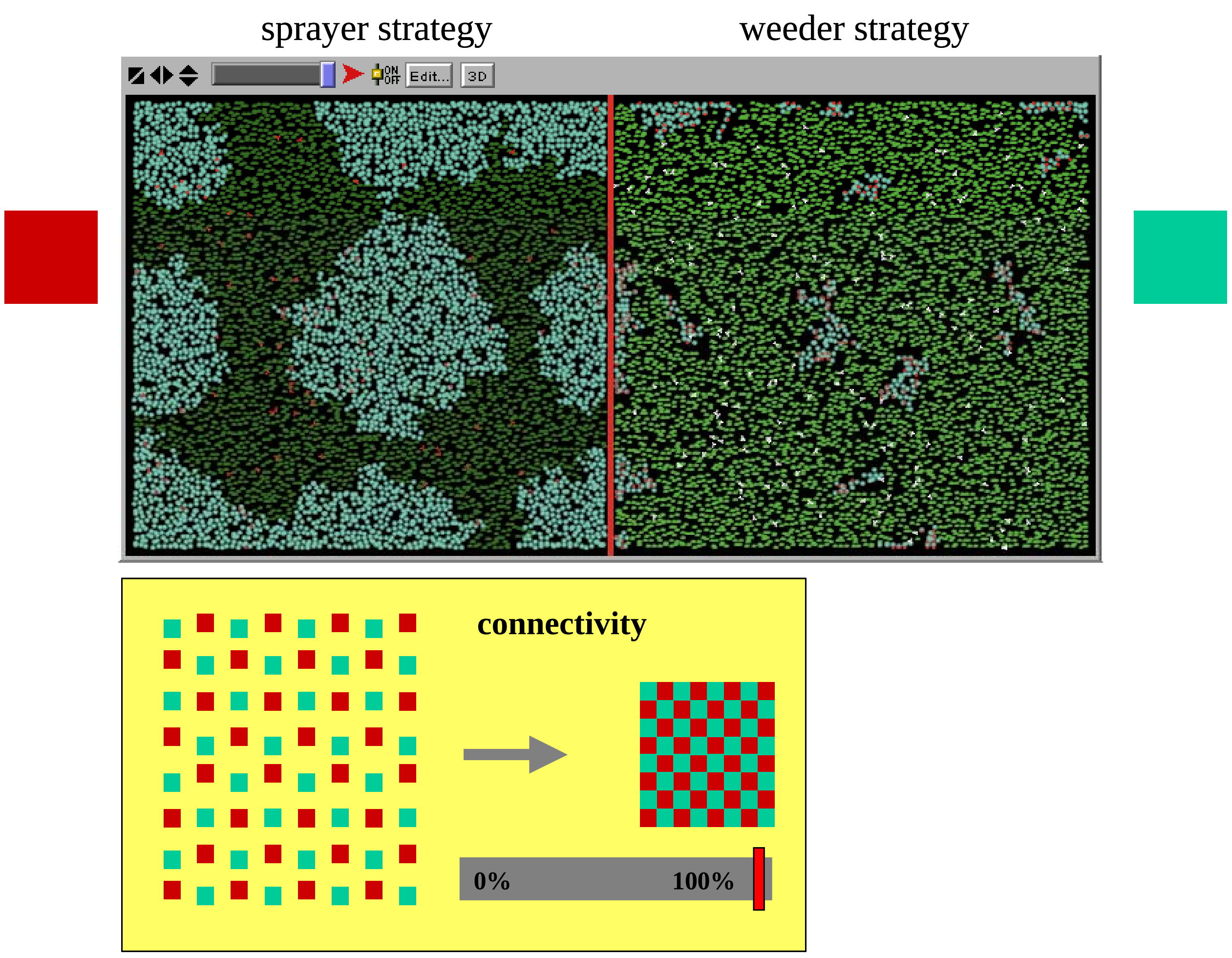


**FIGURE 8** The two-field version of AGRODIVERSITY (ver.2) for the strawberry field game. The left field is owned by a producer who sprays insecticide; the right by a producer who eliminates weeds manually. Field edges are wrapped around in a diagonal-fashion to produce a virtual checkerboard arrangement. The user decides the probability with which an insect can migrate into the other field when it reaches a border. Thus, the connectivity between fields ranges from 0 to 100%.

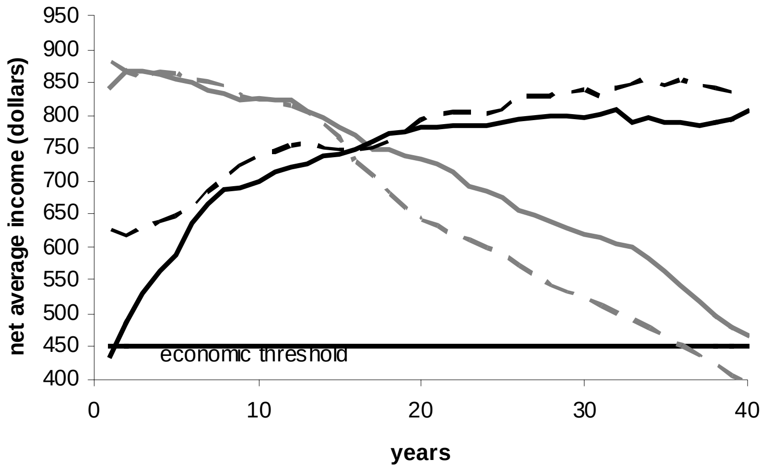


**FIGURE 9** Economic effects of connectivity between fields of fields with contrasting weed/ insect control strategies. Solid lines show income of fields with 0% connectivity and different management strategies (same as in figure 6). Dashed lines represent the effect of 100% connectivity.

We assumed that farmers, once aware of the effects of their own and their neighbor's management strategies, would learn from these experiences and would change their management. Farmers' decisions on management changes are based on two factors, namely the trend of their own income and the income level of their neighbor.

We distinguished five types of learners: (1) the "Quick-learner", (2) the "Semi-quick-learner", (3) the "Intermediate-learner", (4) the "Slow-learner" and the "Extremely-slow-learner". These different types of learners make their decisions to change their management in different stages of the systems. The "Quick-learner" will change his management as soon as he detects a downward trend in his income or sees that his neighbor's income is much higher than his own. The "Semi-quick-learner" makes the same analysis of his situation, but requires a bit more time to do so. The "Intermediate-learner" will not change his management until he realizes that his initially high income is going down and has reached the same level as that of his neighbor. This corresponds to the point in Figure 9 in which the income curves cross. When the farmer's income has a downward trend and has become lower than that of his neighbor the "Slow learner" will decide to change his management. The "Extremely-slow-learner" does not make this decision until his income drops near to the economic threshold.

In our example, five types of learners can be distinguished for farmers shifting from a spraying to weeding strategy. In the case of a weeding farmer changing to an insecticide management only two logical time steps are considered. We analyzed effects of such management changes through a number of simulations, thereby exploring possibilities for our strawberry game to address conflicts between stakeholders. In all cases 100% connectivity between fields is assumed.

We let the WEEDER shift to SPRAYER at two points in time. These farmers focus on the much higher short-term income of their insecticide-spraying neighbor. The management change in both early and semi-early stage produces an unfavorable result for the weeding farmer (Figure 10). His new 5-year average income shows a hump shape, whereas the income that he could have obtained if he would not have changed his management shows an asymptotically high income. The neighboring field where insecticide was being sprayed since the beginning benefits from the WEEDER´s management change of his neighbor.

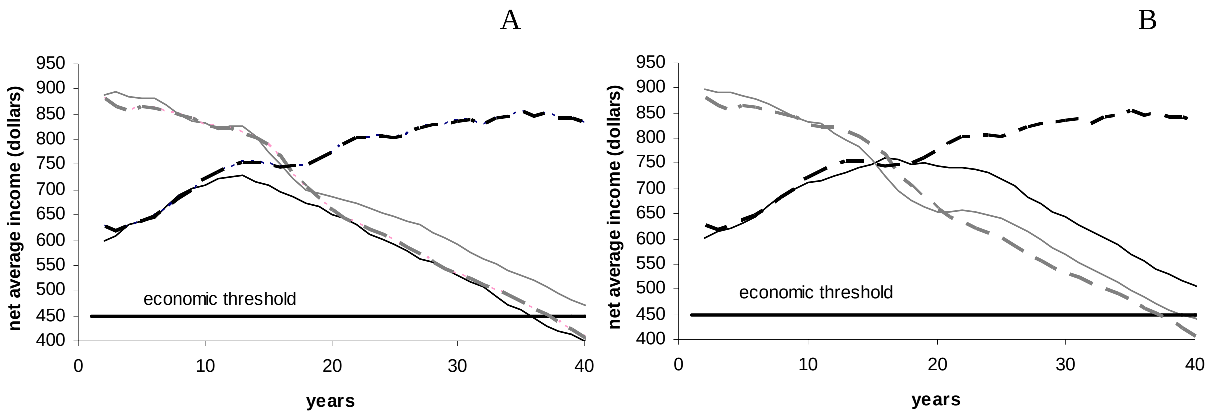


**FIGURE 10** The WEEDER decides to imitate his neighbor and shifts to spraying. This affects both producers' economic outcome. Such a decision can be made at an early stage (a) or at an intermediate stage (b). Black and gray dashed lines, as in Figure 9. Black and gray solid thin lines represent the effect of management change on the corresponding fields.

We allowed the SPRAYER to shift to WEEDER at five different stages. Figure 11 shows the effects of an early (a, b), intermediate (c, d) and extremely-late (e, f) shift.

An early shift produces a rapid drop in income, but the system can recover and reach a high persistent level similar to the WEEDER's long-term income. This behavior implies a transition cost for this type management change. If the shift is delayed, the income level cannot reach the WEEDER´s maximum income level. At all stages an initial fall was observed, followed by an increasing income trend. In some cases, there was no recovery after the initial fall due to particular conditions in the abundance of spiders and weeds at the time of management change (Figure 11- b, d, f). This implies a risk associated to this change in strategy. The neighbor's income was negatively affected by the SPRAYER´s decision to shift. As in the case WEEDER shifting to SPRAYER, this implies a potential conflict between stakeholders.

In all two- field explorations discussed here, connectivity effects and management changes that benefit one farmer's income have negative consequences on the other.

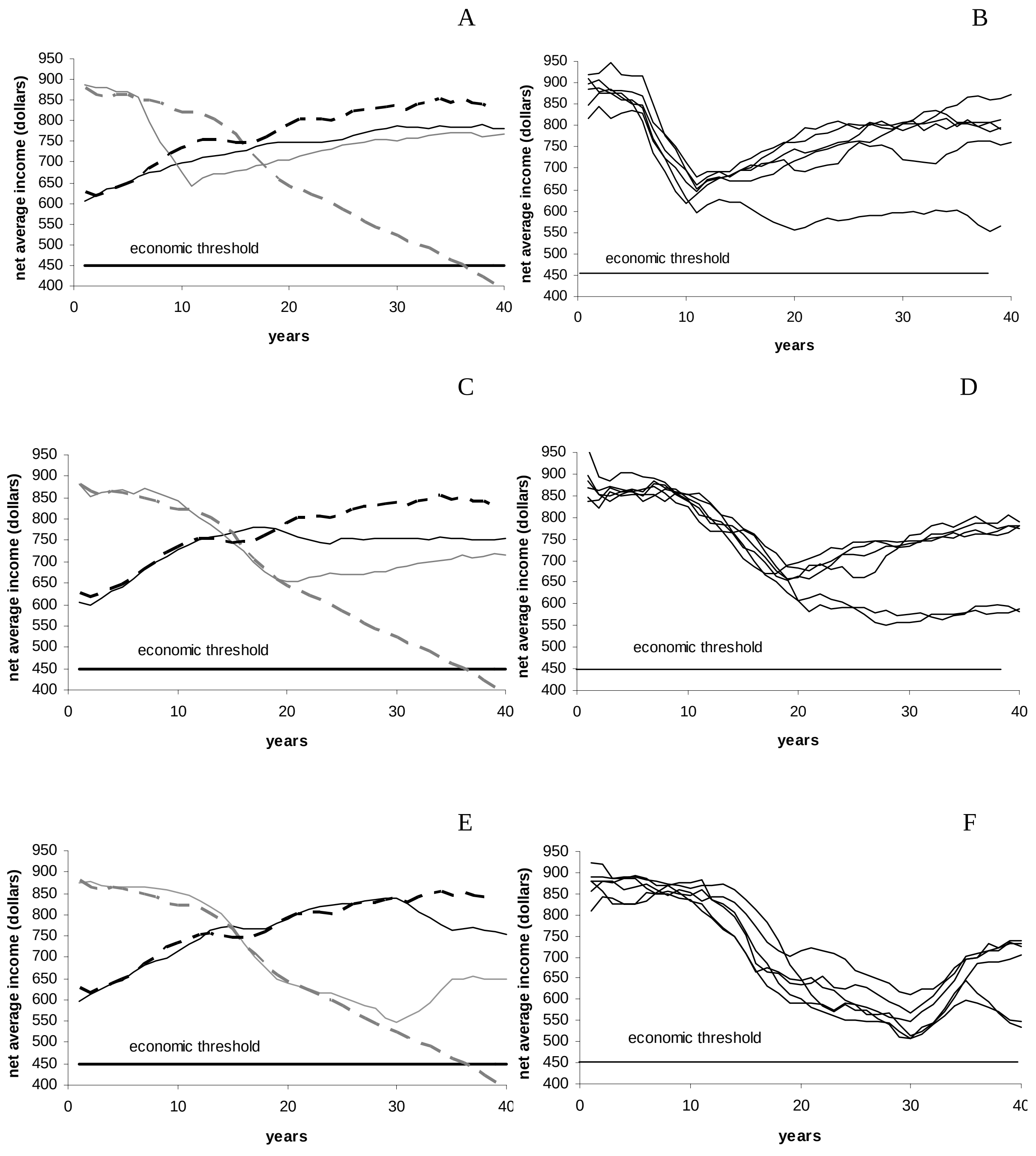


**FIGURE 11** The SPRAYER decides to imitate his neighbor and shifts to weeding. Here we present the effect of shifting at an early stage (a), an intermediate stage (c) or at an extremely-late stage (e). Black and gray dashed lines, as in Figure 9. Solid black and gray thin lines in a, c and e represent the effect of management change on the corresponding fields. Figures b, d and f show a few individual model runs from which the thin gray average curve on the corresponding left-side figure was obtained. The latter show outlier curves corresponding to cases with no or very slow recovery after management change.

## CONCLUSIONS

Spatially explicit agent based models, and user-friendly interfaces such as provided by Netlogo, proved to be an excellent tool to motivate students and other stakeholders while learning about complex socio-ecological systems.

The ecological network in AGRODIVERSITY ver. 2 is simple enough to allow users to keep track of species behaviors and interactions. The user can quickly gain familiarity with such behaviors and eventually understand the underlying network properties and the effects of direct and indirect nonlinear ecological interactions in this simple agricultural system.

The biodiverse ecological system in AGRODIVERSITY can be highly productive but cannot sustain itself; it requires external insect/weed control to persist. This allows users to explore and understand the effects of  management and livelihood strategies, which are the higher-level social drivers of the system.

Both money-supported and labor-supported management strategies in the model showed a tradeoff between short-term and long-term benefits. The first strategy was very profitable in the short term but unsustainable in the medium term. The second strategy is in the opposite case. This allows users to be introduced to tradeoff analysis and decision making in a lively way.

Connectivity between farmers with nearby fields managed in contrasting ways produced negative externalities; Strategy imitation was favorable for one farmer while unfavorable for the other. Both situations imply a potential conflict, and thus for conflict resolution.

All these features offer a number of opportunities for designing a simple but powerful agent-based model-supported multi-stakeholder game. Such a game could contribute to help users address and become more familiar with sustainability issues in agrodiverse systems.

## REFERENCES

Altieri, M.A., 1999, “Applying Agroecology to Enhance the Productivity of Peasant Farming Systems in Latin America,” *Environment, Development and Sustainability* **1**:197-217.

Barreteau, O., 2003, “The joint use of role-playing games and models regarding negotiation processes: characterization of associations,” *Journal of Artificial Societies and Social Simulation*; available at http://jasss.soc.surrey.ac.uk/6/2/3.html

Barreteau, O., C. Le Page, and P. D`Aquino, 2003, **“**Role-Playing Games, Models and Negotiation Processes,” *Journal of Artificial Societies and Social Simulation*; available at  http://jasss.soc.surrey.ac.uk/6/2/10.html

Berger, T., P. Schreinemachers, and J. Woelcke, 2006, “Multi-agent simulation for the targeting of development policies in less-favored areas,” *Agricultural Systems* **88**: 28-43.

Chambers, R., 1993, *Challenging the Professions: Frontiers for Rural Development*, London: Intermediate Technology Publications.

García-Barrios, L., and R. García-Barrios, 1992, “La modernización de la pobreza. Dinámica del cambio técnico entre los campesinos temporaleros de México,” *Revista Estudios Sociológicos* **10** (29): 263-288.

García-Barrios, L., and S. Chakravarty, 2002, “COFFEESCAPE. An agent-based model for simulating interactions among coffee producers in a troubled market place,” unpublished software, developed during the Santa Fe Complex System Summer School, 2002.

García-Barrios, L.E., and M. Pimm, 2006, “Negotiated design of sustainable production systems among social agents with conflicting interests. Interactive play in three acts with model simulation exercises,” in: Masera, O. and Y. Galván- Miyoshi, (Eds) (in press), *Avances teóricos y metodológicos en la evaluación de sustentabilidad: la experiencia y los nuevos retos en la aplicación del marco MESMIS,* MundiPrensa, Mexico; executable program available at www.ecosur.mx/sustentabilidad (Spanish and English version).

García-Barrios, L.E. and E.N. Speelman, 2006, “AGRODIVERSITY ver.2,” CD-rom, El Colegio de la Frontera Sur, San Cristobál de las Casas, Chiapas, Mexico.

Goodman, M., and L. García-Barrios, 2004, “Assesment of biological effects in agriculture in Mexico,” in *Maize & Biodiversity. The effects of transgenic maize in Mexico*; available at http://www.cec.org/maize/index.cfm.

Rosset, P., 2001, *Genetic engineering of food crops for the Third World: an appropriate response to poverty, hunger and lagging productivity?*, Food First/ Institute for Food and development policy, CA, USA.

Speelman, E.N., M. Astier, S. López-Ridaura, P.A. Leffelaar, and M.K. van Ittersum, 2006, "Trade-off Analysis for Sustainability Evaluation; a case study for Purhepecha region, Mexico," *Outlook on Agriculture* **35** (1): 57-64.

Speelman, E.N., M. Astier, and Y. Galván-Miyoshi, (in press), "Sistematización y análisis de los estudios de caso MESMIS: lecciones para el futuro," in Masera, O. and Y. Galván-Miyoshi, (Eds.), *Avances teóricos y metodológicos en la evaluación de sustentabilidad: la experiencia y los nuevos retos en la aplicación del marco MESMIS*, MundiPrensa, Mexico.

Swift, M.J., A.-M.N. Izac, and M. van Noordwijk, 2004, "Biodiversity and ecosystem services in agricultural landscapes-are we asking the right question?" *Agriculture, Ecosystems and Environment* **104**: 113-134.

Vandermeer, J., and I. Perfecto, 2005, "The Future of Farming and Conservation," *Science* **308**: 1257-1258.

Wilensky, U., 1999, "NetLogo," Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL; available at http://ccl.northwestern.edu/netlogo/.