Tutorial on Agent-based Modeling and Simulation

Conference Paper *in* Proceedings - Winter Simulation Conference · January 2005

DOI: 10.1109/WSC.2005.1574234 · Source: DBLP

2 authors:

C. M. Macal
Argonne National Laboratory
159 PUBLICATIONS **4,298** CITATIONS
SEE PROFILE

Michael John North
Argonne National Laboratory
85 PUBLICATIONS **4,211** CITATIONS
SEE PROFILE

Proceedings of the 2009 Winter Simulation Conference
M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin and R. G. Ingalls, eds.

AGENT-BASED MODELING AND SIMULATION

Charles M. Macal

Center for Complex Adaptive Systems Simulation
Decision & Information Sciences Division
Argonne National Laboratory
Argonne, IL 60439 USA

Michael J. North

Center for Complex Adaptive Systems Simulation (CAS^2)
Decision & Information Sciences Division
Argonne National Laboratory
Argonne, IL 60439 USA

ABSTRACT

Agent-based modeling and simulation (ABMS) is a new approach to modeling systems comprised of autonomous, interacting agents. Computational advances have made possible a growing number of agent-based models across a variety of application domains. Applications range from modeling agent behavior in the stock market, supply chains, and consumer markets, to predicting the spread of epidemics, mitigating the threat of bio-warfare, and understanding the factors that may be responsible for the fall of ancient civilizations. Such progress suggests the potential of ABMS to have far-reaching effects on the way that businesses use computers to support decision-making and researchers use agent-based models as electronic laboratories. Some contend that ABMS "is a third way of doing science" and could augment traditional deductive and inductive reasoning as discovery methods. This brief tutorial introduces agent-based modeling by describing the foundations of ABMS, discussing some illustrative applications, and addressing toolkits and methods for developing agent-based models.

1 INTRODUCTION

Agent-based modeling and simulation (ABMS) is a new modeling approach that has gained increasing attention over the past 10 years. This growth trend is evidenced by the increasing numbers of articles appearing in modeling and applications journals, the number of funded programs that call for agent-based models that incorporate elements of human and social behavior, the growing number of conferences on or that have tracks dedicated to agent-based modeling, the demand for ABMS courses and instructional programs, and the number of presentations at conferences such as the WSC that reference agent-based modeling. Some contend that ABMS "is a third way of doing science" and could augment traditional deductive and inductive reasoning as discovery methods (Axelrod 1997). This tutorial provides a necessarily brief introduction to agent-based modeling and simulation. The goals are to show that ABMS is:

- Useful: Why ABMS is an appropriate modeling approach for a large class of problems and has advantages over conventional modeling approaches in many cases,
- Usable: How ABMS is advancing to the point of producing portable, extensible, and transferable software, with better integrated development environments and more examples of good applications, and
- Used: How ABMS is being used to solve practical problems.

This tutorial is organized into two parts. The first part (Section 2-3) is a tutorial on how to *think* about ABMS. The background on ABMS and its motivating principles are described along with some exemplary applications. The second part (beginning with Section 4) is a tutorial on how to *do* ABMS. It addresses modeling approaches and toolkits for developing agent-based models.

2 HOW TO THINK ABOUT ABMS

2.1 The Need for Agent Based Modeling

Why is agent-based modeling becoming widespread? The answer is because we live in an increasingly complex world. First, the systems that we need to analyze and model are becoming more complex in terms of their interdependencies. Conventional modeling tools may not be as applicable as they once were. An example application area is the deregulation of the formerly centralized electric power industry in which agents are suddenly free to make pricing and investment choices based on their individual criteria. Second, some systems have always been too complex for us to adequately model. Modeling economic markets has traditionally relied on the notions of perfect markets, homogeneous agents, and long-run equilibrium because these assumptions made the problems analytically and computationally tractable. We are beginning to be able to relax some of these assumptions and take a more realistic view of these economic systems through ABMS. Third, data are being collected and organized into databases at finer levels of granularity. Micro-data can now support individual-based simulations. And fourth, but most importantly, computational power is advancing rapidly. We can now compute large-scale microsimulation models that would not have been plausible just a few years ago.

2.2 What Is an Agent

There is no universal agreement on the precise definition of the term "agent" in the context of ABMS. It is the subject of much discussion and occasional debate. The issue is more than an academic one, as it often surfaces when one makes a claim that their model is "agent-based" or when one is trying to discern whether such claims made by others have validity. There are important implications of the term "agent-based" when used to describe a model in terms of the model's capabilities or potential capabilities that could be attained through relatively minor modification. A formal definition of "agent" is beyond the scope of this paper; in the literature informal descriptions of "agent" tend to agree on more points than they disagree.

Unlike particle systems (idealized gas particles for example) which are the subject of the field of physical systems simulation, agents as used are are diverse, heterogeneous, and dynamic in their attributes and behavioral rules, as shown in Figure 1. Some modelers consider any type of *independent* component whether it be software or a model to be an agent (Bonabeau 2001). An independent component's behavior can range from simple in nature, e.g., described by simple if-then rules, to the complex, e.g., described by complex behavioral models from the fields of cognitive science or artificial intelligence. Some authors insist that a component's behavior must also be *adaptive* in order for it to be considered an agent. In this view, the agent label is reserved for components that can learn from their environment and dynamically change their behaviors in response to their experiences. Casti (1997) argues that agents should contain both base-level rules for behavior as well as a higher-level set of "rules to change the rules." The base-level rules provide responses to the environment, while the "rules to change the rules" provide adaptation. Jennings (2000) provides a computer science view of "agent" that emphasizes the essential characteristic of *autonomous* behavior. This requires agents to be active responders and planners rather than purely passive components.

For practical modeling purposes, we consider agents to have certain properties and attributes:

- An agent is *autonomous* and self-directed. An agent can function independently in its environment and in its interactions with other agents, generally from a limited range of situations that are of interest. We refer to an agent's behavior as the representation of a process that links the agent's sensing of its environment to its decisions and actions.
- Agents are modular or *self-contained*. An agent is an identifiable, discrete individual with a set of characteristics or attributes, behaviors, and decision-making capability. The discreteness requirement implies that an agent has a boundary in a sense and one can easily determine whether something (that is, an element of the model's state) is part of an agent, is not part of an agent, or is a characteristic shared among agents.
- An agent is social, *interacting* with other agents. Agents have protocols or mechanisms that describe how they interact with other agents, just as an agent has behaviors. Common agent interaction protocols include contention for space and collision avoidance; agent recognition; communication and information exchange; influence; and other domain-or application-specific mechanisms.

Agents often have additional properties, which may or may not be considered as defining properties or necessary for agency.

- An agent may live in an *environment*. Agents interact with their environment as well as with other agents. An agent is *situated*, in the sense that its behavior is situationally dependent, which means that its behavior is based on the current state of its interactions with other agents and with the environment.

- An agent may have explicit *goals* that drive its behavior. The goals are not necessarily objectives to maximize as much as criteria against which to assess the effectiveness of its decision and actions. This allows an agent to continuously compare the outcomes of its behaviors to its goals and gives it a benchmark for possibly modifying it behavior.
- An agent may have the ability to *learn and adapt* its behaviors based on its experiences. Individual learning and adaptation requires an agent to have memory, usually in the form of a dynamic agent attribute. (We contrast *individual adaptation* with *population adaptation*. In population adaptation, the proportion of individuals within the population with certain attributes that better suit them to their environment increases over time. The individuals do not necessarily change their behavior or adapt.)
- Agents often have *resource attributes* that indicate its current stock of one or more resources, e.g., energy, wealth, information, etc.

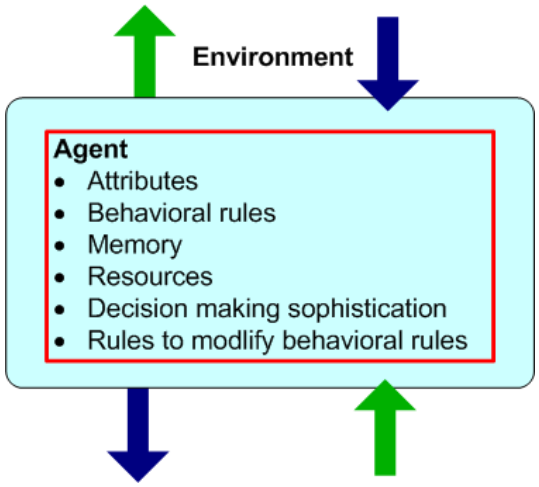


Figure 1: A typical agent

An agent's behavioral rules can vary in their sophistication, how much information is considered in the agent's decision (this is referred to as cognitive load), the agent's internal models of the external world including the possible reactions or behaviors of other agents, and the extent of its memory of past events that an agent retains and uses in its decisions. Often, the agents in a model will lack adaptation because it is not to achieve the model's intended purpose. For example, in a supply chain model it may not be necessary to model agent adaptation if the model's purpose is to evaluate a set of specific inventory management rules.

2.3 Agent-based Modeling and Simulation

Agent-based modeling is known by many names. ABM (agent-based modeling), ABS (agent-based systems or simulation), and IBM (individual-based modeling) are all widely-used acronyms, but "ABMS" will be used throughout this discussion. The term "agent" has connotations in realms other than agent-based modeling as well. ABMS agents are different from the agents typically found in mobile agent systems. "Mobile agents" are light-weight software proxies that roam over the world-wide web and perform various functions for users and to some extent can behave autonomously.

Another point of clarification concerns the term "simulation." Agent-based simulation refers to a model in which the dynamic processes of agent interaction are simulated repeatedly over time, as in systems dynamics, time-stepped, discrete-event, and other types of simulation. An agent-based model, more generally, is a model in which agents repeatedly interact. For example, when agents optimize their collective behavior through simple exchanges of information as is done in ant colony optimization or in particle swarm optimization, the purpose is to achieve a desired end-state, i.e., the optimized system, rather than to simulate a dynamic process for its own sake.

2.4 Background on ABMS

ABMS has connections to many other fields including complexity science, systems science, systems dynamics, computer science, management science, several branches of the social sciences, and conventional modeling and simulation. ABMS

draws on these fields for its theoretical foundations, its conceptual world view and philosophy, and for applicable modeling techniques. ABMS is related to the fields of multi-agent systems (MAS) and robotics from the field of artificial intelligence (AI), as well as Artificial Life (ALife). But ABMS is not only tied to understanding and designing "artificial" agents. Its most common use is in modeling human social and organizational behavior and individual decision-making (Bonabeau 2001). With this, comes the need to represent behaviors, social interaction, collaboration, group behavior, and its possible emergence.

ABMS has its direct historical roots in complex adaptive systems (CAS) and the underlying notion that "systems are built from the ground-up. CAS concerns itself with the question of how complex behaviors arise in nature among myopic, autonomous agents. In addition, ABMS tends to be descriptive, with the intent of modeling the actual or plausible behavior of individuals, rather than normative such as traditional operations research (OR), which seeks to optimize and identify optimal behaviors.

The field of CAS was originally motivated by investigations into *adaptation* and *emergence* of biological systems. CAS have the ability to self-organize and dynamically reorganize their components in ways better suited to survive and excel in their environments, and this adaptive ability occurs, remarkably, over an enormous range of scales. John Holland, a pioneer in the field, identifies properties and mechanisms common to all CAS (Holland 1995) such as (1) Aggregation: allows groups to form, (2) Nonlinearity: invalidates simple extrapolation, (3) Flows: allow the transfer and transformation of resources and information, and (4) Diversity: allows agents to behave differently from one another and often leads to the system property of robustness. CAS mechanisms are: (1) Tagging: allows agents to be named and recognized, (2) Internal models: allows agents to reason about their worlds, and (3) Building blocks: allows components and whole systems to be composed of many levels of simpler components. These CAS properties and mechanisms provide a useful reference for designing agent-based models. Essentially, one models a complex adaptive system by developing an agent-based model.

Social agent-based modeling, modeling social processes starting at the individual level, has been around since at least 1970's with Sakoda's publication of the checkerboard model of social interaction (Sakoda 1971). This model was essentially a cellular automata model (cellular automata are discussed in Section 4.2). More recently, Epstein and Axtell (1996) introduced the idea of artificial societies in their *SugarScape* model. *SugarScape* uses agent-based modeling to represent an entire society "from the ground up" by modeling its individuals and their interactions. Epstein and Axtell showed how an extensive number of social processes could be credibly modeled including life, death, disease, war, reproduction, and wealth. This seminal work and it continues to offer a blueprint for many agent-based models of social processes.

2.5 ABMS Demonstrations on Order Creation

2.5.1 Demonstration: *Life*

We begin with a simple "game" developed by the mathematician John Conway, called *Life* (Gardner 1970). *Life* is based on cellular automata (CA). Perhaps the simplest way to illustrate the basic ideas of agent-based modeling and simulation is through CA. According to Casti (1997), the original notion of CA was developed by the physicist Stanislaw Ulam in response to a question posed by the famous 20th century mathematician John von Neumann. The question was, "could a machine be programmed to make a copy of itself?" In effect, the question had to do with whether it was possible to develop a logical structure that was complex enough to completely contain all of the instructions for replication. Von Neumann answered his own question by developing the abstract mathematical representation of a machine in the form of a cellular automata.

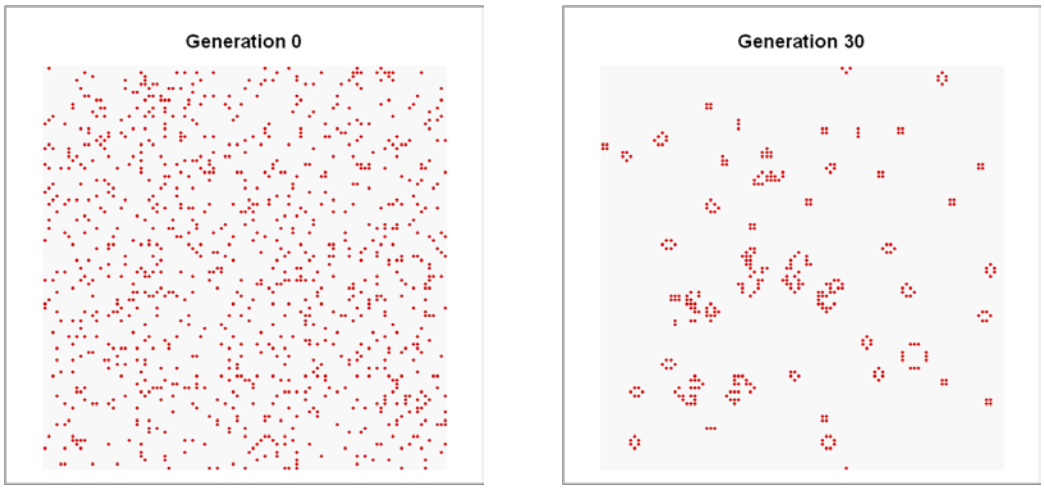
A typical CA is a two-dimensional grid or lattice partitioned into cells. Each cell assumes one of a finite number of states at any point in time. A set of simple rules determines the value of each cell based on the cell's previous state. Every cell is updated each period according to the rules, as in a time-stepped simulation. The value of a cell in the next period depends on the cell's current value and the values of its immediate neighbors in the eight surrounding cells (assuming a "9-cell" Moore neighborhood). Each cell is identical in terms of its update rules.

A CA is deterministic in that the same state for a cell and its neighbors always results in the same updated state. *Life* has three rules that determine the next state (either On or Off) of each cell:

Rule 1: The cell will be On in the next generation if exactly three of its eight neighboring cells are currently On.
Rule 2: The cell will retain its current state if exactly two of its neighbors are On.
Rule 3: The cell will be Off otherwise.

Interpreting *Life* as an agent-based model, we can consider each cell to be an agent. The cell update rules represent an agent's "behavior." The states of a cell (On or Off) are the possible agent states. The states of all the agents taken together at a specific time in the simulation is the state of the model (system). The environment in *Life* is the grid upon which the agents live. The function of the environment is minimal in *Life*. It merely serves as a reference point for determining an agent's neighborhood, which consists of the cells immediately adjacent to it.

Figure 2 shows snapshots from a *Life* simulation. Figure 2a shows an initial random distribution of On cells. The initial random distribution of On cells is the only stochastic element of *Life*. After several updates of all cells in the grid in which the update rules are applied to every cell, distinctive patterns emerge, and in some cases these patterns can sustain themselves indefinitely throughout the simulation (Figure 2b). The nine-cell neighborhood assumption built into *Life* determines the scope of agent interaction and the locally available information for each cell to update its state.



(a) (b)

Figure 2: *Life* simulation: (a) initial random layout of cells in the On state, (b) after all cells updated 30 times

Several observations are important about *Life*: the rules are simple and the rules use only local information. By local information we mean that the state of each cell is based only on the current state of the cell and the cells touching it in its immediate neighborhood. Repeated simulations of *Life* also reveal that the resulting patterns are very sensitive to the initial conditions. Each initial distribution of On cells results in an entirely different pattern of On and Off cells in the long-run. Even changing single cells in the initial distribution of On cells results in vastly different *Life* outcomes.

These are interesting findings, and observing the patterns created by repeated simulations of Life reveals a world of virtually endless creations. Yet the point is that we have a simple system comprised of simple rules that begins in a random state and that yields patterns, order or structure as the case may be. The patterns can be said to "emerge." Another example, the *Boids* Model, yields similar emergent behavior.

2.5.2 Demonstration: *Boids*

The *Boids* simulation is a good example of how interacting agents, characterized by simple behavioral rules, lead to emergent and seemingly organized behavior (Reynolds 2006). Agent behavior is reminiscent of schooling or flocking behavior in fish or birds. In the *Boids* model, each agent has three rules governing its movement:

Rule 1: Cohesion: each agent steers toward the average position of its nearby "flockmates"
Rule 2: Separation: each agent steers to avoid crowding local flockmates
Rule 3: Alignment: each agent steers towards the average heading of local flockmates

Here, nearby or local refers to agents in the immediate neighborhood of an agent as defined by the straight-line distance. A fourth rule is added to the above three rules for the purposes of the demonstration to ensure that the agents stay within a specified area. Initially, a set number of boids are randomly assigned positions and orientations (Figure 3a). With only these simple rules applied at the individual agent level and only to the agents in its neighborhood, agent movements begin to appear coordinated and purposeful. A leaderless flock emerges (Figure 3b). Three observations can be made about the *Boids* Model: (1) the rules are simple, and (2) the rules use only local information, and (3) repeated experiments (not shown here)

demonstrate that the patterns that develop such as group formation and clustering can be extremely sensitive to the initial conditions - in this case, the initial random positions and orientations of the boids.

(a) (b)

Figure 3: (a) *Boids* simulation, initial random configuration (b) *Boids* simulation, after 500 updates showing two apparent clusters of agents

2.6 Simple Rules Result in Emergent Organization and Complex Behaviors

We can make an observation from both *Life* and *Boids*. And that is that sustainable patterns can emerge in systems that are completely described by simple, deterministic rules based on only local information. These observations have practical implications for developing and interpreting agent-based models, as more complex models of the kind that people are likely to build to represent real-world phenomenon would also exhibit emergent behavior resulting from agent interaction.

In the natural world the implications go far beyond this simple kind of emergence illustrated by *Life* and *Boids*. Based on simple rules of behavior and the nature of agent interactions, natural systems seemingly exhibit collective intelligence, or *swarm intelligence*, even without the existence of or the direction provided by a central authority. How is it that an ant colony can organize itself to carry out the complex tasks of food gathering and nest building and at the same time exhibit an enormous degree of resilience if the colony is seriously disrupted? Natural systems are able to not only survive, but also to adapt and become better suited to their environment, effectively optimizing their behavior over time. Swarm intelligence has inspired agent-based modeling as well as practical optimization techniques, such as ant colony optimization and particle swarm optimization that have been used to solve practical scheduling and routing problems (Bonabeau, Dorigo and Theraulaz, 1999). These types of algorithms can be implemented in agent-based modeling.

3 AGENT-BASED MODELING APPLICATIONS

Agent-based modeling is being applied to many areas, spanning human social, physical and biological systems. Applications range from modeling ancient civilizations that have been gone for hundreds of years, to designing new markets for products that do not exist right now. Selected applications areas are listed in Table 1, with an exemplar publication for each area. All of the cited publications make the case for agent-based modeling as the preferred modeling approach versus other modeling techniques for the particular application domain and the problem addressed. In a nutshell, they argue that agent-based modeling is used because only agent-based models can explicitly incorporate the complexity arising from individual behaviors and interactions that exist in the real-world.

For example, Griffin and Stanish (2007) developed an agent-based model, using the Repast agent-based modeling toolkit, for the Lake Titicaca basin of Peru and Bolivia covering the late prehistoric period, 2500 BC to AD 1000. The model was used to study hypotheses for the causal variables affecting prehistoric settlement patterns and political consolidations. The model's geo-spatial structure consists of a 50,000 square km grid composed of 1.5 km square cells. Each cell models its geography, hydrology, and agricultural potential. Agents consist of settlements, peoples, political entities, and leaders that interact with each other and the environment. Agent behavior is modeled as a set of condition-action rules that are based on hypothesized causal factors affecting agricultural production, migration, competition, and trade. The authors report that through

a series of simulation runs, the model produced a range of alternative political pre-histories and the emergence of macro-level patterns that corresponded to observed patterns in the archaeological record.

Charania, et al. (2006) uses agent-based simulation to model possible futures for a market in sub-orbital space tourism. Each agent represents an entity within the space industry, such as consumers, producers, the government, etc., that provides or demands different products and services. Tourism companies seek to maximize profits while they compete with other companies for sales. Individual companies decide the price they will charge for a flight aboard their vehicle. Customers evaluate the products offered by the companies according to their tastes and preferences.

Mock and Testa (2007) develop an agent-based model of predator-prey relationships between transient killer whales and threatened marine mammal species (sea lions and sea otters) in Alaska. The authors state that until now only simplistic, static models of killer whale consumption had been constructed due to the fact that the interactions between transient killer whales and their marine mammal prey are poorly suited to classical predator-prey modeling approaches. Their agent-based killer whale model includes both individuals and hunting groups. Individual agents eat, grow, reproduce, and die. Hunting groups change in size and composition while encountering other marine mammals.

Typically, in any sampling of agent-based models, applications range across a continuum, from small, elegant, minimalist models to large-scale decision support systems. Minimalist models are based on a set of idealized assumptions, and are designed to capture only the most salient features of a system. These models are exploratory electronic laboratories in which alternative assumptions can be varied over many simulations. Decision support models tend to be large-scale applications, designed to answer a broad range of real-world policy questions. These models are distinguished by including real data and having passed some level of validation testing to establish credibility.

There are many more agent-based modeling publications in each of the applications areas listed in Table 1, and there are many more application areas than the table references to which agent-based modeling is being successfully applied.

4 HOW TO DO ABMS

4.1 Modeling Agent Processes

Identifying agents, accurately specifying their behaviors, and appropriately representing agent interactions are the keys to developing useful agent models. One begins developing an agent-based model by identifying the agent types (classes) along with their attributes. Agents are generally the decision-makers in a system whether they be human, organizational, or automated.

Once the agents are defined, agent behaviors are specified. One needs to have a theory of agent behavior as a basis for modeling agent behavior. For example, a normative model in which agents attempt to optimize a well-defined objective can be a useful starting point to eventually developing more descriptive and domain-specific behavioral heuristics. Alternatively, one may begin with a generic behavioral heuristic, such as anchoring and adjustment, to describe agent behavior or more broadly a formal behavioral modeling framework such as BDI (Belief-Desire-Intent) or others (Rao and Georgeff 1991).

In addition to agents, an agent-based model consists of agent relationships, discussed below. One then adds the methods that control which agents interact, when they interact, and how they interact.

Table 1: A Sample of Recent Agent-based Applications

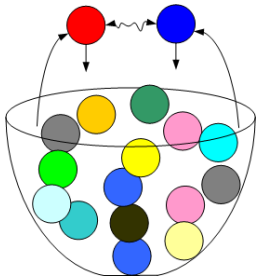
Application Area	Model Description
Air Traffic Control	Agent-based model of air traffic control to analyze control policies and performance of an air traffic management facility (Conway 2006)
Anthropology	Agent-based model of prehistoric settlement patterns and political consolidation in the Lake Titicaca basin of Peru and Bolivia (Griffin and Stanish 2007)
Biomedical Research	*The Basic Immune Simulator*, an agent-based model to study the interactions between innate and adaptive immunity (Folcik, An and Orosz 2007)
Chemistry	An agent-based approach to modeling molecular self-assembly (Troisi, Wong and Ratner 2005)
Crime Analysis	Agent-based model that uses a realistic virtual urban environment, populated with virtual burglar agents (Malleson 2009).
Ecology	Agent-based model of predator-prey relationships between transient killer whales and other marine mammals (Mock and Testa 2007).
Energy Analysis	Agent-based model for scenario development of offshore wind energy (Mast et al. 2007).
Epidemic Modeling	*BioWar*, a scalable citywide multi-agent model, that simulates individuals embedded in social, health, and professional networks and tracks the incidence of background and maliciously introduced diseases (Carley et al. 2006).
Market Analysis	Agent-based simulation that models the possibilities for a future market in sub-orbital space tourism (Charania et al. 2006).
Organizational Decision Making	Agent based modeling approach to allow negotiations in order to achieve a global objective, specifically for planning the location of intermodal freight hubs (van Dam et al. 2007).

4.2 Topologies as a Basis for Social Interaction

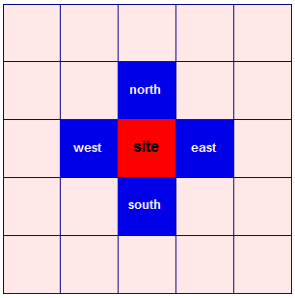
Agent-based modeling concerns itself with modeling agent relationships and agent interactions as much as it does modeling agents and agent behaviors. The primary issues of modeling agent interactions are specifying who is, or could be, connected to who, and the dynamics governing the mechanisms of the interactions. For example, an agent-based model of Internet growth would include mechanisms that specify who connects to who, why, and when.

A set of common topologies used in agent-based models for representing social agent interaction is shown in Figure 4. In the "soup," or aspatial model, agents have no location and the model has no spatial representation (Figure 4a). Generally, pairs of agents are randomly selected for interaction and then returned to the soup from which they came. Cellular automata represent agent interaction patterns and available local information by using a grid or lattice, and the cells immediately surrounding an agent are its neighborhood (Figure 4b). In the Cellular Automata model, agents move from cell to cell on a grid. Generally no more than one agent occupies a cell at a given time. The von Neumann "5-neighbor" neighborhood is shown in Figure 4b. In the Euclidean space model, agents roam in 2D, 3D or higher dimensional spaces (Figure 4c). In the Geographic Information System (GIS) topology, agents move over a realistic geo-spatial landscape (Figure 4d). Networks allow an agent's neighborhood to be defined more generally and sometimes more accurately. For the network topology, networks may be static or dynamic (Figure 4e). For static networks, links are pre-specified and do not change in the model. For dynamic networks, links (and possibly nodes) are determined endogenously according to the mechanisms included in the model.

No matter what topology is used in an agent-based model to connect the agents, the essential idea is that of local interaction and local information transfer between agents. The essential idea is that agents only interact at any given time with a limited (small) number of other agents out of all the agents in the population. This notion is implemented by defining a local neighborhood and thereby limiting interaction to a small number of agents that happen to be in that neighborhood. The point is that there is limited connectivity and information is confined to local exchanges. There is no such thing as global information. This is not to say that agents need to be located in close proximity to one another spatially to be able to interact. The network topology allows agents to be linked on the basis of relationships other than proximity. Most empirically-based social networks are have characteristic network topologies with a individual agents having a limited number of connections.

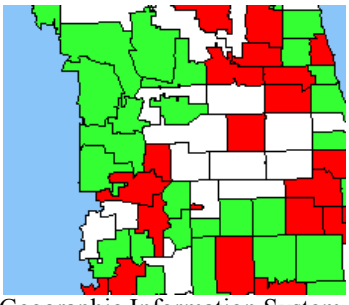


(a) "Soup" Model (Aspatial)

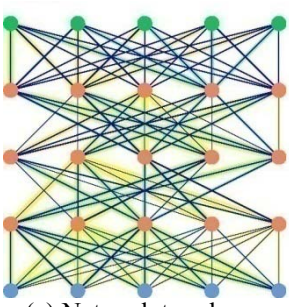


(b) Cellular Automata (von Neumann)

(c) Euclidean Space (2-D)



(d) Geographic Information System (GIS)



(e) Network topology

Figure 4 Topologies for Agent Relationships and Social Interaction

4.2.1 ABMS Software and Toolkits

Agent-based modeling can be done using general, all-purpose software or programming languages, or it can be done using specially designed software and toolkits that address the specific requirements for modeling agents. Agent modeling can be done in the small, on the desktop, or in the large, using large-scale computing clusters, or it can be done at any scale in-between. Projects often begin small, using one of the desktop ABMS tools, or whatever tool or programming language the developers are familiar with, and then grows the initial prototype in stages into larger-scale agent-based models, often using dedicated ABMS toolkits. Often one begins developing their first agent model using the approach that one is most familiar with, or the approach that one finds easiest to learn given their background and experience.

We distinguish several approaches to building ABMS applications in terms of the scale of the software that one can apply according to the following continuum:

Desktop Computing for ABMS Application Development:

- Spreadsheets: Excel using the macro programming language VBA
- Dedicated Agent-based Prototyping Environments: Repast Simphony, NetLogo, StarLogo
- General Computational Mathematics Systems: MATLAB, *Mathematica*

Large-Scale (Scalable) Agent Development Environments:

- Repast
- Swarm
- MASON
- AnyLogic
- Other

General Programming Languages:

- Python
- Java
- C++

Desktop ABMS can be used to learn agent modeling, prototype basic agent behaviors, and perform limited analyses. Desktop agent-based models can be simple, designed and developed in a period of a few days by a single computer-literate modeler using tools learned in a few days or weeks. Desktop agent modeling can be used to explore the potential of ABMS with relatively minor time and training investments, especially if one is already familiar with the tool.

Spreadsheets, such as Microsoft Excel, are in many ways the simplest approach to modeling. It is easier to develop models with spreadsheets than with many of the other tools, but the resulting models generally allow limited agent diversity, restrict agent behaviors, and have poor scalability compared to the other approaches designed specifically for agent modeling. Agent-based modeling in spreadsheets requires some macro-programming to be done in a language such as VBA (Visual Basic for Applications), the macro programming language for Excel and other Microsoft Office applications. Significant agent models have been developed entirely using spreadsheets (Bower and Bunn 2000). In previous WSC papers, we described an spreadsheet implementation of a spatial agent-based shopper model (Macal and North 2007).

Special-purpose agent tools, such as NetLogo, and StarLogo, provide special facilities focused on agent modeling. The most directly visible common trait shared by the various prototyping environments is that they are designed to get first-time users started as quickly as possible. NetLogo is a free ABMS environment developed at Northwestern University's Center for Connected Learning and Computer-Based Modeling (Wilensky 1999). The NetLogo language uses a modified version of the Logo programming language (Harvey 1997). NetLogo is designed to provide a basic computational laboratory for teaching complex adaptive systems concepts. NetLogo was originally developed to support teaching, but it can be used to develop a wide range of applications. NetLogo provides a graphical environment to create programs that control graphic "turtles" that reside in a world of "patches" that is monitored by an "observer." NetLogo includes an innovative participatory ABMS feature called HubNet, which allows groups of people to interactively engage in simulation runs alongside of computational agents (Wilensky and Stroup 1999). NetLogo continues as a subject of active development and new versions with expanded capabilities are released periodically.

General-purpose desktop computational mathematics system (CMS) with integrated development environments (IDEs), such as MATLAB and *Mathematica*, can be used to develop agent models, although the agent-specific functionality has to be written by the developer from scratch, as there are no dedicated libraries or modules that focus on agent-based modeling – at least not yet (Macal 2004). The basic requirements is knowledge of how to program in a scripting language. Computational mathematics systems are structured in two main parts: (1) the user interface that allows dynamic user interaction, and (2) the underlying computational engine, or kernel, that performs the computations according to the user's instructions. The underlying computational engine is written in the C programming language for these systems, but C coding is unseen by the user. The interpreted nature of these systems avoids the compilation and linking steps required in traditional programming languages. Computational mathematics systems have advantages derived from both the mathematical and interactive orientations of these tools. CMS environments have rich mathematical functions, and nearly any mathematical relation or function that can be numerically calculated is available within these tools or their add-on libraries. In some cases, the tools even support symbolic processing and manipulation, which is useful for systems of equations that can be solved analytically and can be exploited quite effectively to do agent-based modeling. In symbolic systems agent classes are defined as abstract data types (Macal 2004). If a CMS environment is already familiar to a developer, this can be a good place to start agent-based modeling.

Many large-scale ABMS software environments are now freely available. These include Repast (North, Collier and Vos, 2006), Swarm (SDG 2006; Minar et al. 1996), NetLogo (NetLogo 2007) and MASON (GMU 2006) among many others. Proprietary toolkits are also available such as AnyLogic (2006). A review and comparison of Java-based agent modeling toolkits is provided by Tobias and Hoffman (2004) and Nikolai and Madey (2009).

Swarm was the first ABMS software development environment, launched in 1994 at the Santa Fe Institute. Swarm was originally written in Objective C and was later fitted with a Java interface.

Following the original Swarm innovation, the Repast (REcursive Porous Agent Simulation Toolkit) toolkit was developed as a pure Java implementation (North, Collier and Vos, 2006). Repast will be described here in some detail to illustrate the capabilities that an agent-based toolkit can provide, as the authors have familiarity with the toolkit being its developers. Repast has been used extensively in social simulation applications (North and Macal 2005). Repast is a widely used free and

open source agent-based modeling and simulation toolkit (ROAD 2007). Repast Simphony (Repast S) is the latest version of Repast, designed to provide visual point-and-click tools for agent model design, agent behavior specification, model execution, and results examination. The Repast S agent model designer is being developed to allow users to visually specify the logical structure of their models, the spatial (e.g., geographic maps and networks) structure of their models, the kinds of agents in their models, and the behaviors of the agents themselves. Once their models are specified, users can use the point-and-click Repast S runtime environment to execute model runs as well as visualize and store results. In addition, the Repast S runtime environment includes automated results analysis and connections to a variety of spreadsheet, visualization, data mining, and statistical analysis tools, virtually all of which are free and open source.

As computational capabilities continue to advance in both hardware and software, new capabilities are continuously being incorporated into the latest versions of ABMS toolkits. The field is advancing rapidly toward highly scalable, high productivity agent development environments.

5 WHY AND WHEN ABMS

We conclude by offering some ideas on the situations for which agent-based modeling can offer distinct advantages to conventional simulation approaches such as discrete event simulation (Law 2007), systems dynamics (Sterman 2000) and other quantitative modeling techniques. Axtell (2000) discusses several reasons for agent-based modeling especially compared to traditional approaches to modeling economic systems. When is it beneficial to think in terms of agents? When one or more of the following criteria are satisfied:

- When the problem has a natural representation as being comprised of agents
- When there are decisions and behaviors that can be well-defined
- When it is important that agents have behaviors that reflect how individuals actually behave (if known)
- When it is important that agents adapt and change their behaviors
- When it is important that agents learn and engage in dynamic strategic interactions
- When it is important that agents have a dynamic relationship with other agents, and agent relationships form, change, and decay
- When it is important to model the processes by which agents form organizations, and adaptation and learning are important at the organization level
- When it is important that agents have a spatial component to their behaviors and interactions
- When the past is no predictor of the future because the processes of growth and change are dynamic
- When scaling-up to arbitrary levels is important in terms of the number of agents, agent interactions and agent states
- When process structural change needs to be an endogenous result of the model, rather than an input to the model

ACKNOWLEDGMENTS

This work was supported by the U.S. Department of Energy under contract number DE-AC02-06CH11357. Portions of this tutorial have appeared in previous tutorial papers presented at the Winter Simulation Conference for 2005 - 2008.

REFERENCES

AnyLogic. 2006. <http://www.xjtek.com/>.

Axelrod, R. 1997. Advancing the art of simulation in the social sciences," in Conte., R., Hegselmann, R. and Terna, P., eds. *Simulating social phenomena*, Berlin: Springer-Verlag: 21-40.

Axtell, R. 2000. *Why agents? On the varied motivations for agent computing in the social sciences,* Working Paper 17, Center on Social and Economic Dynamics, Brookings Institution, Washington, D.C.

Bonabeau, E., M. Dorigo and G. Theraulaz. 1999. *Swarm intelligence: from natural to artificial systems*, Oxford: Oxford University Press.

Bonabeau, E. 2001. Agent-based modeling: methods and techniques for simulating human systems. In *Proceedings of National Academy of Sciences* 99(3): 7280-7287.

Bower, J. and D. Bunn. 2000. Model-based comparisons of pool and bilateral markets for electricity. *The Energy Journal* 21(3):1–29.

Carley, K. M., D. B. Fridsma, E. Casman, A. Yahja, N. Altman, L.-C. Chen, B. Kaminsky and D. Nave. 2006. BioWar: scalable agent-based model of bioattacks, *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 36(2):252-265.

Casti, J. 1997. *Would-be worlds: how simulation is changing the world of science*, New York: Wiley.

Casti, J. 2001. Bizsim: the world of business - in a box, *Complexity International*, 08:6 <http://www.complexity.org.au/ci/vol08/casti01/>.

Charania, A. C., J. R. Olds and D. DePasquale. 2006. Sub-Orbital Space Tourism Market: Predictions of the Future Marketplace Using Agent-Based Modeling, SpaceWorks Engineering, Inc., Atlanta, GA, Available online at http://www.sei.aero/uploads/archive/IAC-06-E3.4.pdf.

Conway, S. R. 2006. An Agent-Based Model for Analyzing Control Policies and the Dynamic Service-Time Performance of a Capacity-Constrained Air Traffic Management Facility, ICAS 2006 - 25th Congress of the International Council of the Aeronautical Sciences Hamburg, Germany, 3-8 Sep. 2006, Availabe online at http://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20060048296_2006250468.pdf.

Epstein, J. M. and R. Axtell. 1996. *Growing artificial societies: social science from the bottom up*, Cambridge, MA: MIT Press.

Folcik, V.A., G.C. An and C.G. Orosz. 2007. The Basic Immune Simulator: An Agent-Based Model to Study the Interactions between Innate and Adaptive Immunity, *Theoretical Biology and Medical Modelling* 4(1):39-56.

Gardner, M. 1970. The fantastic combinations of John Conway's new solitaire game "Life", *Scientific American* 223:120-123.

GMU (George Mason University). 2006. MASON home page, <http://cs.gmu.edu/~eclab/projects/mason/>.

Griffin, A. F. and C. Stanish. 2007. An Agent-Based Model of Prehistoric Settlement Patterns and Political Consolidation in the Lake Titicaca Basin of Peru and Bolivia, Structure and Dynamics: eJournal of Anthropological and Related Sciences, 2(2) Availabe online at http://repositories.cdlib.org/imbs/socdyn/sdeas/vol2/iss2/art2.

Harvey, B. 1997. *Computer Science Logo Style*, MIT Press: Boston, Massachusetts USA

Holland, J. H. 1995. *Hidden order: how adaptation builds complexity*, Addison-Wesley:Reading, Mass.

Jennings, N. R. 2000. On agent-based software engineering, *Artificial Intelligence,* 117:277-296.

Law, A. M. 2007. *Simulation modeling and analysis*, 4th ed. New York: McGraw-Hill.

Macal, C. M. 2004. Agent-Based Modeling and Social Simulation with *Mathematica* and MATLAB, in *Proceedings of Agent 2004 Conference on Social Dynamics: Interaction, Reflexivity and Emergence*, Macal, C., D. Sallach, and M. North, eds., Chicago, IL, Oct. 7-9, available at <www.agent2004.anl.gov>, pp. 185-204.

Macal, C. M. and M. J. North. 2007. Agent-based Modeling and Simulation: Desktop ABMS. *Proceedings of the 2007 Winter Simulation Conference*. S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, eds. Washington, DC, pp. 95-106.

Malleson, N., 2009. Using Simulation to Predict Prospective Burglary Rates in Leeds and Vancouver, *7th National Crime Mapping Conference*, Manchester, U.K., May 7-8, 2009.

Mast, E. H. M., G.A.M. van Kuik and G.J.W. van Bussel. 2007. *Agent-Based Modelling for Scenario Development of Offshore Wind Energy*, Delft University of Technology, The Netherlands.

Minar, N., R. Burkhart, C. Langton, and M. Askenazi. 1996. *The Swarm simulation system, a toolkit for building multi-agent simulations*, <http://www.santafe.edu/projects/swarm/overview/overview.html>.

Mock, K. J. and J. W. Testa. 2007. *An agent-based model of predator-prey relationships between transient killer whales and other marine mammals*, University of Alaska Anchorage, Anchorage, AK, May 31, 2007, Available online at http://www.math.uaa.alaska.edu/~orca/.

Nikolai, C. and G. Madey. 2009. Tools of the Trade: A Survey of Various Agent Based Modeling Platforms, *Journal of Artificial Societies and Social Simulation* 12(2)2, <http://jasss.soc.surrey.ac.uk/12/2/2.html>.

North, M. J. and C. M. Macal. 2005. Escaping the accidents of history: an overview of artificial life modeling with Repast, in *Artificial Life Models in Software*, eds., A. Adamatzky and M. Komosinski, Springer-Verlag: Dordrecht, Netherlands.

North, M. J., and C. M. Macal. 2007. *Managing business complexity: discovering strategic solutions with agent-based modeling and simulation*, Oxford University Press: Oxford, U.K.

North, M., N. Collier, and J. Vos. 2006. Experiences in creating three implementations of the repast agent modeling toolkit, *ACM Transactions on Modeling and Computer Simulation*, 16(1):1-25.

NRC (National Research Council). 2003. *Dynamic social network modeling and analysis: workshop summary and papers,* R. Brieger, K. Carley, and P. Pattison, Committee on Human Factors, Washington, DC: National Academies Press.

Rao, A. S. and M. P. Georgeff. 1999. Modeling agents within a BDI-architecture, In *Proc. International Conference on Principles of Knowledge Representation and Reasoning (KR)*, Eds., R. Fikes and E. Sandewall, Cambridge, MA: Morgan Kaufmann.

ROAD (Repast Organization for Architecture and Design). 2007. Repast Home Page, Available at http://repast.sourceforge.net/

Reynolds, Craig. 2006. Boids. <http://www.red3d.com/cwr/boidss/>.

Sakoda, J. M., 1971. The checkerboard model of social interaction, *Journal of Mathematical Sociology*. 1:119-132.

SDG (Swarm Development Group). 2006. Swarm Development Group home page, <http://www.swarm.org>.

Sterman, J. D. 2000. *Business Dynamics: Systems Thinking and Modeling for a Complex World*, Boston: McGraw-Hill.

Tobias, R. and C. Hofmann. 2004. Evaluation of free Java-libraries for social-scientific agent based simulation, *Journal of Artificial Societies and Social Simulation*, 7(1), Jan. 31.

Troisi, A., V. Wong and M. A. Ratner. 2005. An Agent-Based Approach for Modeling Molecular Self-Organization, *Proceedings of the National Academy of Sciences* USA 102(2):255–260, Available online at <www.pnas.org/cgi/doi/10.1073/pnas.0408308102>.

van Dam, K. H., Z. Lukszo, L. Ferreira and A. Sirikijpanichkul. 2007. Planning the location of intermodal freight hubs: an agent based approach, *Proceedings of the 2007 IEEE International Conference on Networking, Sensing and Control*, pp. 187-192, London, UK, 15-17 April 2007.

Wilensky, U. 1999. *Netlogo*, Center for Connected Learning and Computer-Based Modeling, Northwestern University:Evanston, IL USA, http://ccl.northwestern.edu/netlogo/.

Wilensky, U., and W. Stroup. 1999. *Hubnet*, Center for Connected Learning and Computer-Based Modeling, Northwestern University: Evanston, IL USA, <http://ccl.northwestern.edu/ps/>.

AUTHOR BIOGRAPHIES

CHARLES M. MACAL, Ph.D., P.E., is the Director, Center for Complex Adaptive Agent Systems Simulation (CAS^2), Argonne National Laboratory. He is a member of the INFORMS-Simulation Society, Association for Computing Machinery, the Society for Computer Simulation International, the Systems Dynamics Society and a founding member of the North American Association for Computational Social and Organizational Science. Charles has a Ph.D. in Industrial Engineering & Management Sciences (Operations Research) from Northwestern University and a Master's Degree in Industrial Engineering (Computer Simulation) from Purdue. He is also a Registered Professional Engineer. Contact: <macal@anl.gov>.

MICHAEL J. NORTH, M.B.A., Ph.D., is the Deputy Director of CAS^2 at Argonne. Michael has over 15 years of experience developing advanced modeling and simulation applications for the federal government, international agencies, private industry, and academia. Michael has a Ph.D. in Computer Science from the Illinois Institute of Technology. Contact: <north@anl.gov>.