Discrete Event Simulation and Social Science:

The XeriScape Artificial Society

Gordon C. Zaft
NewMonics
877 S. Alvernon Way Suite 100
Tucson, Arizona 85711 USA

and

Bernard P. Zeigler
Arizona Center for Integrative Modeling and Simulation
University of Arizona
Tucson, Arizona 85721 USA

ABSTRACT

Social scientists use artificial society simulations to explore complex behaviors that result from the interaction of agents. One such artificial society is Epstein and Axtell's Sugarscape simulation. In Sugarscape, agents are born, eat 'sugar', travel, reproduce, and die in a torus-shaped virtual world. Sugarscape uses simple rules to create a virtual society where agent interactions aggregate to form surprisingly complicated social structures.

This paper presents XeriScape, a Sugarscape-style artificial society based on the Discrete Event System Specification (DEVS) formalism implementation in the Java language (DEVS-Java). DEVS-Java is a powerful tool that makes XeriScape a more efficient, flexible, and extensible artificial society simulation than Sugarscape. A number of experiments illustrate the capabilities and advantages of the DEVS-Java environment for artificial societies and other social science applications.

Keywords: Artificial society, discrete event simulation, DEVS, Java, agent based simulation.

INTRODUCTION

Epstein and Axtell's Sugarscape project [1] explored the application of agent-based simulation to social science in a way that was both accessible to non-specialists and interesting to social scientists. Their work was unusual in that it was conceptually simple and yet capable of broad interpretation and easy extension.

Consideration of the Sugarscape project raised the possibility of applying Zeigler's Discrete Event System Specification (DEVS) [2] technology to a Sugarscape-style simulation. Such a simulation would be intriguing for two reasons: first, because DEVS technology has primarily been applied to problems in the physical sciences, and second, because the power, flexibility and efficiency of DEVS could result in a very capable and broadly extensible system. In applying DEVS to an artificial society, the DEVS problem space has been expanded in a fruitful and interesting way – one that can benefit both the social science and simulation communities.

The XeriScape artificial society was implemented using the Java implementation of DEVS (DEVS-Java). Java was chosen due to its platform independence, simplicity of implementation, and easy integration of a graphical user interface.

DISCRETE EVENT SYSTEM SPECIFICATION (DEVS)

The DEVS formalism was developed by Zeigler in order to provide a solid theoretical foundation for discrete event simulation. Discrete event simulation is a technique that, as the name implies, is based not on time steps but on events. Where a discrete time simulation has a clock that steps in even increments (1, 2, 3…), a discrete event simulation steps from event to event (A, B, C…). Although discrete event simulations do have a clock, it is not constrained to step in even increments, but from the time of the first event to the time of the second event, and so forth.

The classic discrete event system specification [3, pp. 75-76] is a structure

$$M = \{X, S, Y, \delta_{int}, \delta_{ext}, \lambda, ta\}$$

Where:

X is the set of input values
S is the set of states
Y is the set of output values
$\delta_{int}: S \rightarrow S$ is the *internal transition* function
$\delta_{ext}: Q \times X \rightarrow S$ is the *external transition function*, where
  $Q = \{(s,e) \mid s \in S, 0 \leq e \leq ta(s)\}$ is the *total state* set
  e is the *time elapsed* since last transition
$\lambda: S \rightarrow Y$ is the output function
$ta: S \rightarrow R^{+}_{0,\infty}$ is the set of positive reals from 0 to ∞

Briefly stated, at any given time the system is in some state s. If no disturbing event occurs, the system will stay in that state until time *ta* has elapsed; at which point the system will process its output ($\lambda(s)$) and then change to state $\delta_{\mathrm{int}}(s)$ (i.e. an internal transition occurs). If an external event occurs, the system will change to state $\delta_{\mathrm{ext}}(s,e,x)$.

DEVS models may be *atomic* (i.e. a single, independent model) or *coupled* (models hierarchically composed of other models). The particular strength of DEVS is that it has been shown to be closed under coupling, that is, if a model A is composed of models B, C, and D, and B, C, and D are valid DEVS models, then A is also a valid DEVS model. This ability to compose complex models from simple building blocks makes it possible to build very large models.

THE XERISCAPE MODEL

The XeriScape model is composed of agents, cells, and rules that interact to define and guide the behavior of the XeriScape. Agents live in a space (the XeriScape) made up of cells. Their behavior, and the behavior of the 'environment' in which they exist, is regulated by a number of rules which interact to control the behavior of the agents and cells. XeriScape components are very general and are easily extensible.

Agents

Agents are the dynamic part of the XeriScape simulation. Their actions and interactions create the artificial society of the XeriScape. Agents move from cell to cell seeking resources such as water. Each agent is a unique individual with its own life cycle, attributes, history, and resources.

The initial implementation of XeriScape includes four different types of agents – the basic agent definition (Agent) and its extension into three additional agent types: the Finite Agent, the Polluting Agent, and the Gendered Agent. These four agent types share most properties in common.

Agent. Each Agent has a number of attributes that define it. Agents move from cell to cell, seeking resources they need, gathering them and consuming them. Agents have a limited vision and movement range. Vision and movement are linked; a vision of *n* cells implies a movement capability of *n* cells as well. The agent has an unlimited lifespan; given enough resources, it is immortal. Each agent has its own rate of resource consumption (metabolic rate).

The agent lifecycle consists of three activities: looking, moving, and consuming. The agent looks for resources, moves to the best available site, and then consumes the available resources.

Finite Agent. The Finite Agent is an agent that has a definite, determined lifespan. Finite agents live to be a specific age that is determined when they are created; then they die. Finite agents have the same attributes as Agents, but in addition, they have a maximum age (i.e., the age at which they will die).

Polluting Agent. The Polluting Agent is a finite agent that creates pollution. Whenever the polluting agent gathers resources or consumes them, pollution is produced as a side effect. The amount of pollution produced is a multiplier of the amount of resources gathered or consumed and a constant determined when the agent is created.

The polluting agent is sensitive to the pollution it creates. When a polluting agent is deciding where to move, it tries to avoid moving to areas that are polluted. In making its movement decision the polluting agent takes into account the amount of pollution in possible destinations and the amount of water available; all other factors being equal, it will move to the site with the least pollution.

Polluting agents have two attributes in addition to the finite agent attributes: the resource-gathering pollution multiplier constant α, and the resource-consumption pollution multiplier constant β. These constants are used to compute the amount of pollution produced by the agent when it gathers or consumes resources.

Gendered Agent. The Gendered Agent is a finite agent that has gender and the ability to reproduce. Gendered agents have a number of attributes in addition to those of the finite agent. They include age at puberty, age at which fertility ends (menopause), and the names of the father and mother.

The gendered agent has a modified lifecycle. After the standard look, move, and consume activities, the gendered agent will look to its neighboring cells to see if a suitable mate is available. If the gendered agent is fertile and one or more fertile neighbors is found, mating will occur with the first suitable partner found.

Gendered agent fertility is determined by a number of factors. First, the agent must be of childbearing age, i.e., older than puberty and younger than menopause. Second, the agent must have an amount of water resources (wealth) equal to at least half of the amount it had when it was born. These resources (half each from mother and father) are given to the child at birth. Children's vision and metabolic rates are determined in a Mendelian genetic cross; while simplistic, it is a reasonable approximation. Thus, the child will receive either its mother's or its father's vision, and either its mother's or its father's metabolism.

The child's inheritance from its parents consists solely of its genetic makeup and the amount of water it receives when it is born. There is no other inheritance mechanism for transfer of resources between generations currently implemented. The lack of an inheritance mechanism can be thought of as being equivalent to a 100% inheritance tax.

Cells

The cell is the basic unit of space in the XeriScape model. Each cell has four neighbors: north, south, east, and west (diagonals are not used in the XeriScape).

Cells on the edge of the XeriScape wrap to the other side of the space, that is, the space is not really a flat, two-dimensional space but is instead a torus (doughnut shape) made up of two-dimensional elements. Each cell is considered a point, thus, everything inside the cell is present to everything else.

Cells contain resources that are consumed and replenished over time. In the case of the XeriScape, the primary resource is water. Since the distribution of resources is not uniform over the XeriScape, some resource "peaks" and "valleys" exist. The rate at which resources are replenished is determined by a rule.

Cells contain agents, and each cell can contain a pre-specified number of agents. In Sugarscape, cells could only contain 1 agent at a time, and so the XeriScape defaults to this behavior. XeriScape supports the ability to put an arbitrary number of agents in each cell, although this ability is not demonstrated in this paper.

Cells can become polluted by the activities of the agents living in them. This pollution increases as agent gathering and consumption activity continues, but is dispersed gradually by a diffusion process. Note that pollution is dispersed, but never disappears from the environment, just as in the real world. This dispersal process averages the amount of pollution in a cell with the pollution level in its immediate neighbors.

The topology of the XeriScape simulation is built to resemble that of Sugarscape, with two "peaks" or concentrations (oases) of water, one at the top right and another at the bottom left of the simulation. This topology is called the "Twin Peaks" topology. Other topologies can be easily created.

Rules

Rules are the laws of the XeriScape. The variety of rules available and the ease with which new ones can be written is where the true power and flexibility of the XeriScape model originates. Rules control such elements as the initial distribution of agents in the XeriScape, the rate at which resources regenerate in a cell, the "genetic makeup" of each agent, the criteria used to compare possible destination cells, and much more. There are only four fundamental rules for each simulation: a resource comparison rule, a resource initialization rule, an agent initialization rule, and an agent distribution rule.

Resource Comparison Rule. The Resource comparison rule (RCR) determines how an agent compares the resource values of a cell into which the agent is considering moving. In the simplest case, it involves a straight comparison of values, but in some cases, it is more complex. For example, the presence of pollution in a cell may make it a less desirable destination than another cell with less water but no pollution.

Resource Initialization Rule. The resource initialization rule (RIR) creates the basic landscape of the simulation. It determines the types of resources, the resource capacity and initial resource allocation for each cell. The RIR uses a sub-rule to determine the manner or rate at which resources are replenished in each cell. This is termed the growback rate. The growback rate is expressed in terms of units of resource per unit time. Note that it is also permissible to have an infinite growback rate, i.e., resources grow back to their full capacity immediately. The RIR, combined with the number of rows and columns in the simulation, determines the topology of the XeriScape.

Agent Initialization Rule. The agent initialization rule (AIR) is responsible for creating new agents. The simulation calls the AIR when the simulation starts in order to generate the initial complement of agents. The AIR is also used to generate replacement agents when new agents are injected into the simulation, for example by sexual reproduction, or when agents die.

Agent Distribution Rule. The agent distribution rule (ADR) determines where newly-created agents are injected into the simulation. This distribution may be random, clustered, or organized in some fashion. Only a random distribution is used in this initial implementation.

EXPERIMENTAL RESULTS

A number of experiments were performed to demonstrate the XeriScape system. Selected results are described below, for a more complete presentation see [4].

Finite Agents

In this experiment agents have a finite lifetime. This lifetime is established at the agents' creation and is random in a predetermined range (in this case, it varies from 60 to 100). When the agent dies (whether by starvation or old age), it is replaced by a new agent of age 0 that is injected into the simulation at a random location. This replacement is called replacement mode. The number of agents in this experiment was set to 125 to approximate the carrying capacity of the XeriScape.

This experiment provides a platform for examining water resource (wealth) distributions and how they change over time. The data show that as time goes on agent wealth becomes more concentrated. The top decile of the societal wealth is held by fewer and fewer individuals. This is due to the fact that longer-lived agents which are born in resource-rich areas of the XeriScape can accumulate relatively great wealth, especially if they have a low metabolism and a high vision.

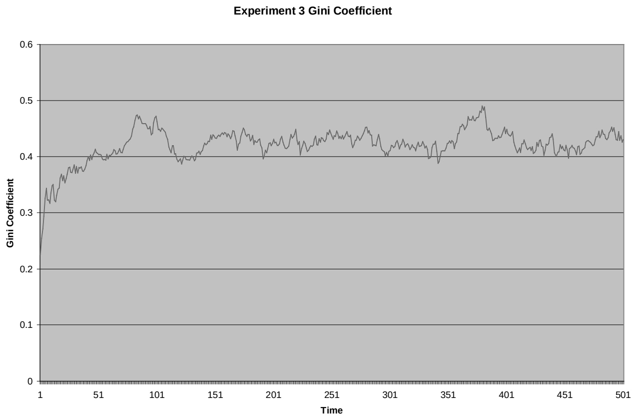

Figure 1

Figure 1 shows the Gini coefficient for this experiment (Experiment 3) as a function of time. The Gini coefficient [5] is used to gauge the equity of wealth distribution in a society (0.0 is perfectly equitable; 1.0 is perfectly inequitable). Figure 1 shows that the wealth distribution rapidly becomes more skewed, then settles into a range from roughly 0.4 to 0.5.

Pollution

This experiment explores the role of pollution in altering agent behavior and the environment. The experiment starts with 400 agents. As in the previous experiment, these agents have a finite lifetime that is randomly fixed between 60 and 100 time steps. Unlike the previous experiment, these agents are not replaced when they die. The agents use a new rule to determine their movements; according to this rule, agents will move to cells with the highest ratio of water to pollution.

This experiment shows the agents initially concentrating on the resource peaks, then gradually moving away from them as the levels of pollution grow.

Gendered Reproduction

This experiment utilizes the GenderedAgent class. Agents grow, consume, and gather as before. If they are fertile, have enough resources and are adjacent to an agent of the opposite sex who is also fertile, they also mate and produce a child. Agents will mate once every cycle if they have enough resources and a suitable partner is available. Children receive a Mendelian cross of their parent's metabolism and vision attributes. The simulation starts with 400 agents randomly distributed (roughly 50% male and 50% female, randomly generated).

Other than their initial endowment of water, children receive no inheritance from their parents in this setup. The resources that an agent has gathered when it dies disappear from the environment.

In previous experiments the population dropped off precipitously and then stabilized at a level much lower than the initial population. In this experiment (Experiment 6), the population (that is, the carrying capacity) of the XeriScape with gendered agents actually increases with time (see Figure 2 below).

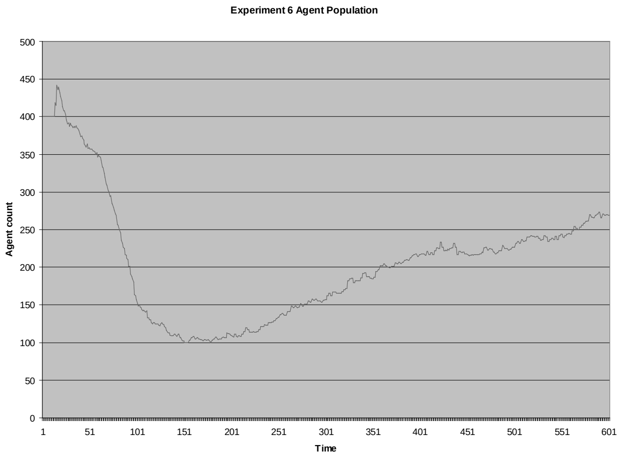

Figure 2

Mean agent vision increases over time and mean agent metabolism decreases. This is due to the genetic selection occurring with each generation selecting for increased vision and decreased metabolism. Agents with these characteristics are more likely to breed, and will breed more often since they generally will have greater wealth.

This effect explains the increase in carrying capacity; as mean metabolism decreases and mean vision increases, agents become more efficient gatherers and use fewer resources, so the XeriScape can support more of them.

Variable Time Steps

This experiment illustrates the enhanced capabilities made possible by DEVS-Java. In this experiment the XeriScape contains a mixture of finite and polluting agents. In addition, each agent has its own time step, that is, each agent has its own level of activity, which determines the rate at which its life cycle is executed. The time step varies from agent to agent, but it is fixed for the life of each agent. The time step values are randomly distributed from 5 to 15 time units. Since agents will not be waking up as often, they are given a larger initial resource value to prevent them from dying before they wake up.

This variable time step translates the discrete time simulation approach of Sugarscape into a true discrete event simulation. It can be viewed as a measure or indicator of the agent's vitality or energy level, or perhaps of its will to survive or alertness. Since the agents do not cycle in lock step the simulation is more like a real society, where individuals move at their own pace.

This experiment starts with 400 mixed finite (non-polluting) and polluting agents distributed randomly throughout the environment.

This experiment shows, in a particularly striking fashion, the enhanced capabilities DEVS-Java makes possible. The ability of the simulation to respond to each individual agent means that each time step is reached only if one or more agents wake up at that time step. Thus, during a simulation run, the clock may step from 5, to 7, to 8, to 10, and so forth. This means the simulation is as efficient as possible since no computations are done for the time steps that are stepped over. For example, executing the simulation for the first 50 time units required only 33 computation cycles instead of the 50 that Sugarscape would have required.

CONCLUSIONS

The XeriScape simulation showed how the power and flexibility of DEVS-Java could be applied to social science problems to provide more efficient and robust simulation capability.

One way that DEVS is more efficient lies in its object orientation and discrete event paradigm. Since DEVS is event-driven, only cells that contain agents require computation. Figure 3 shows the cumulative number of active cells as a function of time for the gendered reproduction experiment. The upper line represents the number of computational cycles performed in a discrete time simulation; the lower line shows the number of actual computational cycles for XeriScape. Clearly XeriScape can perform more efficiently even without taking advantage of the additional capabilities inherent in the DEVS environment.

The variable time step experiment showed an even greater performance increase. In the first 30 time units of simulation for that experiment, only 3,332 cumulative active cells existed. In the corresponding discrete time simulation the number would be 30 time units × 2,500 cells, or 75,000. The DEVS-Java implementation is over 22 times faster! Furthermore, as the calculation above shows, the larger the environment the greater the performance increase is likely to be. DEVS-Java is a high-performance simulation environment.

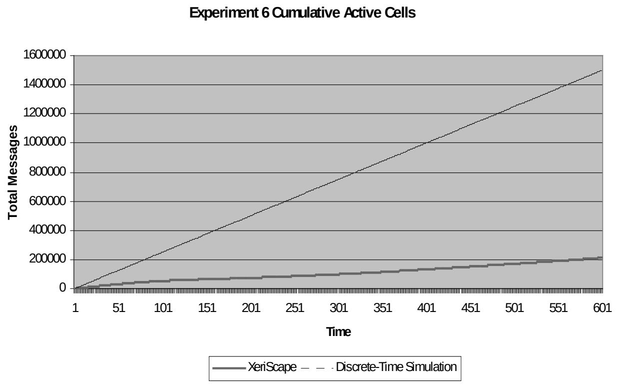

Figure 3

Besides these additional capabilities, XeriScape has shown that DEVS (and specifically DEVS-Java) is a very flexible, capable environment for social science simulations. The object-oriented nature of Java and the general design of XeriScape make it easy to develop new rules and new types of agents and insert them into the simulation environment. The inheritance property of Java means that new agents and new rules can inherit most of their code from previous work; it is only necessary to write the new code that differentiates them from their predecessors. An examination of the agent code for XeriScape shows that a great deal of code for the more sophisticated agents (GenderedAgent, for example) is inherited from the base Agent class. Furthermore, Java's ability to instantiate objects on the fly allows new agents and rules to be plugged in at any time. Finally, Java's cross-platform portability means that the XeriScape simulation can be run on any computer that has a Java implementation.

This initial version of the XeriScape simulation utilized an early version of DEVS-Java. More recent versions of DEVS-Java include facilities for distributed and collaborative modeling and simulation, as well as support for the US Department of Defense High Level Architecture (HLA) distributed modeling and simulation framework. More information is available at [6].

REFERENCES

[1] J.M. Epstein and R. Axtell, Growing Artificial Societies: Social Science from the Bottom Up, Washington, D.C.: Brookings Institution Press/Cambridge, Mass.: MIT Press, 1996.

[2] B.P. Zeigler, Theory of Modeling and Simulation, New York: John Wiley, 1976.

[3] B.P. Zeigler, H. Praehofer and T.G. Kim, Theory of Modeling and Simulation, 2nd edition. Academic Press, 2000.

[4] Gordon Zaft, Social Science Applications Of Discrete Event Simulation: A DEVS Artificial Society. Tucson, Arizona: University of Arizona Electrical and Computer Engineering Department Master's Thesis, 2001.

[5] C. Dagum,, "Gini Ratio," The New Palgrave Dictionary of Economics, edited by J. Eatwell, M. Milgate, and P. Newman. New York: Stockton Press, 1990.

[6] http://www.acims.arizona.edu, the website for the Arizona Center for Integrative Modeling and Simulation (ACIMS).