12

Computational Spectrum of Agent Model Simulation

Kalyan S. Perumalla
Oak Ridge National Laboratory
USA

1. Introduction

1.1 Overview

The study of human social behavioral systems is finding renewed interest in military, homeland security and other applications. Simulation is the most generally applied approach to studying complex scenarios in such systems. Here, we outline some of the important considerations that underlie the computational aspects of simulation-based study of human social systems. The fundamental imprecision underlying questions and answers in social science makes it necessary to carefully distinguish among different simulation problem classes and to identify the most pertinent set of computational dimensions associated with those classes. We identify a few such classes and present their computational implications. The focus is then shifted to the most challenging combinations in the computational spectrum, namely, large-scale entity counts at moderate to high levels of fidelity. Recent developments in furthering the state-of-the-art in these challenging cases are outlined. A case study of large-scale agent simulation is provided in simulating large numbers (millions) of social entities at real-time speeds on inexpensive hardware. Recent computational results are identified that highlight the potential of modern high-end computing platforms to push the envelope with respect to speed, scale and fidelity of social system simulations. Finally, the problem of shielding the modeler or domain expert from the complex computational aspects is discussed and a few potential solution approaches are identified.

1.2 New Computational Challenges

Computational social science has been an area of research for several decades now. Generally speaking, experiments in computational social science so far have been on the side of small scale (perhaps, at the limit, a few thousands of interacting entities). Lately, a general surge is apparent in an interest to represent and capture detailed effects at much larger scale. Scales of interest include population counts of cities, states, nations or even the world (10^4-10^9). Computational aspects that were not prominent at the smaller scale are now becoming pronounced at larger scale. Important orthogonal dimensions are emerging, making it necessary to revisit the computational problem with a fresh look. Dimensions

such as simulation speed, model scale, simulation system usability, and multi-system interoperability, which were all once implicitly combined together and insignificant in and by themselves for small scale models, are now separating themselves out as independent dimensions at larger scale. This separation is requiring the exploration and investigation of optimal locations in the dimension space for specific problems (or problem classes) of interest.

A meta-level question of course remains, namely, whether, and to what degree, simulation-based study is useful for the purposes of studying large-scale social systems. Perhaps new modeling and analysis methods are to be invented and applied to better deal with accuracy, precision, sensitivity and other concerns. In the absence of a generally applicable, comprehensive alternative modeling paradigm, simulation-based analysis remains the best promise towards studying these large-scale social systems. Simulation, in combination with additional theoretical methods such as design of experiments, seems to be the method of vogue and acceptance in the community. It is in this context that we focus on the new computational ramifications of large-scale social science simulations.

An important insight we put forward here is that simulation-based studies fall into distinct classes, each class being characterized by its specific combination of scale and accuracy. The purpose (also known as "use-case") behind using a particular class of simulation models becomes important to articulate and define, since the purpose defines both the way in which results from the simulation are to be interpreted, as well as the computational effects that one has to expect from using that class of models. For example, when a simulation is intended to generate an overall qualitative result (such as stumbling upon or uncovering surprising behavior), the simulation runs must be fast enough to qualitatively explore a large search space, yet the exact quantitative outcomes must not be interpreted literally. Similarly, population models intended to serve as reasonable situational surrogates for the masses (e.g, in order to test a detailed model of an antagonist group leader) must be capable of sustaining a large number of individuals, yet be computationally fast to allow for multi-scenario experimentation; consequently, a high degree of fidelity may not be an appropriate expectation for the masses in such a usage.

The challenge, then, is to either automatically find the right level of fidelity for a specific usage, or be able to sustain as high a fidelity level as possible at any scale that may be presented by the modeler to the simulation system. This is a grand challenge, which perhaps will remain unsolved in the near future. An intermediate step is to become aware of the issues and realize the distinctions so that expectations and choices are made appropriately.

The rest of document is organized as follows. The main computational dimensions underlying the simulations are presented in Section 2, along with placement of popular modeling systems in the space of speed, scalability and fidelity dimensions. The notion of simulation usage scenarios and some of the common usage classes are described in Section 3. A quantitative estimate of computational time requirements is presented in Section 4, for different ranges of factors constituting simulation execution. Two case studies are presented on state-of-the-art large-scale simulation systems to highlight the potential of next generation social behavioral simulation systems. The first is a graphics processor-based solution called GARFIELD, presented in Section 5, and the second is a cluster computing-based solution called μsik, benchmarks of which have been scaled to supercomputing platforms, presented in Section 6. The observations of the article are summarized in Section 7.

2. Orthogonal Computational Dimensions

As mentioned before, the computational side of social behavioral simulation can be split into multiple dimensions in light of the new generation of large-scale simulation scenarios that are being contemplated. We identify five important dimensions, which are mutually orthogonal. The orthogonality is defined in the sense that any given combination of values along the five dimensions can correspond to a desired combination for some social science simulation scenario of interest.

2.1 Dimensions

The five dimensions are: scale, speed, fidelity, usability and interoperability. Each of these dimensions is discussed next.

Scale

Scale can be defined as the largest number of encapsulated units logically or actually instantiated in the model during simulation. The encapsulated units correspond to concepts widely referred to as entities, agents, actors, players, components and so on. In agent-based simulations, for example, the number of agents is a natural measure of scale; each agent is an encapsulated unit in the model and the agents are actually instantiated in the model during simulation. In aggregate methods, the determination of scale is less obvious, since the units being modeled might be logically represented, rather than actually instantiated. Nevertheless, logically aggregate representation can be used to define the scale. For example, in epidemic models that are based on differential equations (e.g., the SIR model (Daley and Gani 2001; Staniford, Paxson et al. 2002; Zou, Gao et al. 2003)), the number of units is represented by a single variable N. Each of the units (from the uninfected, susceptible or infected populations) constitutes a logically encapsulated modeling unit, although they are lumped together in one model variable. For our purposes, the scale is therefore N.

In general, scale is harder to identify in aggregate models, while it is easier in more detailed models that have an approximately one-to-one mapping from system-level units to modeled units. However, when logical representation is included in the account, along with instantiated representation, this definition of scale makes the computational dimension of scale orthogonal to other dimensions, especially to fidelity (discussed later in this section).

Speed

Speed is the inverse of wall clock time elapsed from configuration/initialization to the end of simulation. This is a dimension that is easily measured for an execution in a given simulation system. Speed does depend on some of the other dimensions in a fundamental sense. However, for a given system, different implementation approaches can exist, each giving a different level of speed. The computing hardware platform can also have a significant bearing on the speed. There is, however, an upper bound on speed for any given combination of model and platform. Often, of interest is either the raw speed (e.g., to help estimate the time for parameter sweeps in a multi-simulation design of experiments), or the real-time scale factor (a fraction less than unity being slower than real-time, unity being exactly real-time, larger than unity being that many fold faster than real-time). Clearly,

speed is affected by many variables, including the complexity of underlying algorithms, synchronization efficiency, hardware/software implementation platform and so on.

Fidelity

Fidelity of a model is a concept that is harder to define absolutely yet possible to discuss about in a comparative fashion. Fidelity in general is the extent of detail of the system captured by the model. Note that this is distinct from scale in the sense that scale measures the count of encapsulated units, where as fidelity measures the amount of behavioral detail captured per encapsulated unit. In general, the detail could be not only intra-unit, but also inter-unit (e.g., additional global phenomena, such as ambient economic conditions, that span multiple units). Operationally, fidelity is a qualitative combination of the size of state and the number of activity "threads" representing the behavior in each encapsulated entity. Fidelity clearly has implications on computational efficiency; this is discussed later in Section 0.

For some objectives, adding more detail to the simulation may not bring a proportional amount of precision to the results. Nevertheless, the issue of value of fidelity is a separate concern; the dimension of fidelity can be discussed without necessarily linking it to overall value.

Note that coarseness of model is different from level of fidelity. Coarseness of modeling unit is typically reflected in the number of entities modeled as one modeling unit. For example, either an entire town could be modeled as one unit (giving several thousand modeled units per entity), or each individual is explicitly represented as a modeled unit. Fidelity on the other hand can be viewed as the amount of detail assigned to the behavior of each modeled unit. Of course, in this view, there is an implicit assumption on the separation of constituent entities from their behavioral dimensions.

Usability

An important concern that has practically dominated computational social science so far is that of usability. Usability is simply the inverse of the total amount of effort expended by the modeler to define, develop, debug, test, execute, animate, visualize, interpret and analyze simulations. Since social system modelers are not necessarily computational experts, there is emphasis on reducing the amount of effort needed to pose questions, explore, and get answers (often visually), in a point-and-click fashion.

Many of the popular social simulation systems today are driven primarily by this dimension. Once this usability is achieved to some good degree, other dimensions are explored as additional "wishes", such as scale and speed, in a secondary fashion. Usability simply reigns as supreme among the dimensions in social science simulation systems today.

In light of next generation modeling and simulation of social systems, unfortunately, the usability concern is no longer an easy one to address without having to consider its interaction with the other dimensions. While it is relatively straightforward to attain high levels of usability (ease of overall use) at low levels of scale, speed and/or fidelity, it is an entirely different matter to do so at large scale, high fidelity and/or high speed.

With current usability techniques, unfortunately, scale and/or fidelity cannot be increased without significantly affecting speed. Performance penalties, hidden heretofore under small scale/fidelity, rise to significant levels, with slowdowns exceeding 1000×. The sources of the penalties are varied, from the slow speed nature of interpreted languages, to overheads of

heavy graphics, to instrumentation overhead for runtime flexibility of configuration and monitoring.

Interoperability

Interoperability is the ability to interface and integrate disparate, complementary subsystems into an integrated system. In social science simulations, interoperability can be used to reuse previously developed models in a new scenario or to interoperate models at different resolutions. For example, a model of a popular leader may be interfaced with a model of the general masses, in order to exercise the leader model dynamically or to uncover overall system behaviors under various scenarios.

Interoperability is a hard problem. In general, it remains hard even in the simplest setting, namely, of two models developed in the same programming language, same modeling framework and simulation system. There is quite a bit of literature on interoperable systems, many of the systems falling under the category of syntactic interoperability. Semantic interoperability, on the other hand, is the more difficult component.

Interoperability is a dimension orthogonal to the rest in the sense that one could achieve any combination of the rest four dimensions without making any headway in the interoperability aspect. Alternatively, one could strive for interoperability but that needs to be done with awareness of the regimes of the other dimensions at which the system being interoperated spans. For example, interoperability of graphics processor-based automata models with supercomputing-based agent models can only be undertaken at the levels of fidelity and scale that the automata and agent models on those platforms afford.

2.2 Modeling Approaches Spanning Scalability and Fidelity

There are several modeling frameworks that are available to use in modeling social systems. Each framework is implemented in some software system; a (non-exhaustive) list of implemented systems includes JSAF(Davis, Lucas et al. 2005), SEAS(Chaturvedi, Foong et al. 2005), PMFServ(Silverman 2008), CultureSim(Silverman 2008), NetLogo(Wilensky 1999), Mason(Luke, Cioffi-Revilla et al. 2004), Repast J/.Net(North, Collier et al. 2006), Symphony, Swarm(Walter, Sannier et al. 2005), TeD(Perumalla, Fujimoto et al. 1998), Maisie(Bagrodia and Liao 1994), SSF(Cowie, Liu et al. 1999), Arena and NetLogic, to pick a few representative ones from each type of framework.

Each modeling framework possesses its ranges of scalability, fidelity and speed. The typical ranges of some of the prominent frameworks are shown in Fig. 1. The region below the real-time diagonal indicates the combination of fidelity and scalability levels that can be executed fast enough to meet or beat real-time. The region above the diagonal line indicates the specific combinations of fidelity and scalability that take more than one second of wall clock time for each second simulated in the model.

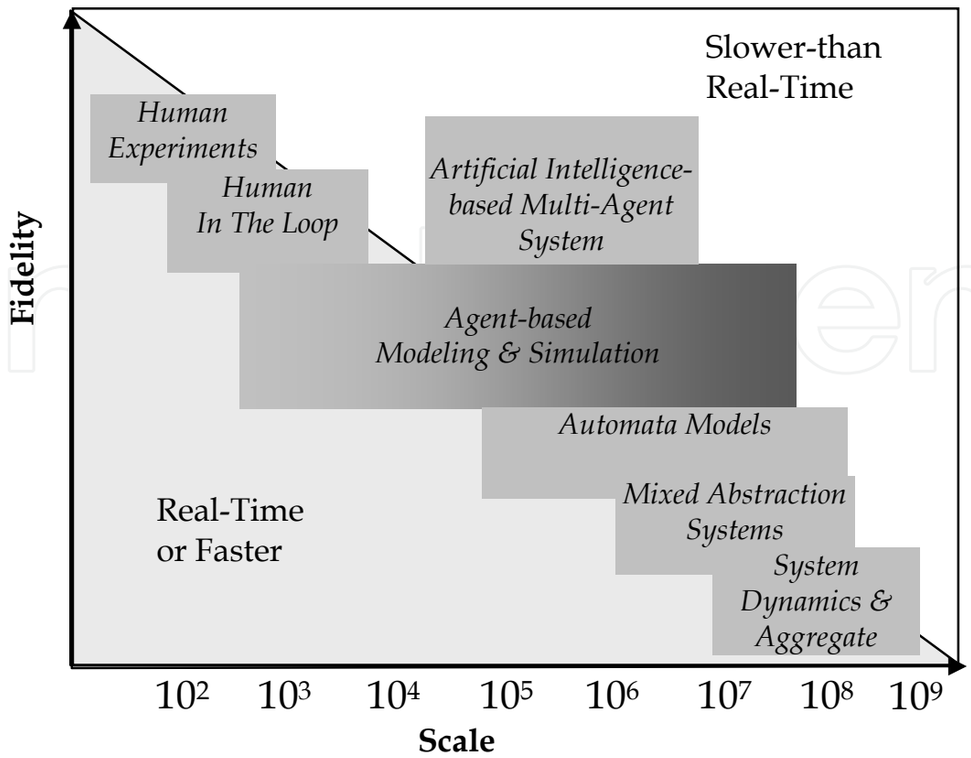


Fig. 1. Spectrum of behavioral modeling alternatives, and the scalability and fidelity ranges they afford.

Experiments involving real humans affords the maximum fidelity, but can only be practically performed with at most 10^2-10^3 individuals. Human-in-the-loop systems, such as semi-automated forces, can conceivably be performed with 10^2-10^3 human participants coupled to the system containing other artificially generated behaviors. The fidelity afforded is high on the human participation side of the system, but is reduced due to bi-directional interaction with artificial behaviors. Agent-based Modeling and Simulation(North and Macal 2007) affords the maximum range of scalability at moderate levels of fidelity. It is shown with extended (darkened) range to the right, denoting the recent increase in the number of agents that could be simulated using novel computing solutions such as data-parallel execution on graphical processors(D'Souza, Lysenko et al. 2007; Perumalla and Aaby 2008) and reverse computation-based control-parallel execution on supercomputers. Automata-based models are computationally simple enough to be executed in large entity counts, but at the cost of strictly lower fidelity. Mixed abstraction systems are those that combine more than one modeling paradigm, for ease of modeling or increase in speed. A classic example is the use of a small number of agent models placed in the context of a large population of automata models, delivering the higher fidelity of the agent models for important components while delivering the high speed of automata models for the less detailed population behaviors. *System Dynamics* and aggregate models are based on coupled differential equations. They afford the maximum level of scalability, since increasing the unit count could be as simple as increasing the value of a variable in the equations. However, they are also the lowest in fidelity.

2.3 Computational Aspects of Fidelity

A note can be made about the interaction of fidelity with computational burden. The issue at heart of tradeoffs between scale, speed and fidelity is the nature of computational artifacts that underlie the modeling primitives. An activity, such as random walk on a plane, can be realized as a computational thread instantiated in the context of an entity. Each activity thus consumes computational resources (computer memory and wall clock time), and also typically comes with additional state size represented in the entity (e.g., current location, movement pattern specification, etc.). Each entity can in general have several such activities coordinating with each other to realize the overall entity behavior. The number of activities translates to a corresponding number of computational threads that are instantiated during simulation execution. At runtime, these activities need to be allocated, scheduled, de-scheduled, synchronized and so on, all of which consume wall clock time as well as computer memory. Fidelity of the model translates to greater number of activities, more complex computation within each activity, and/or greater frequency of invocations to activity functionality. An automata-based model is a special case in which each entity contains a singleton activity which manipulates a (simple) state according to a state transition table. Thus, automata models typically are computationally lighter in weight, enabling larger scale at the same simulation speed level as agent-based models.

3. Simulation Usage Scenarios

In order to understand the computational challenges underlying computational social sciences, it is important to distinguish among various typical usage scenarios of social behavioral simulations. The usage scenarios are sufficiently disparate from each other, both qualitatively and quantitatively, whose distinction becomes prominent only at higher levels of scale and fidelity. At low scale (10^1-10^3 entities), the implications of the type of usage are not as pronounced as when the scale is increased beyond (10^4-10^9 entities). At low scale, one can resort to the modeling system that affords the highest fidelity, and be able to simulate without runtime effects becoming noticeable or problematic. At a larger scale, however, it becomes important to select the simulation system with the right tradeoff between scale and fidelity to stay within the simulation speed requirements. Using a low-fidelity simulation framework (e.g., automata-based system) can help scale to millions of entities, but the same system might be inappropriate when greater behavioral detail is attempted to be incorporated into the entities. Similarly, a high level of detail for entities in an application might not scale if most of the entities are merely used as background activity in a simulation.

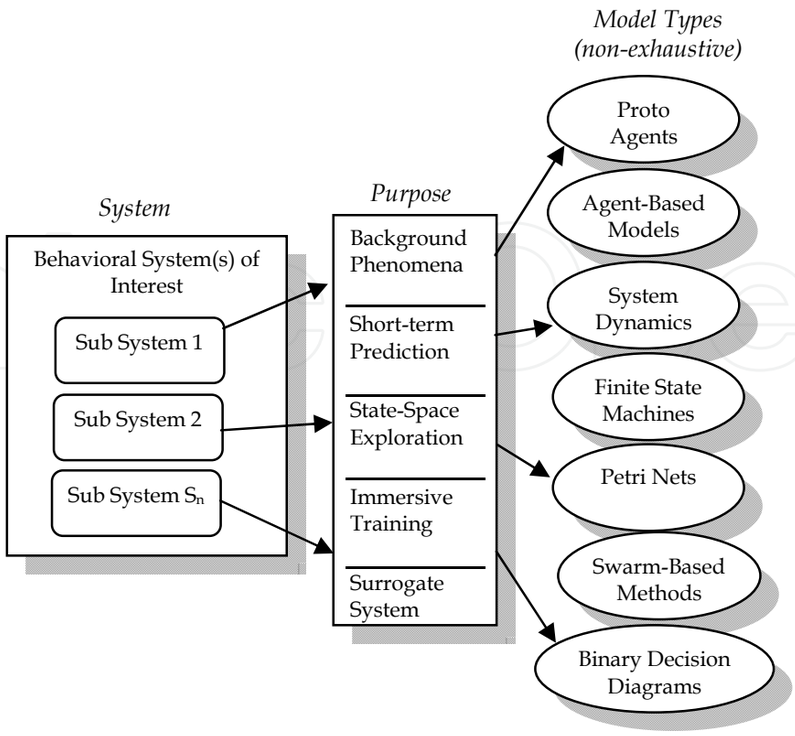


Fig. 2. Relation among Modeled System, Modeling Purpose, and Modeling Alternatives

To make an analogy, in financial market applications, simulation is used in different ways with different end goals and computational demands. Off-line simulations such as financial analytics and portfolio risk analysis are performed at relatively higher computational cost but with emphasis on longer-term prediction. Online simulations such as real-time trading solutions are a relatively recent trend in which the focus is on short-term prediction; but with as a high a fidelity as can be accommodated under real-time constraints. The important distinction is that the expectations of result quality and execution speed are set based on the intended usage of the particular type of simulation.
Here, we identify a few important classes of simulation usage in social behavioral simulations.

Situational Background Phenomena

In this class of simulations, the objective is to achieve reasonably rich, heterogeneous behavior among a large population of entities, but the emphasis is relatively less on sophistication in any one entity. Examples of this type of simulation usage are:

1. Evaluation of key social players; their behavior and influence are evaluated in the context of large low-fidelity populace (e.g., LeaderSim(Silverman 2008))
2. Animation of background crowds; need low fidelity but good heterogeneity of background entities; the main (foreground) entities, such as police force, are of much higher-fidelity (e.g., crowd simulation(Reynolds 2006))
3. In an unrelated field (Internet simulations), evaluation of distributed computing/communication applications; the network application can be evaluated

under the effects of background traffic and general network operation (e.g., fluid and packet-level simulation (Nicol, Liljenstam et al. 2003))

Short-Term Prediction

Prediction of immediate trajectory constitutes another class of simulations, in which heavy initialization is performed based on data and calibration, followed by short extrapolation via simulation. The assumption is that prediction will be more accurate due to closeness of initial state to reality and shortness of future period, together minimizing chance of deviation. Typically, this type of simulation is performed in a series of what-if case analysis to choose the best recourse from among a set of possible actions. Simulation needs are characterized by the need for extensive assimilation of configuration data and for high simulation speed for each scenario. Accuracy expectations tend to be relatively high.

Emergent Phenomena

This class of efforts is characterized by its emphasis on exploration of interesting aggregate behaviors evolved from disaggregated behaviors of an ensemble of entities. The emphasis on exploration brings the needs of long simulation times and large number of simulation runs. However, generally speaking, the level of fidelity is relatively low, since the focus is on observing the nature of overall patterns from relatively simple individual behaviors. Examples of this type of simulations include Ising Spin (Tomov, McGuigan et al. 2005), Segregation (Schelling 1978), and SugarScape models.

Training and Entertainment – Surrogate or Immersive

A class of behavioral simulations deals with training and/or entertainment objectives. For example, to train government agencies in emergency management, population reactions and responses are modeled and simulated in a synthetic environment; the environment serves as surrogate of the real world for training purposes (Chaturvedi, Foong et al. 2005). Similarly, for law enforcement and military purposes, immersive environments are used that include intelligent agents serving the role of antagonists (and sometimes allies as well) (Mastaglio and Callahan 1995; Devine and Gross 1998; Verdesca, Munro et al. 2005). The level of fidelity expected is extremely high for high quality training. Semi-automated forces fall in this category(Davis, Lucas et al. 2005). In civilian use, corporations can use such surrogate or immersive environments for management training in negotiation, team building, and so on. Again, the fidelity level expected of the automated behavior is very high, with consequent indulgence in higher-end computing platforms. Analogous needs are arising in entertainment industry, in which intelligent actors in a highly realistic synthetic environment are employed (e.g., SimCity). However, there is typically a tradeoff between fidelity and speed for real-time performance on commodity (low-end) computing platforms.

4. Computational Time Requirements

A truly enabling generation of simulation tools for computational social science would, ideally, bring progress along all five dimensions simultaneously. While progress along any single dimension is not difficult, achieving any two together is difficult, achieving three is daunting, four is heroic and all five is a grand challenge.

Among the five, of immediate interest for moving to next generation is simultaneous progress along scale and speed. In this combination, the components at play can be itemized in order to get an estimate of the computational requirements at hand. For an estimate, we will focus on agent-based view of the model, although equivalent measures can also be defined for other modeling frameworks.

The overall organization of agent-based simulation is that a series of experiments is run (e.g., with different leadership theories in an anti-establishment simulation), each experiment being executed using multiple scenarios (e.g., with different levels of displeasure among populations). Each scenario is executed multiple times, to arrive at good confidence intervals (e.g., with different random number seeds). Each simulation is typically organized as a series of time steps, each time step advancing the state of the entire model to that time step. At each time step, the rule sets of each agent are executed. A table itemizing these components of computational runtime that affect the speed of simulation is shown in Table 1.

Factor	**Notes**	**Factor-Specific**		**Cumulative**	
		Low	**High**	**Low**	**High**
Agent Rule Execution Time	From μs (for simple algebra) to ms (for complex optimization)	10^{-6} seconds	10^{-3} seconds	10^{-6} seconds	10^{-3} seconds
Number of Agents	From handful of team agents up to global world populations	10^{1} agents	10^{9} agents	10^{-5} seconds	10^{6} seconds
Number of Time Steps	Simple evolution to complex dynamics	10^{1}	10^{9}	10^{-4} seconds	10^{7} seconds
Number of Simulation Runs	Single sample run to multiple trials	10^{0}	10^{2}	10^{-4} seconds	10^{9} seconds
Number of Scenarios	From a few to full parameter sweeps or with Monte Carlo	10^{1}	10^{6}	10^{-3} seconds	10^{15} seconds
Number of Experiments	From focused contexts to cross-domain/cross-context	10^{0}	10^{2}	10^{-3} seconds	10^{17} seconds

Table 1. Range of Computational Time (on One Processor) Depending on Different Factors

4.1 Symbolic Simplicity vs. Computational Complexity

When dealing with computational complexity, it is worth noting that apparent simplicity or complexity of symbolic representation of rule sets does not necessarily have a direct correlation with computation time. A common misconception is that "simple" rules translate to "simple" computation. In particular, there is a prevalent notion that simulations aimed at discovering emergent behavior are characterized by simple computation. The fallacy in logic is the translation from simplicity of symbolic expression of rules to simplicity of computation. Here, we present two counterexamples to illustrate.

Richness of Temporal Behavior despite Symbolic Simplicity

An example of a "simple" rule set that requires long simulation times is the Manneville-Pomeau Map:

$$x_{n+1} = x_n + x_n^z (mod 1)$$

This map is useful to model turbulent, sporadic or intermittent behavior: The state variable x_n corresponding to time step n could represent a behavioral component, such as the mood of a person, or level of resentment towards oppression. Although this rule is relatively "simple" with respect to its symbolic expression, it is well-known that this map exhibits interesting behavior only when evaluated along the span of prolonged iteration. For $z \geq 3$ and $x_0=10^{-3}$, the onset of interesting behavior does not arrive until about 10^9 iterations. This type of model is an example of scenarios in which, although the dynamics are relatively "simple" to express symbolically, very long simulation times are needed to adequately exercise and capture the relevant dynamics.

Algorithmic Complexity despite Small Formulation Size

Yet another instance of simplicity of symbolic expression not equating to simplicity of computation is optimization-based rule sets. For example, Mixed Integer Programming-based model formulations are widely used in social behavioral models to capture individual optimization capabilities of interacting entities. For example, a behavioral model of people acting for/against an order from a leader is formulated as an integer programming problem in Ref(Whitmeyer 2007). These types of formulations are prone to heavy computational demands, despite simplicity of formulation. Consider the following "simple" model based on a Mixed Integer Programming formulation:

$$\max x_1 + x_2$$
$$5x_1 - 5x_2 \leq 6$$
$$5x_1 + 5x_2 \leq 9$$
$$x_1 \geq 0, x_2 \leq 5$$
$$x_1, x_2 \text{ int}$$

Solving this problem via branch-and-bound involves solving 29 instances of its linear-programming relaxations, each relaxation being solved by Simplex method. The cost thus adds up even for a simple problem, at the level of each agent. Note that the cost of this integer solving operation is incurred at every time step of every agent. At large scale, the computational cost can become prohibitive for reasonable use in a design of experiments.

4.2 Continuous vs. Discrete Event Execution – Semantics and Implementation

Another important class of computational considerations involves resolving notions of continuous and discrete time advancement mechanisms among model entities.

Most social science simulations are commonly defined in terms of time-stepped execution, in which time is advanced globally in fixed increments, and all agents are updated at each time step. However, some models are either inherently asynchronous in their formulation, or amenable to an alternative, asynchronous execution style for faster evolution. In such

asynchronous execution models, updates to agents are processed via staggered timestamps across agents. An example of such asynchrony of agent activity in the model definition itself is a civil violence model(Epstein 2002). In general, asynchronous (discrete event) execution can deliver much faster advances in simulation time than naïve time-stepped approaches. Computational consideration arises in this regard for speeding up simulations via asynchronous model execution(Perumalla 2007), both in sequential and parallel execution contexts.

In the sequential execution context (executing on one processor), a computational consideration is in automatically converting a time-stepped execution to an asynchronous (discrete event) execution, with consequential improvement in speed of simulation. Techniques for executing continuous models in a discrete event fashion are known(Nutaro 2003) and additional ones are being developed(Perumalla 2007), which could be applied to social system execution.

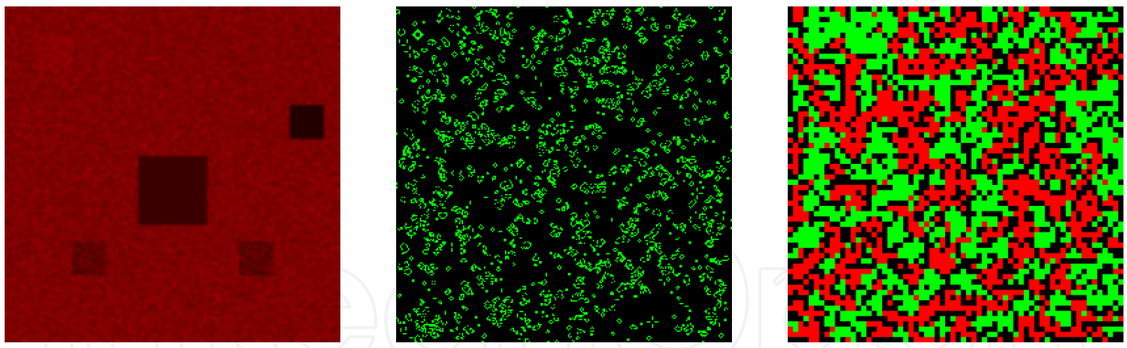
In the parallel execution context, additional complexity arises. While it is relatively straightforward to map synchronous (time-stepped) execution to parallel computing platforms, the mapping of asynchronous execution is not so obvious. Efficient execution of asynchronous activity requires much more complex handling. The fundamental issue at hand is the preservation of correct causal dependencies among agents across time instants(Fujimoto 2000). Correctness requires that agents be updated in time-stamp order.

A naïve method would be to compute the minimum time of update required among all model units (e.g., agents), and perform a parallel update of all units' states up to only that minimum update time. Clearly, only one unit (or only a very few number of units, if more than one unit has the same minimum time of update) can be processed at each iteration. The cost of computing the global minimum time can constitute a large overhead; additionally, parallel execution of all units when only one (or very few) units are eligible for update can add significant overhead. Specialized time synchronization algorithms are needed to resolve these challenges (e.g., see (Perumalla and Fujimoto 2001)).

5. Case Study: GARFIELD

Lately, the state-of-the-art is advancing the computational capabilities of social behavioral simulations. The focus has increased on improving scale, speed or both. One of the recent directions is exploiting graphics processors' data parallel execution capabilities for fast agent-based simulations. Here we present some performance results from a system called GARFIELD (Graphical Agents Reacting in a Field) for simulating agent-based models on graphics processors. Additional implementation details can be found in Ref. (Perumalla and Aaby 2008).

One of the strengths of systems like GARFIELD is the combination of large scale and high speed achievable on some low- to medium fidelity models. Fig. 3 shows graphical snapshots of a few example simulations executed under GARFIELD. Fig. 4 shows the speed differential that can be obtained by optimizing fine-grained models to GPUs, as compared to the speed of interpreter-based frameworks such as Repast and NetLogo.



(a) Mood Diffusion (b) Game of Life (c) Schelling Segregation

Fig. 3. Graphical Snapshots from Large-scale Execution of Different Behavioral Simulation Models using GARFIELD

Benefits of large-scale simulation: The Game of Life invented by John H. Conway, published in (Gardner 1970), is a simulation when executed on large grid sizes exhibits some interesting benefits of large-scale execution. In Game of Life, it is well-known that several interesting patterns emerge both spatially and temporally. With a small-scale execution (e.g., with 100×100 grids), several experiments have to be initiated in order to explore and cover many possible patterns and to exercise their dynamic behaviors. However, in an execution with a 2048×2048 grid randomly initialized with live and dead cells, a large number of patterns emerge naturally, reducing the need for instantiating a simulation run for exercising/exploring each pattern separately. It is edifying to see the patterns emerge out of evolution from a randomized initial condition. The laborious and painstaking process of cogitating about potentially interesting behaviors can be substituted by exploration via the brute force of scale.

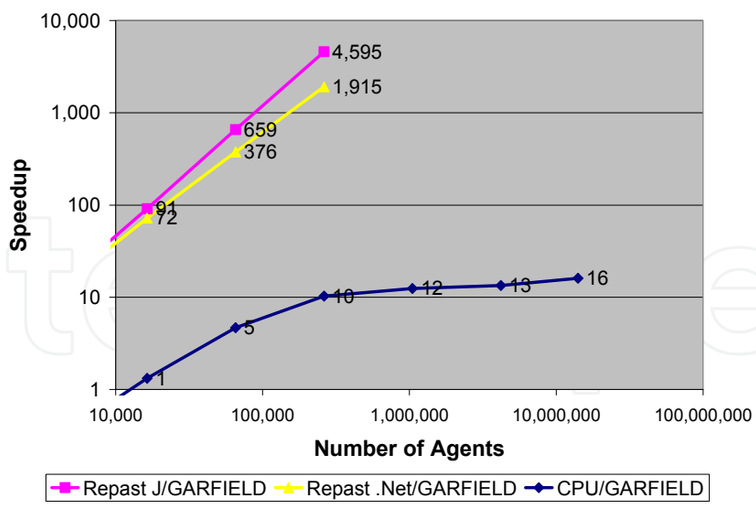


Fig. 4. Speedup of GARFIELD compared to Repast for the Game of Life Model. GARFIELD simulations are over 4000-fold faster than with traditional tool kits, and also scale to over 100-fold larger scenarios. However, usability is not as high as with other traditional toolkits.

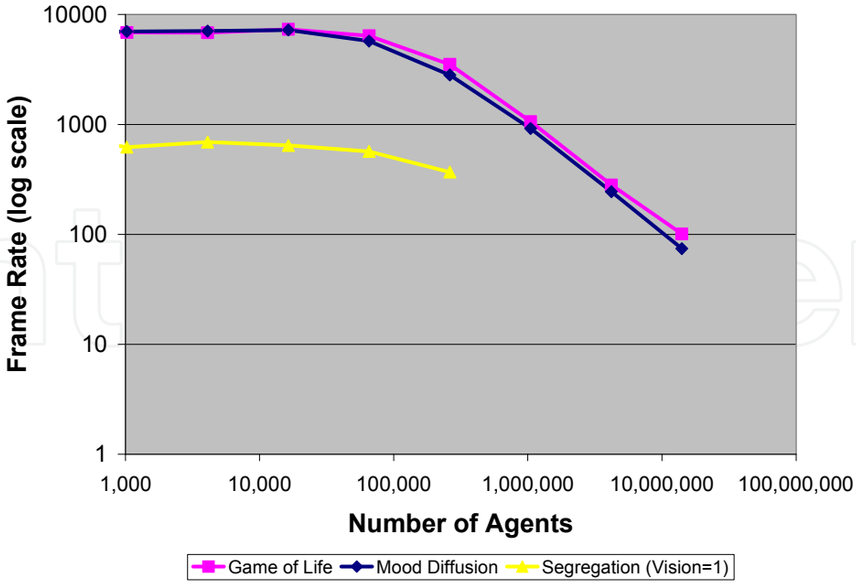


Fig. 5. Number of iterations per wall-clock second (absolute frame rate) achieved with the models on GPU

With regard to real-time aspect of speed, Fig. 5 shows the frame rate achieved for increasing number of agents in three applications. Real time execution is seen to be achievable in GARFIELD, with over 100 frames (time steps) per second clocked even in the largest grid sizes, of over 16 million agents. In smaller configurations, the frame rate is orders of magnitude higher.

5.1 Emergent Behavior with a Loyalty Model

In a more sophisticated application, it is possible to simulate populations of over several million agents with fine-grained model computation. For example, a scenario of a "leadership model" with 16 million individuals can be simulated, each individual obeying a loyalty-based leadership model that maximizes an individual's "utility" at each time step. The example model is reproduced from (Brecke and Whitmeyer 2007; Whitmeyer 2007) as follows:

$$\text{Order} \quad = O \in \{-1, 0, 1\}$$

$$\text{Behavior} = B \in \{-1, 0, 1\}$$

$$\text{Position} \ = P \in \{-1, 0, 1\}$$

$$\text{Loyalty} \ = L = \lambda \frac{|O - B|}{2}$$

$$\text{Lambda} \ = \lambda = \lambda_{previous}(1 - \delta) + M_l \delta$$

$$M_l = \text{Mean (previous) Loyalty of Neighbors}$$

$$\text{Coercion} = C = R \frac{|O - B|}{2}$$

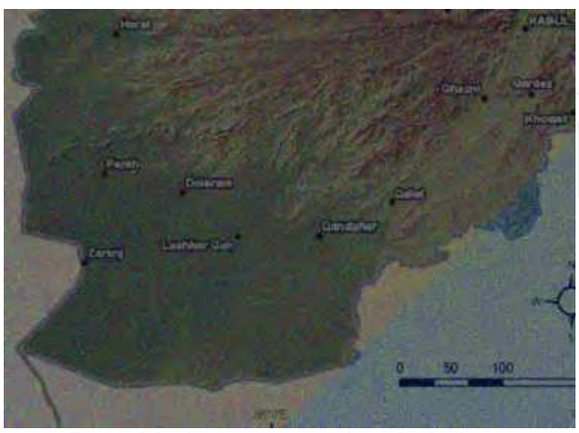
$$\text{Idealogy} = I = \frac{|P - B|}{2}$$

$$\text{Utility} \quad = U = 1 - w_l L^2 - w_c C^2 - w_i I^2$$

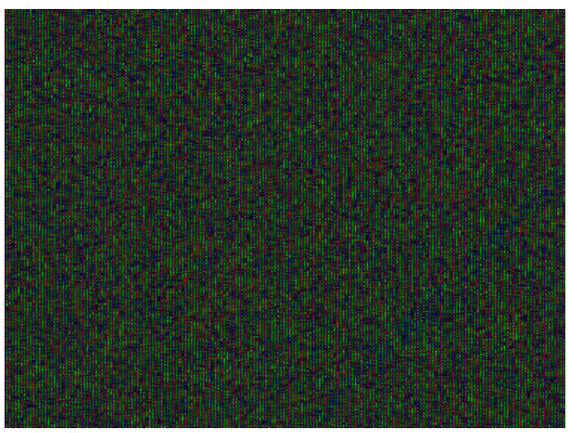
Given an order O, of interest is the variation of behavior B that is chosen by each individual to maximize the individual's utility U. Lambda's time dependence induces variation of B over time.

When M_l is defined as the mean loyalty of neighbors, the variation of B is less interesting, as lambda follows some sort of a diffusion process which can be expected to converge to an overall average across all individuals. To accommodate some dynamics, we make one change, namely, M_l is defined as the *maximum* loyalty, instead of *mean* loyalty, among neighbors. The rationale behind this variation is that the neighbor with the largest loyalty, even if there is only one, potentially has an overbearing influence on all its neighbors.

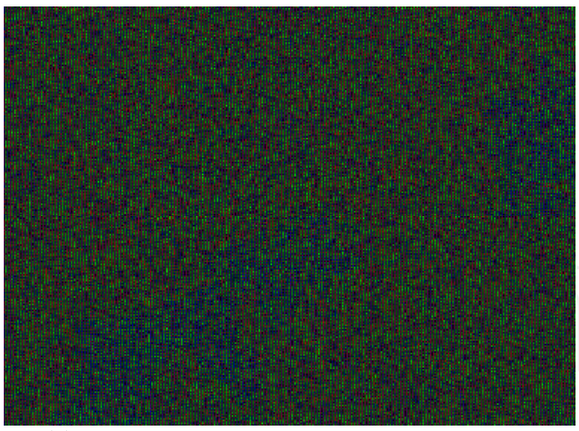
Fig. 6(a)-(f) shows snapshots of simulation for a population of over 2 million individuals, each executing the preceding loyalty model, with O=1, R=0.25, W_l=0.33, W_c=0.33, W_i=0.34, and δ=0.01. P is uniformly randomized across the population.



(a) Initial behavior map divided along a country border; loyal behaviors are below the diagonal (blue), and opposite are above

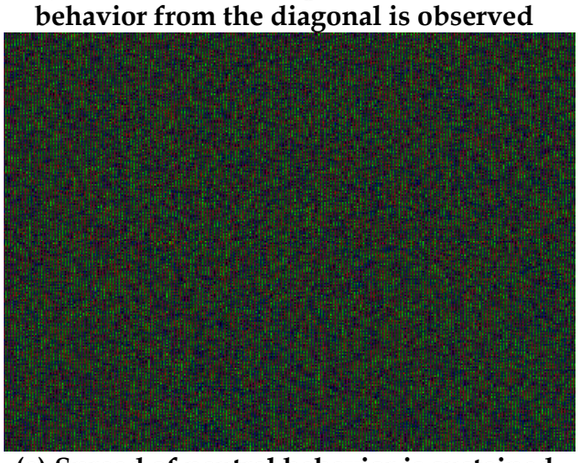


(b) Behavior smoothens after a few time steps, but neutral behaviors emerge along diagonal [geographical map is omitted]

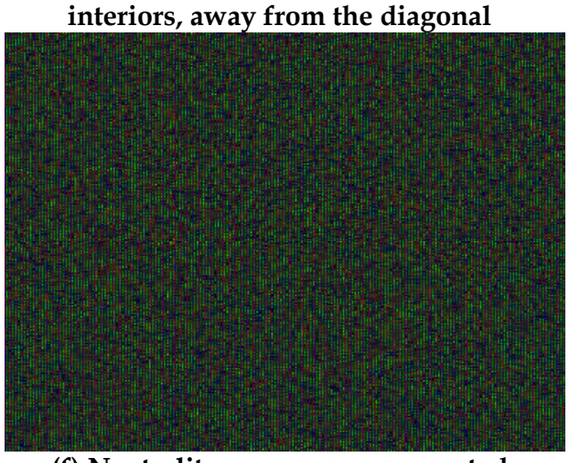


(c) Fluctuations and growth of neutral behavior from the diagonal is observed

(d) Growth smears over time into the interiors, away from the diagonal



(e) Spread of neutral behavior is sustained even after thousands of time steps



(f) Neutrality waves are regenerated despite intermediate ebbs in prior time steps

Fig. 6. Interesting spatial and temporal variation in the behavior of population under a "loyalty-based" model (Brecke and Whitmeyer 2007; Whitmeyer 2007). Blue denotes behavior loyal to leadership, green denotes neutrality and red represents anti-order stance. Sustained waves of switching from/to neutral position indicates prolonged "unrest" due to divisions in initial conditions

The high simulation speed of GARFIELD was helpful in uncovering this emergent phenomenon, which was discovered when key parameters were varied in a large number of combinations.

6. Case Study: μsik

While systems like GARFIELD are designed with scale and speed in mind for low- to medium fidelity applications, new scalable systems are needed for the general class of social behavioral simulations that can include high fidelity models, possibly enhanced by usability features. For such applications, a system that scales from traditional multi-core desktop machines to supercomputing platforms is very useful.

The parallel discrete event simulation engines, such as μsik (Perumalla 2004; Perumalla 2005; Perumalla 2007), and ROSS (Holder and Carothers 2008), execute time-stepped and discrete event simulations at very high speed on a single desktop machine for users with limited compute power. The same engines are also capable of scaling to much larger problem sizes on cluster machines, multi-core workstations, or even supercomputers. In particular, the performance of the engine on supercomputing platforms is of particular relevance, as it shows the potential of realizing extremely large-scale social behavioral model simulations at very high speeds by leveraging tens of thousands of processor cores. Recent demonstration of the possible performance shows the capability to simulate up to half a billion events per wall clock second on 16,384 processors of a Blue Gene supercomputer.

6.1 Agent Interaction Model

The performance benchmark used to demonstrate this capability is the PHOLD application, which was designed as a generalized core of most simulations that include multiple interacting entities that interact via time stamped events/messages. The PHOLD benchmark (Perumalla 2007) is an abstraction of the interaction among multiple encapsulated units, to capture the simulation dynamics of computational performance. Most interacting-entity simulations map well to this model. In a way, good performance of this model is a necessary condition for good performance of other detailed behavioral models.

In this benchmark, each unit selects a target unit at random for interaction in the future. The selection of the destination unit is made at random, with some bias towards those units that are instantiated on the same processor (more than one unit/agent is mapped to the same processor). The interaction is scheduled with a period drawn from an exponential distribution with mean 1.0, plus a minimum increment of 1.0 (i.e., a "lookahead" of 1.0).

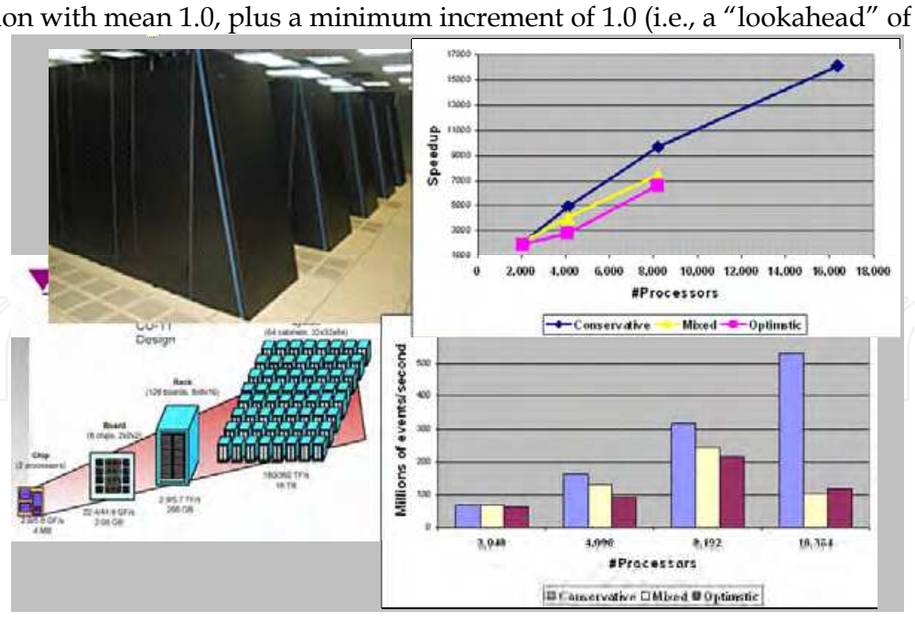


Fig. 7. Runtime Performance of μsik on the PHOLD Benchmark on up to 16,000 Cores of a Blue Gene Supercomputer.

The performance results are shown in Fig. 7, for the PHOLD scenario containing 1 million interacting entities. These results are reproduced from Ref. (Perumalla 2007). The most important metric to note is the number of events simulated per second, which translates to a per-event overhead that is on the order of 20 to 30 microseconds per event. Such a low event overhead makes it possible to contemplate executing even the finest grained agent simulations at high efficiency. In other words, the engine is capable of sustaining synchronized agent state evolution across processors with excellent parallel speedup.

7. Summary

The time seems to be ripe with respect to motivation as well as promise for next generation modeling and simulation tools in support of computational social science. Dimensions such as scale, speed, fidelity, usability and interoperability, which were once implicitly merged together at small-scale, are now getting separated as a result of focus on next levels along combinations of those dimensions. It is now possible to consider organizing the various modeling frameworks along their respective features, and select the best combination based on their fit with the specific purpose behind the simulation. The purposes behind simulations are equally important to distinguish among themselves, in order to be able to place the right levels of expectations on scale, fidelity and speed. Interoperability remains a difficult challenge, as is the problem of shielding the modeler from complexities of computational aspects driving the next generation systems. Automated compiler-based parallel execution on shared memory, LAN, GPU and/or supercomputers is potentially achievable, as is the possibility of integrating models at varying resolutions. The future looks bright for lifting computational social science to a new "enabling" plane.

8. Acknowledgements

This article has been authored by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the U.S. Department of Energy. Accordingly, the United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. This effort has been partly supported by the RealSim project at Oak Ridge National Laboratory sponsored by the Department of Homeland Security.

9. References

Bagrodia, R. and W.-T. Liao (1994). "Maisie: A Language for the Design of Efficient Discrete-Event Simulations." IEEE Transactions on Software Engineering **20**(4): 225-238.

Brecke, P. and J. Whitmeyer (2007). Data for Leadership Theory Test.

Chaturvedi, A., C. M. Foong, et al. (2005). Bridging Kinetic and Non-kinetic Interactions over Time and Space Continua. Interservice/Industry Training, Simulation and Education Conference, Orlando, FL, USA.

Cowie, J., H. Liu, et al. (1999). Towards Realistic Million-Node Internet Simulations. International Conference on Parallel and Distributed Processing Techniques and Applications.

D'Souza, R., M. Lysenko, et al. (2007). SugarScape on Steroids: Simulating Over a Million Agents at Interactive Rates. AGENT 2007 Conference on Complex Interaction and Social Emergence. Evanston, IL.

Daley, D. J. and J. Gani (2001). Epidemic Modelling: An Introduction. Cambridge, UK, Cambridge University Press.

Davis, D. M., R. F. Lucas, et al. (2005). "Joint Experimentation on Scalable Parallel Processors." Journal of the International Test and Evaluation Association.

Devine, P. and G. Gross (1998). Lessons Learned from Human-in-the-Loop Trainer HLA Implementation. Proceedings of the 1998 Interservice/Industry Training, Simulation and Education Conference. Orlando, FL.

Epstein, J. (2002). "Modeling Civil Violence: An Agent-based Computational Approach." PNAS **99**(3): 7243-7250.

Fujimoto, R. M. (2000). Parallel and Distributed Simulation Systems, Wiley Interscience.

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

Holder, A. O. and C. D. Carothers (2008). Analysis of Time Warp on a 32,768 Processor IBM Blue Gene/L Supercomputer. European Modeling and Simulation Symposium. Italy, Liophant.

Luke, S., C. Cioffi-Revilla, et al. (2004). MASON: A New Multi-Agent Simulation Toolkit. SwarmFest Workshop.

Mastaglio, T. W. and R. Callahan (1995). "A Large-Scale Complex Environment for Team Training." IEEE Computer **28**(7): 49-56.

Nicol, D., M. Liljenstam, et al. (2003). Multiscale Modeling and Simulation of Worm Effects on the Internet Routing Infrastructure. International Conference on Modeling Techniques and Tools for Computer Performance Evaluation (Performance TOOLS), Urbana, IL.

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

North, M. J. and C. M. Macal (2007). Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation, Oxford University Press.

Nutaro, J. (2003). Parallel Discrete Event Simulation with Application to Continuous Systems. Department of Electrical and Computer Engineering. Tucson, AZ, University of Arizona. **Ph.D.:** 182.

Perumalla, K., R. Fujimoto, et al. (1998). "TeD - A Language for Modeling Telecommunications Networks." Performance Evaluation Review **25**(4).

Perumalla, K. S. (2004). "µsik - Software Package Homepage." Retrieved 2004/04/01, 2004, from www.cc.gatech.edu/computing/pads/kalyan/musik.htm.

Perumalla, K. S. (2005). µsik - A Micro-Kernel for Parallel/Distributed Simulation Systems. Workshop on Principles of Advanced and Distributed Simulation, Monterey, CA, USA.

Perumalla, K. S. (2007). Model Execution. Handbook of Dynamic System Modeling, CRC Press.

Perumalla, K. S. (2007). Scaling Time Warp-based Discrete Event Execution to 10^4 Processors on the Blue Gene Supercomputer. International Conference on Computing Frontiers, Ischia, Italy.

Perumalla, K. S., Ed. (2007). Symposium on Asynchronous Methods in Scientific and Mathematical Computing (ASYM). International Workshop on Principles of Advanced and Distributed Simulation. San Diego, CA, USA, IEEE.

Perumalla, K. S. and B. Aaby (2008). Data Parallel Execution Challenges and Runtime Performance of Agent Simulations on GPUs. Agent-Directed Simulation Symposium.

Perumalla, K. S. and R. M. Fujimoto (2001). Virtual Time Synchronization over Unreliable Network Transport. Workshop on Parallel and Distributed Simulation.

Reynolds, C. (2006). "Big Fast Crowds on PS3." Retrieved 2006/09/12, 2006, from www.research.scea.com/pscrowd.

Schelling, T. (1978). Micromotives and Macrobehavior, W. W. Norton.

Silverman, B. G. (2008). "Human Behavior Model Research." Retrieved 2008/01/15, from www.seas.upenn.edu/~barryg/HBMR.html.

Staniford, S., V. Paxson, et al. (2002). How to Own the Internet in Your Spare Time. USENIX Security Symposium, San Francisco, CA.

Tomov, S., M. McGuigan, et al. (2005). "Benchmarking and Implementation of Probability-based Simulations on Programmable Graphics Cards." Computers and Graphics **29**(1).

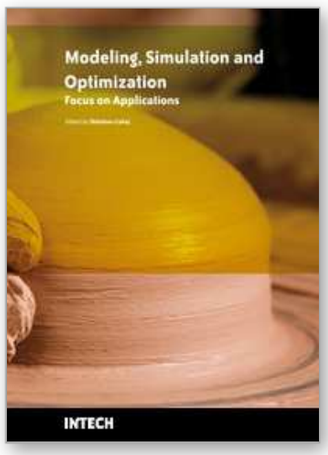
Verdesca, M., J. Munro, et al. (2005). Using Graphics Processor Units to Accelerate OneSAF: A Case Study in Technology Transition. Interservice/Industry Training, Simulation and Education Conference (IITSEC).

Walter, B., A. Sannier, et al. (2005). UAV Swarm Control: Calculating Digital Phermone Fields with the GPU. Interservice/Industry Training, Simulation and Education Conference (IITSEC), Orlando, FL.

Whitmeyer, J. (2007). Learning Theories for Loyalty-based Leadership Model.

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

Zou, C. C., L. Gao, et al. (2003). Monitoring and Early Warning for Internet Worms. ACM Conference on Computer and Communication Security (CCS), Washington, DC.



Modeling Simulation and Optimization - Focus on Applications

Edited by Shkelzen Cakaj

ISBN 978-953-307-055-1
Hard cover, 312 pages
Publisher InTech
Published online 01, March, 2010
Published in print edition March, 2010

The book presents a collection of chapters dealing with a wide selection of topics concerning different applications of modeling. It includes modeling, simulation and optimization applications in the areas of medical care systems, genetics, business, ethics and linguistics, applying very sophisticated methods. Algorithms, 3-D modeling, virtual reality, multi objective optimization, finite element methods, multi agent model simulation, system dynamics simulation, hierarchical Petri Net model and two level formalism modeling are tools and methods employed in these papers.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen