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.

UTILIZING MODEL CHARACTERISTICS TO OBTAIN EFFICIENT PARALLELIZATION IN THE CONTEXT OF AGENT BASED EPIDEMIOLOGICAL MODELS

Steven Naron

Steven Naron Associates
9909 Sorrel Avenue
Potomac, MD 20854, U.S.A.

Segev Wasserkrug

IBM Haifa Research Lab
Haifa University, Mount Carmel
Haifa, 31905, ISRAEL

ABSTRACT

There exist many problem agnostic frameworks and algorithms for parallel simulation. However, creating parallel simulation models that take advantage of characteristics specific to either the problem domain or a specific model can create significant performance benefits. This article provides an overview of general frameworks and algorithms for paralleling simulation execution, and also demonstrates two ways in which assumptions underlying the implementations of epidemiological models can be used to enable such parallelization in an efficient manner. These examples are based on planning and developing agent-based models activities carried out as part of the NIH's MIDAS (Models of Infectious Disease Agent Study) family of grants.

1 INTRODUCTION

In many cases, the computational resources required to obtain meaningful results from a simulation model are such that it is necessary to run the simulation in parallel across multiple computers. An example of such an application is a global agent based epidemiological model, which is a model intended to provide insights regarding the spreading of diseases, based on the behaviors of the individual people, and perhaps other disease vectors, that may contract and spread the disease. More details about such models appear in Section 2.

In order to enable such parallelization, there exist generic methodologies that can be used to distribute any discrete event simulation. These methodologies include general algorithms for coordinating the distributed simulation time – which is a need unique to discrete event simulation distribution. The generic distribution methodologies provide many advantages, such as correctness of the distribution mechanism, and explicitly specified distribution assumptions. These advantages are due, in a large part, to the fact that these methodologies are widely published and reviewed. Section 3 provides additional details regarding distributing discrete event simulations, as well as the existing relevant methodologies.

As with any distributed algorithms, distributed simulation models may incur a significant overhead in terms of the coordination and communication required between the computers, which may significantly impact the overall speedup obtained. In extreme cases, the implementation of the distributed model may be such that the communication overhead outweighs any performance benefits obtained by parallelizing the simulation execution. In order to achieve speedup benefits from the distribution, the implementation of the distributed simulation must take into account the characteristics of the model, including any assumptions which may be relevant to the distributed mechanism chosen. If one of the existing generic methodologies is used to implement the distribution, the characteristics of the model can be used to both choose the appropriate methodology, and to drive the specification of the implementation parameters.

In some specific models, there may be characteristics which would enable the significant reduction of the communication overhead required to distribute these models. Moreover, due to their generic nature, the general methodologies may not be able to take advantage of such characteristics. In some cases, implementing a customized distribution approach may yield significant speedup benefits over an implementation based on a generic methodology.

In this paper, we show examples of characteristics of agent based epidemiological models, which are used to efficiently distribute the simulation execution. These examples are based on planning and developing agent-based models activities carried out as part of the NIH's MIDAS (Models of Infectious Disease Agent Study) family of grants.

The rest of this paper is organized as follows: Section 2 provides background regarding agent based epidemiological models, as well as the computational resources required to execute such models. Section 3 describes the general methodologies available for discrete event simulation in general, and agent based simulation models in particular. Section 4 demonstrates how characteristics specific to agent based epidemiological models can be used to efficiently distribute the simulation. We close with a discussion and possible future work in Section 5.

2 AGENT BASED EPIDEMIOLOGICAL MODELS

There is a need for global epidemiological modeling, i.e., the modeling and understanding of characteristics relevant to the spreading of an epidemic or pandemic throughout the globe. Modeling the spreading of diseases is required to both enable policy makers to understand the hazards of a specific disease, and to determine policies which may be used to mitigate these hazards. The need for modeling the spreading of a disease at a global scale is required due to the ease by which a potential carrier of the disease may travel in today's highly interconnected world.

Epidemiological models have existed for at least several decades, e.g. Rvachev and Longini (1985). Such models have attempted to capture the following main characteristics:

1. The stages an individual may go through with regards to the disease. For example, a widely used model is the SEIR model, in which an individual may go through the following four sequential stages:
   (a) **Susceptible**: An individual may contract the disease.
   (b) **Exposed**: An individual has contracted the disease, but does not display any symptoms.
   (c) **Infected**: An individual displays symptoms of the disease.
   (d) **Recovered**: The individual has recovered from the disease.

   This model may be depicted by a sequential state chart, as shown in Figure 1. Obviously, other models based on the same states are also possible For example, an individual may go directly from the exposed to the recovered state, which enables modeling a disease for which there is the possibility that an individual has contracted the disease, but never displays symptoms.
2. The way in which a disease may be transmitted between individuals. This may take into account characteristics such as the following:
   (a) The rate by which a disease is transmitted. This can be modeled using a discrete probability.
   (b) The states of a disease in which an individual can transmit the disease to other individuals. For example, for some diseases, a disease may be contagious in the exposed state, while, for others the disease can only be contracted by coming into contact with someone who is in the infected state, and is already displaying symptoms.
3. The way in which individuals travel. This may take into account day to day frequent travel, e.g. to work, school, etc., as well as less frequent travel such as by air, ships, or international trains. This determines with whom, and how frequently, each individual may come into contact. In addition, the travel characteristics may be influenced by the disease state. For example, an individual may or may not be able to travel when in the infected state.

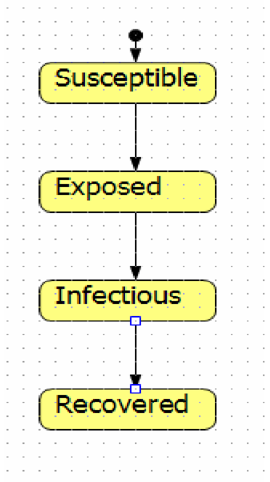

Figure 1: SEIR states model

Models such as the one in Rvachev and Longini (1985) model the spreading of diseases using aggregate behaviors. The model in Rvachev and Longini (1985) and Grais, Ellis and Grass (2003) only capture the *rates* of the characteristics relevant to the spreading of diseases. Examples of such rates include the rate by which transitions through the disease states occur, and the rates by which individual travel between cities. The progression of a disease is then calculated in an iterative manner, based on a set of differential equations. However, based on the characteristics of such models as outlined above, it seems that there may be value in modeling each individual separately. This is as such modeling may enable taking into account individual differences with respect to these characteristics. Several examples of this are the following:

1. There may be different individual reactions to the disease, resulting in different model of transition through the disease stages
2. Disease transmission behavior could differ between individuals
3. The people with which an individual comes into contact, as well as patterns of individual travels could vary widely between individuals
4. Individuals may response differently to interventions such as vaccinations.

Therefore, the results of a model based on individuals may differ substantially from the results of aggregate models, such as the ones based on rates. At the very least, such individual models could be used to test when the aggregate models capture the relevant aspects of an epidemic or a pandemic, and when such aggregate models may be sufficient to capture the relevant aspects of the real world.

Due to the above justifications for the need of individual disease modeling, several such recent efforts have been carried out, and more are underway. Such efforts model individual behavior relevant to epidemiological modeling, using a simulation paradigm known as *agent based simulation model* (Macal and North 2005). This paradigm is a special type of discrete event simulation, suitable for modeling individual behavior, as well as interactions between individuals. An example of an ongoing agent based epidemiological effort is the MIDAS initiative (<https://www.epimodels.org/midas/about.do>), which is a set of grants funded by the NIH for the agent based modeling of infections diseases.

To obtain the benefits of the additional potential accuracy provided by agent based epidemiological models requires that the individuals be modeled in sufficient detail. Therefore, the creation of global model requires fairly detailed representation of a large part (if not all) of the world's population. As a result, significant computational resources are required for the execution of such models – far beyond what can be provided by today's even most advanced single node computer. Therefore, such simulations require the ability to distribute such models across CPUs, and even across computers. This need is compounded by the fact that in the event of an actual outbreak, the characteristics of existing models may need to be modified to fit the actual outbreak, and then executed as quickly as possible in order to enable policy makers to reach decisions regarding the actual outbreak.

2.1 Architectural Considerations in Distributing Agent-based Models

Agent-based discrete simulation models represent agents (sometimes only active ones) as the modeled component of their virtual world. Rules are defined that apply to each agent. Therefore on a next scheduled event or synchronous basis these rules are reviewed by the simulation program for each agent and action is taken. This action can change the state of the agent or another agent or the virtual environment In an epidemiological model this might include passing an infection, moving between state of susceptible, exposed, infected, recovered/dead i.e., the SEIR epidemiological progression), or a spatial movement (i.e., home to work.)

Some hard problems in large agent-based from a computer point-of-view are:

1. Keeping quick access to all the information about the agents -- Typically this means the information has to be kept resident in local memory. As the numbers of agents grow, potentially into the billions, this becomes a constraint. Sometimes, depending on the question the model is trying to answer it maybe reasonable to reduce the number of agents by clumping them (i.e., an agent becomes a family versus an individual for very infectious diseases.) On the other hand, some important epidemiological problems may require the inclusion of large sets of new agents to represent disease vector.
2. The computation needed dealing with all the agents – Dealing with complex operations on billions of agents can be a daunting computational task. Different computer science techniques are often tried, such as very efficient algorithms, distributed computation (SMP or MPI), and modeling only the active agents. Some of these computations can be computationally expensive, such as calculation of spatial closeness.
3. Handling movement of agents – We have to keep the data about an agent close to the computational resources using it. If we use a cluster-based distributed approach to providing extra computation and/or memory we have to continually move the information about agents. This means moving potentially vast amounts of data between nodes in a cluster. Work will be held up while the system waits for the data to be delivered. At a cost in modeling accuracy, one approach to helping with this problem is to only update agents that need to move between computational resources on fixed schedules (synchronous agent-based models.) A hardware approach is to use low latency internode communications like Myrinet. Still, if the problem is big enough and is divided between enough nodes, even this approach eventually ends up spending so much time on data management and communications that relatively little is actually applied to solving the actual problem.
4. Many reruns and a massive total computation needs -- These, like the more general class of discrete simulation models, often have to be run many times for any interesting epidemiological experiment. This is as the models are typically stochastic in nature and often have to be run many times with different random seeds to get a distribution of results in order to obtain statistically significant outputs. Typically, these set of runs themselves have to be repeated many times in order to do parameter sweeps that allow sensi-

tivity analysis on the results. A given scenario might therefore require hundreds of runs. For that reason it might, sometimes make sense to not worry about wall clock time for any individual run, since all runs may need to be completed before human interventions. Two of the MIDAS models are not parallelized at all and are run many times on different nodes of a Linux cluster at the same time. This approach to computation goes by the humorous name of "Embarrassingly Parallel." Therefore an architecture with many parallel slow and steady parts might make sense in some modeling situation.

5. Agent based models can produce explosively large logs of agent movement. Since the large ones run as batch activities, only by saving these can detailed analysis and visualization be done after the run. Clearly this information can be limited, but that comes at a cost of the types of analysis that are possible. For example, post analysis of the EpiSims logs can tell how many from each age group infects each other age group. Post analysis of multiple run logs allows the understanding of the distribution of these results. The amount of data produced can be so vast as to add a significant amount to communications load that running the simulation produces at the same time that the computer it is running on has to handle the heavy movement of agent information.

These considerations demonstrate just some of the issues and tradeoffs in deciding when and how to distribute an epidemiological agent-based model.

3 GENERAL PARALLELIZATION FRAMEWORKS

The need for parallel execution of simulation models is a general need for large scale detailed simulation. Therefore, as mentioned in the introduction, generic methodologies for such parallel implementation and execution exist.

A discrete event simulation models a physical (real) system in terms of *events* and *states*. The execution of the simulation consists of processing events to modify the state. Each simulation also has an abstraction of the time at which events occur, known as *simulation time* or *virtual time*. Each event occurs at a particular instant in simulation time, known as the *timestamp* of the event. A single processor (sequential) discrete event based simulation therefore consists of the following:

- A set of *state variables* which collectively describe the state of the system.
- An *event list* containing those events yet to be processed
- A *global clock* denoting the current simulation time.

A sequential discrete event simulation can easily ensure that events are processed in timestamp order as it processes the event with the smallest timestamp in the event list. However, ensuring that the events are processed in timestamp order when a simulation is distributed requires coordination, and is the major source of the communication overhead existing in parallel simulation models. This is as a parallel discrete event simulation (PDES) is distributed across multiple Logical Processes (LPs). Distributing the simulation over multiple processes requires multiple event lists, one for each LP. A consequence of this is that ensuring the events are processed in timestamp order is less straightforward.

Two main types of synchronization algorithms exist to ensure that the events are processed in timestamp order: Conservative algorithms and optimistic algorithms. In conservative algorithms, each LP advances its clock only when it can be ensured that this advancement will not result in events being processed out of timestamp order. Such algorithms require defining a parameter known as a lookahead time, that is an amount of time in which each LP can advance its local clock while ensuring that no events can be received out of time stamp order. In optimistic algorithms, each LP can advance its time independently of others. In such algorithms, events may arrive out of time stamp order. To handle such cases, optimistic algorithms define rollback mechanisms.

The speedup achieved by conservative algorithms is directly proportional to the size of the lookahead time. Therefore, implementing conservative distribution algorithms for simulation models in which either the lookahead time is too small, or no lookahead times can be assumed will result in a very large communication overhead. In optimistic systems, on the other hand, the communication overhead depends on the actual event rollbacks that occur in the system. Such rollbacks are proportional to size of the differences in the simulation time of the different LPs in the system. For more details regarding optimistic and conservative algorithms, see Fujimoto (2001).

In order to attempt and overcome the shortcomings of conservative and optimistic algorithms, hybrid algorithms have been proposed. Such algorithms allow events to arrive out of time stamp order, and contain a rollback mechanism, similarly to optimistic algorithms. However, such algorithms also ensure that the difference between the simulation times of different LPs in the system does not exceed some size, thereby attempting to ensure that the number of rollbacks is not too large. For examples of such algorithms, see Fersha (1995) and Sokol and Stucky (1990).

Attempts have also been made to add to these generic frameworks features that will be suitable for general agent based simulation (i.e., without taking into account the problem domain of the simulation). In agent based simulation, it does not make sense to distribute the simulation of an existing agent across multiple LPs. This is due to the fact that one of the main aspects modeled in agent based systems is complex behavior of an agent. Therefore, distributing the simulation of this complex behavior across multiple LPs would result in a large amount of inter LP communication. Therefore, the question becomes how to distribute the agents between LPs in an efficient manner. Another important aspect of agent based simulation is the interaction between the agents. The way in which agents interact depends on the environment in which they exist. One aspect of such an environment is the geographical aspect. For example, in global pandemic models, the geographical environment is the globe, as well as the manner in which agents may travel around the globe. Therefore, a central question in distributed agent based simulation is how to distribute the environment. In Logan and Theodoropolous (2000), a suggestion has been made to separate the LPs of a distributed simulation such that there are separate LPs for agents, and separate LPs for the environment. A different type of adaptation to agent based simulation appears in Lees, Logan and Theodoropolous (2003), in which hybrid synchronization algorithms specific to agent based modeling have been proposed, based on the Sphere of Influence (SoI) of a simulation event. The SoI of an event is defined as the set of state variables read or written as a consequence of the event and depends on the type of event, the state of the source of the event (agents or environment) and the state of the environment (Lees, Logan, and Theodoropolous 2003).

As described above, an agent based simulation is distributed by breaking up the environment into logical processes. For example, in a global pandemic model, different LPs would model different cities or different groups of cities. However, due to agents moving around in their environment, any static partitioning of the environment may cause some LPs to become extremely overloaded. For an example, consider a global pandemic model in which at some stage, most of the infected agents become concentrated in a small number of cities. The LPs in which these cities are present will require considerable computational power in order to continue the simulation. It is obviously impossible to know before the simulation begins if such concentration will occur, and if so, in which cities. Dynamic load balancing mechanisms, which redistribute the environment between different LPs during the execution of a simulation can overcome this problem. In Logan and Theodoropolous (2000), such a dynamic load balancing algorithm is proposed, based on creating a conceptual tree of logical processes, upon which a distributed load balancing algorithm is defined.

In the realm of distributed simulation, the IEEE *High Level Architecture* (HLA) standard (IEEE standard 1516) has been defined. There have been implementations of distributed agent based simulation based on HLA, e.g. Lees, Logan and Theodoropolous (2002). However, HLA was designed more for allowing simulations to interoperate, rather then for efficient parallel simulation.

In order to implement parallel/distributed simulation algorithms, a parallel/distributed software infrastructure is required, providing elements such as communication between nodes. Two libraries for generic implementations of parallel/distributed algorithms are OPENMP for shared memory architectures (<http://www.openmp.org/drupal/>), and MPI (<http://en.wikipedia.org/wiki/Message_Passing_Interface>) for architectures with a distributed memory (e.g. separate computers connected by some high speed backbone). A good example a model based on such low level frameworks are the MIDAS EpiSims model lead by Stephen Eubanks at Virginia Biotechnical Institute (VBI). The model can be used to represent large urban areas (~10M person) and provides a very high level of fidelity for movements and contacts. This model distributes individuals in geographical groups to different computers and has custom designed approach based on MPI to implement movements between geographies/computers. High speed computer interconnect speeds were critical in benchmarks of EpiSims (e.g., using Myrinet or instead of Gbyte Ethernet.) For this reason, the core MIDAS computational facility was designed with high speed computer interconnectivity (Myrinet.)

As both MPI and OPENMP are low level frameworks, there have been attempts made to abstract away the details of creating parallel/distributed algorithms. Examples of this are the OptimalGrid framework, (<http://www.alphaworks.ibm.com/tech/optimalgrid>), and the Distributed Computing Toolbox from Mathworks (<http://www.mathworks.com/products/distribtb/index.html>). The OptimalGrid framwork is a generic framework for distributed computing, that abstracts away the communication layer and provides automatic load balancing capabilities. The Distributed Computing Toolbox from Mathworks provides some abstraction for parallel computation when using the Matlab tool.

4 PROBLEM SPECIFIC CHARACTERISTICS OF GLOBAL EPIDEMIC MODELLING

As discussed in Section 2, an agent based simulation of epidemics needs to take into account the following factors: The stage an individual goes through regarding the epidemic, the way the epidemic is transmitted between individuals, and the manner the individual travels around a globe.

An assumption that is made in many existing models is that an individual travels much more frequently within a close geographical proximity then within a larger one. For example, an individual is much more likely to remain in the same city then to travel to cities which are in another continent. This assumption seems reasonable in today's world. This assumption seems to indicate that a good way to distribute these types of simulation would be according to geographic proximity. In such cases, an LP would tend to simulate geographic locations which are close together in the sense relevant to the model. For example, a single LP could simulate suburbs within a city that are close together, or cities which are geographically close or have a direct airline connection.

The question that then arises is what would occur if an agent travels between geographical locations that are simulated by different LPs. In order to deal with this case, use is made of another assumption that is fairly common in these types of models. This assumption is that it in the model, agents can be constrained to pass between certain types of geographical locations **only** at certain points in time (rather than continuously), without impacting the validity of the simulation results. For example, in the model appearing in Rvachev and Longini (1985), it is assumed that it is sufficient to model the traveling of agents between cities as if they occur only at the end of each day. In such a case, if different LPs simulate different cities, each LP can simulate an entire day, and still no events will arrive out of time stamp order. This is an example of discrete (or batched) vs. continuous time. Note that the model in Rvachev and Longini (1985) is equation, rather than agent-based, and never was distributed across multiple computers All agent based models explicitly or implicitly assume a certain time accuracy.

A good example of modeling assumptions made to improve performance to allow better scalability are those made by Jon Parker at Brookings as part of MIDAS. Both assumptions reduce the necessary frequency of synchronization. Communications between computers typically has high latency (bad) and throughput (good). Therefore one wants to greatly reduce the frequency of synchronizations.

First he tried out a default time precision of about 14 minutes. (one 1/100th of a day). Any events by all agents during this time are treated as simultaneous. This assumption is equivalent to saying that we are indifferent to the order of processing of events within this time block. This allows him to reduce synchronization to this time period (or multiples.) Importantly, Jon parameterized this time period to allow him to validate this assumption with increasing smaller time slices. For example, he could run the occasional model at 1/1000th of a day and see if the results varied.

As a second approach to limiting the number of synchronizations, Jon moved the synchronization period to just less than the disease's incubation period (infected until infectious.) This is typically one or more days. This approach greatly reduced the total number of synchronizations during a simulated epidemic.

This turns out to be a "sweet spot" for epidemic models. In the worse case, if a person is infected and wait until just before becoming infectious to travel between geographies (agent moved between computers) they can, at most only pass on the disease to one more generation. Since this synchronization period is much longer than the time precision, events can get out of order between synchronizations. But, since these can only be one generation of events, backing out the effects is computationally simple and we do not have geometric growth of effected agents.

Through these two assumptions, the first testable and the second which has no effect on model results, he greatly reduced the number of necessary synchronizations during the run of a simulated pandemic. A number of architectural approaches now become relatively viable for running vast agent-based models such as:

1. Clusters with simple and cheap Ethernet interconnections
2. Grid computing, perhaps even including "at home" shared volunteered PCs over the internet
3. Single PC operations where each spatial area is run for a given synchronization period and then swapped with the next spatial area.

As is common to agent based models, dynamic load balancing may also be required for such epidemic modeling, (e.g., when the simulation develops in a way such that most of the infected people are concentrated in a small number of cities.) Separating the simulation according to geographical regions provides the basis for such load balancing, as geographic regions (e.g. cities) may dynamically migrate between LPs, based on the load.

Note that the above assumptions enabled the creation of a parallel simulation of epidemic models, using the assumptions of geographic distribution and inter city travel time, without having to implement a generic synchronization algorithm.

5 SUMMARY AND FUTURE WORK

In this paper, we have described the need for parallelizing simulation execution. We have demonstrated this need in the context of the modeling of global epidemics. We've provided an overview of general mechanisms for enabling parallel/distributed simulation. Finally, we have shown how using the characteristics and assumptions relevant to specific domains and models can be used to implement a parallel simulation without the use of a generic simulation distribution framework.

Efficiently distributing a simulation, even in a specific domain, entails intelligently utilizing modeling assumptions that can be made for specific models. Therefore, depending on the specific assumptions, it may be possible to define patterns (or anti patterns) for efficiently distributing a simulation. To define such patterns requires working closely with as many teams as possible teams who are creating different related models, and sharing knowledge regarding the model itself and the distribution mechanisms. The quality of the patterns/anti patterns created will be in direct proportion to such sharing, and the various types of models (in a specific domain) which are distributed.

REFERENCES

Fersha, A. 1995. Probabilistic adaptive direct optimism control in time warp. In *Proceedings of the 9th Workshop on Parallel and Distributed Simulation*, 120-129.

Fujimoto, R.M. 2001. *Parallel and Distributed Simulation*. Wiley Interscience, Wiley-Interscience

Grais, R. F., J. H. Ellis, and G. E. Glass. 2003. Assessing the Impact of Airline Travel on the Geographic Spread of Pandemic Influenza. *European Journal of Epidemiology* 18: 1065–1072.

Macal, C. M., and M. J. North. 2005. Tutorial on Agent Based Modeling and Simulation. In *Proceedings of the 2005 Winter Simulation Conference*, ed. M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, 2-15. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Lees, M., B. Logan, and G. Theodoropoulos. 2003. Adaptive Optimistic Synchronisation for Multi-Agent Simulation. In *Proceedings of the 17th European Simulation Multiconference*, 77-82.

Lees, M., B. Logan, and G. Theodoropoulos. 2002. Simulating Agent-based Systems with HLA: The Case of Sim_Agent. *European Simulation Interoperability Workshop*, 285-293.

Logan, B., and G. Theodoropolous. 2000. The Distributed Simulation of Multi-Agent Systems. *Proceedings of the IEEE* 89(2):174-185.

Rvachev, R. A., and I. M. Longini Jr. 1985. A Mathematical Model for the Global Spread of Influenza. *Mathematical Biosciences* 75:3-22.

Sokol, L., and B. Stucky. 1990. Mtw: Experimental results for a constrained optimistic scheduling paradigm. In *Proceedings SCS Multiconf. Distributed Simulation*, 169-173.

AUTHOR BIOGRAPHIES

Steven Naron was for over two decades an analyst, solutions architect and project manager for IBM with patents, invention disclosures, and published papers related to organizational and informational system optimization. Steve has a BS in Operations Research from Johns Hopkins University, a MS in Computer Science from University of Maryland, and a MBA focused on operations analysis from American University. He has his PMP certification from PMI. For three years Steve was the lead of the IBM team that supported the NIGMS MIDAS family of grants. He now runs his own consulting organization.

Segev Wasserkrug is a research staff member in IBM Haifa Labs. Segev has both a B.A. and an M.Sc. Degrees in computer science from the Technion – Israel Institute of Technology. Segev has practical and academic background in the areas of Operations Research (including optimization, workforce predictive models, and simulation), computer science, and machine learning. Segev has over thirteen years practical experience, including six years in applying analytics to solving customer problems, specifically in the area of workforce management and high performance simulation.