# Dynamic Data Driven Simulation

**Xiaolin Hu**
Department of Computer Science
Georgia State University
Atlanta, GA 30303
xhu@cs.gsu.edu

**Keywords**: *Dynamic Data Driven Simulation, Data Assimilation, Wildfire Spread Simulation*

**Abstract**

This article presents dynamic data driven simulation as a new simulation paradigm where a simulation system is continually influenced by real time data for better analysis and prediction of a system under study. This is different from traditional simulations that are largely decoupled from real systems by making little usage of real time data. We present a framework of dynamic data driven simulation based on sequential Monte Carlo methods, describe the opportunities and challenges associated with dynamic data driven simulation, and show an illustrative example.

## 1. Introduction

Computer simulations have long been used for studying and predicting behaviors of complex systems, such as wildfires, urban traffic, and infectious disease spread. The accuracy of these simulations depends on many factors, including data used in the simulations and fidelity of the simulation models. Considering wildfire spread simulation as an example, the simulation relies on the terrain data, vegetation data, and weather data in the wildfire area. Due to the dynamic and stochastic nature of wildfire, it is impractical to obtain all these data with no error. For example, the weather data used in simulation is typically obtained from local weather stations in a time-based manner (e.g., every 10 minutes). Before the next data arrives, the weather is considered unchanged in the simulation model. This is different from the reality where the real weather constantly changes (e.g., due to the mutual influences between wildfires and the weather). Besides data errors, the wildfire behavior model introduces errors too because of its computational abstraction. Due to these errors, the predictions from the simulation model will almost certainly be different from what is in a real wildfire. Without assimilating data from the real wildfire and dynamically adjusting the simulation model, the difference between simulation and real wildfire is likely to continuously grow.

Incorporating real time data into a running simulation model has the potential to greatly improve simulation results. Unfortunately, until recently this line of work did not receive significant research attention in the simulation field. While sophisticated simulation models have been developed, traditional simulations are largely decoupled from real systems by making little usage of real time data from the systems under study. With recent advances in sensor and network technologies, the availability and fidelity of such real time data have greatly increased. As a result, a new paradigm of *dynamic data driven simulation* is emerging where a simulation system is continually influenced by the real time data streams for better analysis and prediction of a system under study.

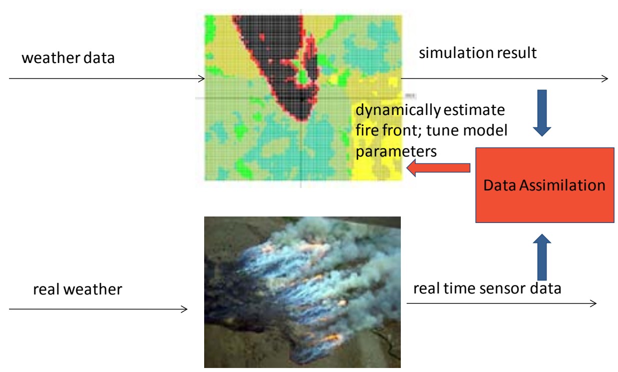



Figure 1: Dynamic Data Driven Simulation for Wildfire Spread Prediction

Figure 1 illustrates the idea of dynamic data driven simulation based on the application of wildfire spread simulation. In the figure, the top part represents the wildfire simulation model; the bottom part represents the real wildfire. As the wildfire spreads, streams of real time data collected from fire sensors deployed in the fire area are transmitted to and assimilated by the data assimilation component. This allows the simulation system to continuously adjust itself in order to achieve more accurate predictions of wildfire spread. By coupling a simulation system with real time data, dynamic data driven simulation can greatly increase the power of simulation-based study. In the wildfire example, based on the real time sensor data of a wildfire, a wildfire simulation can better estimate the current fire front and fire intensity and thus providing more accurate fire spread predictions. The sensor data also carries "feedback" information for a simulation to calibrate its model parameters to reduce discrepancies between its simulation results and the observations. These capabilities are very useful for supporting real time decision makings of wildfire containment.

The work of dynamic data driven simulation is closely related to *Dynamic Data-Driven Application System (DDDAS)*, which entails the ability to incorporate dynamically data into an executing application simulation, and in reverse, the ability of applications to dynamically steer measurement processes [1]. DDDAS advocates a conceptual framework including bi-directional influence between the application simulation and the measurement system. Dynamic data driven simulation is also related to the work of *data assimilation* by incorporating data into a running model. Data assimilation is an analysis technique, in which the observed data is assimilated into the model to produce a time sequence of estimated system states [2]. It has achieved significant success in fields such as oil and gas pipeline models and atmospheric, climate, and ocean modeling. Important estimation techniques used in data assimilation include Kalman Filter and its variance [3, 4]. Conventional estimation techniques, however, cannot be effectively used in supporting dynamic data driven simulation due to the non-parametric representation and the non-linear, non-Gaussian behavior of many simulation models. In this article, we employ *Sequential Monte Carlo (SMC) methods* to assimilate real time sensor data into the simulation model for dynamic data driven simulation. A framework of dynamic data driven simulation based on SMC methods is presented in Section 2. We develop this work based on the application of wildfire spread simulation using the DEVS-FIRE model [5, 6]. It is important to note that the general concept of dynamic data driven simulation and the developed framework are not dependent on the wildfire example and can be generalized to other applications.

## 2. A Dynamic Data Driven Simulation Framework Based on Sequential Monte Carlo Methods

The new paradigm of dynamic data driven simulation asks for new approaches for incorporating real time data in an effective and efficient manner. Assimilating data for large-scale simulation models is a challenging task due to the large number of possible state variables and model parameters and the complex and dynamic system behavior. For example, to assimilate data into a wildfire spread simulation, an important complexity is associated with the non-stationary, non-linear, non-Gaussian behavior of wildfire [7], which makes it ineffective to use conventional inference techniques such as Kalman filter and its variants (e.g., ensemble Kalman filter [4]). Furthermore, simulation models usually have complex model structures and state update mechanisms. Considering the DEVS-FIRE model as an example, it is a DEVS-based model with discrete event simulation mechanism and a cellular space model structure. This makes it difficult to apply inference techniques that rely on equation-based model representations. To overcome these challenges, we choose SMC methods as the data assimilation algorithm for supporting dynamic data driven simulation.

SMC methods, also called particle filters, are a set of sample-based methods that use Bayesian inference and stochastic sampling techniques to recursively estimate the states of dynamic systems from some given observations [8, 9]. They approximate the sequence of probability distributions of interest using a large set of random samples, named particles, and assign importance weights to these particles based on the observation data feedback. The particles are propagated over time using Importance Sampling (IS) and resampling mechanisms. It has been shown that the large number of particles are able to converge to the true posterior even in non-

Gaussian, non-linear dynamic systems [10]. For systems with strongly nonlinear behavior, SMC methods thus are more effective than the wildly used Kalman filter and its various extensions. A key advantage of SMC methods is their ability to represent arbitrary probability densities and to have little or no assumption about the structure of the system model. This makes it an effective method for supporting dynamic data driven simulation with sophisticated simulation models. Meanwhile, SMC methods are recursive methods that are able to recursively adjust their estimations of system states when new observation data becomes available. This feature is suited for dynamic data driven simulation where new sensor data arrives sequentially and the simulation system needs to be continuously updated. More details about SMC methods can be found in [8].

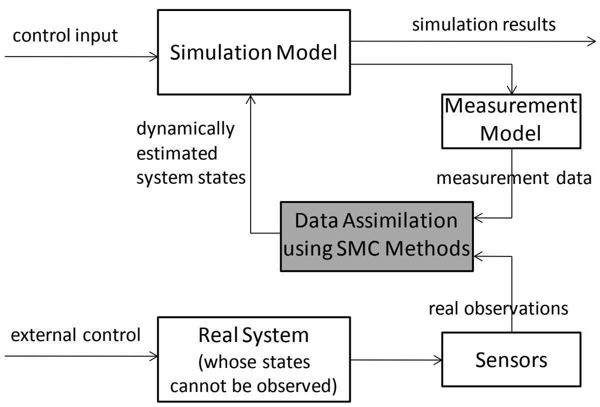



Figure 2: Dynamic Data Driven Simulation Based on SMC Methods

A dynamic data driven simulation framework based on SMC methods is shown in Figure 2. Major components of the framework include a *real system* under study, a *simulation model* that simulates the dynamic behavior of the real system. Typically, the real system's states, which change over time, cannot be directly observed and is unknown to the simulation model. This usually makes the simulation start from a state different from the state of the real system, leading to inaccurate simulation results. A major task in dynamic data driven simulation is to dynamically estimate the "current" state of the real system and then feed the estimated states to the simulation model for follow-on simulations. This is achieved through *data assimilation* based on SMC methods, which assimilate real time sensor data for better inference of the "current" system state. To support dynamic data driven simulation, *sensors* are deployed and real time sensor data (referred to as *real observations* in Figure 2) are collected. Meanwhile, a *measurement model* is developed to model the real sensors. This measurement model computes the *measurement data* from the simulated system state generated from the simulation model. Both the real observations collected from real sensors and the measurement data computed from the measurement model are utilized by the SMC method-based data assimilation component in estimating the real system's state.

To carry out data assimilation, a SMC algorithm goes through multiple iterations in a recursive manner. Here we provide an overview of a basic SMC algorithm that implements the sequential importance sampling with resampling (SISR) procedure, with the proposal function chosen to be the state evolution function defined by the simulation model. More rigorous description and corresponding mathematical notations of the SMC algorithm can be found in [8]. In each iteration, the algorithm receives a sample set representing the previous belief of the system state, and a real observation collected from the sensors. Each sample in the sample set represents a specific system state. In the importance sampling step, each sample is used to predict the next state through the state evolution model that represents the system dynamics (which is the simulation model in our work). Based on the new states, the measurement data of each sample is computed using the measurement model. Then the importance weight of each sample is computed based on the difference between the real observation and the measurement data. Finally, in the resampling step, a new set of offspring samples are drawn with probability proportional to the normalized sample weights. These samples represent the posterior belief of the system state and are used for the next iteration. To apply this algorithm to data assimilation for a specific simulation application, the problem needs to be properly formulated according to the algorithmic structure of SMC methods. Section 4 provides an example based on the application of wildfire spread simulation.

## 3. Opportunities and Challenges of Dynamic Data Driven Simulation

The dynamic data driven simulation represents a new paradigm where a running simulation system is coupled with the real system by assimilating real time sensor data. This new paradigm is enabled by several technological advances in recent years. First, simulation plays increasingly important roles in supporting real time decision-makings, and a large number of sophisticated simulation models have been developed in a wide range of application areas. This creates a strong need for a simulation system to dynamically assimilate real time data to better support real time decision making. Second, with recent advances in sensor technology and sensor network, the availability and fidelity of real time sensor data have greatly increased. Wired or wireless communication networks make real time sensor data readily available for dynamic data driven simulation. Finally, advances in high performance computing technologies make it possible to apply advanced inference techniques such as SMC methods that can effectively work with simulation models with complex model structure and non-linear, non-Gaussian behavior.

As a new paradigm, dynamic data driven simulation brings new opportunities to improve the results of simulations. Based on the wildfire spread simulation example, below we list several functions that dynamic data driven simulation enables.

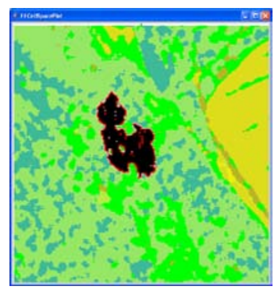
- Dynamic state estimation: Estimating the dynamically changing system states from observation data is a fundamental task in dynamic data driven simulation. The state estimation from real time data allows a simulation to start from a state "closer" to the real system's state, and thus leads to more accurate simulation results.
- Online model parameter calibration: Besides estimating state variables that represent system behavior, it is also desirable to dynamically calibrate the model parameters that characterize the system structure. For example, due to the complexity of wildfires, a wildfire model needs to be calibrated in order to "fit" a specific forest area [11]. One can formulate the problem of online model calibration as a joint state-parameter estimation and uncertainty assessment problem, which treats model parameters as stochastic state variables that need to be estimated.
- Dynamic data driven event reconstruction: We define dynamic data driven event reconstruction as the process of estimating the occurrences and characteristics (e.g., when, where) of some events in interest. Such events are not explicitly modeled by the simulation model but can significantly affect the system behavior. For example, while a wildfire is spreading, new fires may be ignited in the vicinity of fire front, resulting in multiple fires. Being able to estimate the dynamic occurrences of such new fires from real time sensor data can greatly improve fire spread simulation results.

With new opportunities, dynamic data driven simulation also brings new research challenges. These challenges are mainly due to the large state spaces associated with sophisticated simulation models and the demanding computation cost required by SMC methods. The large state space poses challenging issues for effective inference of system states as well as convergence of the SMC methods. This asks for advanced sampling and resampling methods in order for the SMC methods to achieve effective inference and quick convergence. Another important challenge is associated with the high computation cost of applying SMC methods with sophisticated simulation models. SMC methods have a demanding computation cost because of the large set of particles, each of which needs a full scale simulation to evolve to the next system state. This issue of computation cost is manifest especially for large-scale spatial temporal simulations such as simulations of wildfire. Developing advanced methods to reduce the demanding computation cost is crucial for supporting real time decision making in time critical situations. Research challenges also come from how sensor data are collected, which is an essential part of data assimilation. In general, more and higher quality sensors are needed in order to collect higher quality sensor data to be used in data assimilation. However, this will raise the cost of sensor deployment and also increase the computation needs. An important research task is to study how to deploy sensors in an effective manner and how to extract useful information from sensor data for data assimilation. For example, in the wildfire example, an effective sensor deployment strategy is to deploy more sensors in the areas with higher fire risks and/or close to the active fire propagation areas.

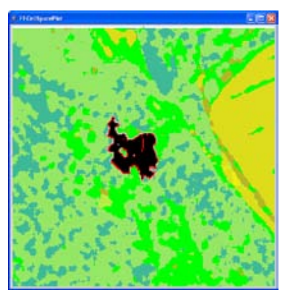
## 4. An Illustrative Example – Dynamic Data Driven Simulation for Wildfire Spread Simulation

We present an example to illustrate how dynamic data driven simulation works based on the application of wildfire spread simulation. This example is adapted from experiments described in [12]. In this example, the wildfire spread simulation model is a discrete event simulation model called DEVS-FIRE [5, 6]. DEVS-FIRE is a two dimensional cellular space model where the forest is modeled as a two-dimensional cell space. The cell space contains individual forest cells each of which contains its own GIS data and weather data. Each cell in the cell space is represented as a DEVS atomic model and is coupled with its eight neighbor cells according to the Moore neighborhood. Consequently, the forest cell space is a coupled model composed of multiple forest cell models. Fire spread is simulated as a propagation process where burning cells ignite their unburned neighbors. The rate of spread of a burning cell is calculated using Rothermel's fire behavior model [13] and then decomposed into eight directions corresponding to the eight neighboring cells. Detailed description of the DEVS-FIRE model can be found in [5, 6].

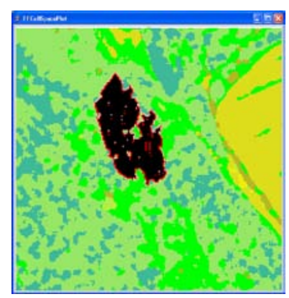
We use the identical-twin experiment, which is widely used in data assimilation research, to show how the data assimilation works and its results. The purpose of identical-twin experiments is to study data assimilation in ideal situations and evaluate the proximity of the prediction to the true states in a controlled manner. In the identical-twin experiment, a simulation is first run, and the corresponding data is recorded. These simulation results are considered as "true"; therefore, the observation data (real time sensor data) obtained here is regarded as the real observation data (because they come from the "true" model). Consequently, we estimate the system states from the observation data using SMC methods, and then check whether these estimated results are close to the "true" simulation results. In the following description, we use three terms: "real" fire, filtered fire, and simulated fire, to help us to present the experimental results. A real fire is the simulation from which the real observation data is obtained. A simulated fire is the simulation based on some "error" data ("error" in the sense that the data are different from those used in the real fire), for example, imprecise weather data. This is to represent the fact that wildfire simulations usually rely on imperfect data as compared to real wildfires. Finally, a filtered fire is the data assimilation-enhanced simulation based on the same "error" data as in the simulated fire. In our experiments, we intend to show a filtered fire gives more accurate simulation results by assimilating sensor data from the real fire even it still uses the "error" data.



(a) real fire



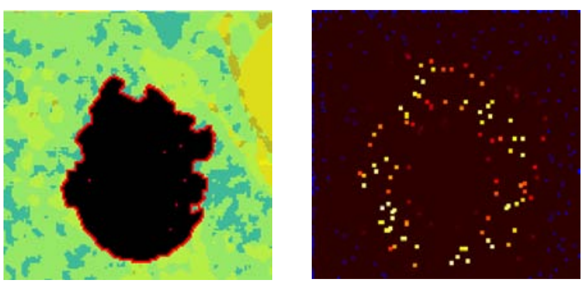
(b) simulated fire 1: wind speed is smaller than real wind speed



(c) simulated fire 2: wind speed is larger than real wind speed

Figure 3: Inaccurate fire spread simulations due to imprecise wind speed

The differences between a real fire and a simulated fire are due to the imprecise data such as wind speed, wind direction, GIS data, and fuel model, used in the simulation. In this example, we choose to use the imprecise wind speeds as the "error" data. Specifically, the real wind speed and direction are 8 (m s-1) and 180 degrees (from south to north) with random variances added every 10 minutes. The variances for the wind speeds are in the range of –2 to 2 (m s-1), and the variances for the wind direction are in the range of -20 to 20 (degrees). We introduce errors to the wind speeds and make the wind directions to be exactly the same as the real wind directions. Two simulations are carried out. In the first simulation the wind speed is randomly generated based on 6 (m s-1) with variances added in the range of –2 to 2 (m s-1). In the second simulation the wind speed is randomly generated based on 10 (m s-1) with variances added in the range of –2 to 2 (m s-1). Figure 3(a) displays the real fire after 3 hours of simulation. Figure 3(b) and 3(c) show the two simulated fires for the same simulation duration (3 hours). In the figures, the burning cells and the burned cells are displayed in red and black respectively. The other colors display different fuel types of the cells. From the figures we know that the real fire and the simulated fires have large deviations due to the imprecise wind speeds. In the first simulation, the real fire spreads fasters than the simulated fire because the real wind speeds are larger than the error wind speeds. In the second simulation, the real fire grows slower than the simulated fire since the real wind speeds are smaller than the error wind speeds.

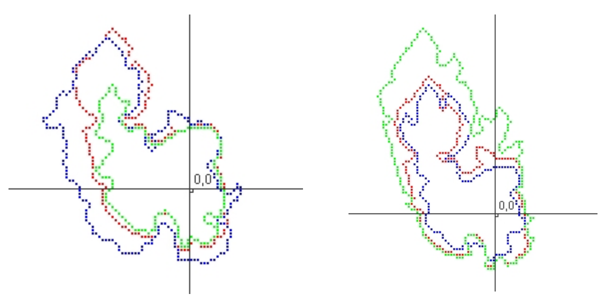


(a) Fire front of the real fire (b) Temperature sensor data

Figure 4: Fire front and temperature sensor data

Ground temperature sensors are randomly deployed in the fire area. These sensors collect new temperature data every 20 minutes to be assimilated by the data assimilation component. In this example, the sensor data are generated based on the real fire described above. A measurement model is used to compute a sensor's temperature data based on the fire intensity and the distance between the sensor and the fire front. Figure 4(a) shows a snapshot of the real fire's fire front during the simulation. Figure 4(b) shows the temperature sensor data collected by the temperature sensors, with different colors indicating different temperature values. These data are assimilated by the SMC method for estimating the "real" fire front. In Figure 4(b), the blue dots in the outside area represent the locations of the ground temperature sensors. Note that as the fire front evolves, the collected temperature sensor data change over time too.

For both the simulations where imprecise wind speeds were used as shown in Figure 3, we assimilate the temperature sensor data in order to improve their simulation results. The data assimilation algorithm is based on SMC methods. In this example, we define the system state being the evolving fire front, and dynamically estimate the real fire's fire front when new sensor data arrive. Detailed description of how the data assimilation works can be found in [12]. We used 210 particles in the experiments, and assimilated real time sensor data every 20 minutes. The simulation results using data assimilation are called the filtered fires. Figure 5(a) and 5(b) display the filtered fires (displayed in blue) after 3 hours of simulation, compared with the real fire (displayed in red), and the simulated fires (displayed in green). Figure 5 shows that in both cases, the filtered fire shapes match the real fire shape better than the simulated fires do. This is particularly true at the head area (the north direction) of the fire where the fire spreads fast. For example, in Figure 5(a), the simulated fire is much smaller than the real fire at the head area due to the smaller wind speeds. However, using the same "error" wind speeds as the simulated fire did, through data assimilation the filtered fire was able to match the real fire well at the head area. Similar effect can be seen for the second case as shown in Figure 5(b). This demonstrates the effectiveness of the data assimilation method.




(a) data assimilation result for the fire spread simulation using smaller wind speed

(b) data assimilation result for the fire spread simulation using larger wind speed

Figure 5: Data assimilation results by assimilating temperature sensor data

We note that the current data assimilation method still has a lot room for improvement. For example, in Figure 5(a) although the filtered fire matches well with the real fire at the head area, it spreads faster than the real fire does on the west side. This asks for further development of the data assimilation method for improving the data assimilation results in future work.

## 5. Conclusions

This article presents dynamic data driven simulation as a new simulation paradigm where a simulation system is continually influenced by the real time data streams for better analysis and prediction of a system under study. A framework of dynamic data driven simulation based on sequential Monte Carlo methods is presented. We describe the opportunities and challenges associated with dynamic data driven simulation, and provide an illustrative example of dynamic data driven simulation based on the application of wildfire spread simulation.

## 6. Acknowledgement

This research is supported by a NSF CAREER grant (CNS-0841170) and a NSF Cyber-Enabled Discovery and Innovation (CDI) grant (CNS-0941432) from the National Science Foundation.

## References:

1. Darema, F., 2004. Dynamic Data Driven Applications Systems: A New Paradigm for Application Simulations and Measurements. *International Conference on Computational Science.*: pp. 662–669.
2. BOUTTIER, F. AND COURTIER, P. 1999. Data assimilation concepts and methods. *Training course notes of ECMWF*.
3. Kalman, R.E., 1960. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering* 82 (1): 35–45
4. Evensen, G., 2007. *Data Assimilation. The Ensemble Kalman Filter*. Springer.
5. HU, X., SUN, Y., AND NTAIMO, L. 2010. DEVS-FIRE: Design and formal specification of discrete event wildfire spread and suppression models. *SIMULATION*, submitted.
6. NTAIMO, L., HU, X., AND SUN, Y. 2008. DEVS-FIRE: Towards an integrated simulation environment for surface wildfire spread and containment. *SIMULATION: Transactions of The Society for Modeling and Simulation International 84*, 4, 137-155.
7. MANDEL, J., BENNETHUM, L.S., BEEZLEY, J.D., COEN, J.L., DOUGLAS, C.C., KIM, M., AND VODACEK, A. 2008. A wildland fire model with data assimilation. *Math. Comput. Simul. 79*, 3, 584-606.
8. Doucet, A., N. D. Freitas, N. Gordon (eds.). 2001. *Sequential Monte Carlo methods in practice*, Springer
9. Arulampalam, M., Maskell, S., Gordon, N. and Clapp, T. 2002. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking, *IEEE Transactions on Signal Processing*, 50, pages 174-188.
10. Liu, Jun and Chen, R., 1998. Sequential Monte Carlo Methods for Dynamic Systems, *Journal of the American Statistical Association* 93: 1032–1044.
11. Grabner, K.G., J.P. Dwyer, B.E. Cutter, 1997. Validation of BEHAVE Fire Behavior Predictions in Oak Savannas Using Five Fuel Models, *Proc. Eleventh Central Hardwood Conf.*, Univ. of Missouri, Columbia, pp. 202–215.
12. Hu, X., F. Gu, X. Yan, H. Xue, 2010. Data Assimilation Using Sequential Monte Carlo Methods in Wildfire Spread Simulation, working paper
13. ROTHERMEL, R.C. 1972. A mathematical model for predicting fire spread in wildland fuels. USDA For. Serv. Res. Pap. INT-115.

## Biography

Dr. Xiaolin Hu is an Associate Professor in the Department of Computer Science at Georgia State University, Atlanta, Georgia. He received his Ph.D. degree from the University of Arizona, M.S. degree from Chinese Academy of Sciences, and B.S. degree from Beijing Institute of Technology in 2004, 1999, and 1996 respectively. His research interests include modeling and simulation theory and application, agent and multi-agent systems, and complex systems science. He has served as program chairs for several international conferences/symposiums in the field of modeling and simulation, and is an associate editor for Simulation: Transaction of The Society for Modeling and Simulation International. Dr. Hu is a National Science Foundation (NSF) CAREER Award recipient.