

# Position Paper on Validation of Network simulation models

Rajive Bagrodia, Mineo Takai  
Computer Science Department  
UCLA

[rajive@cs.ucla.edu](mailto:rajive@cs.ucla.edu), [mineo@cs.ucla.edu](mailto:mineo@cs.ucla.edu)

## 1. Introduction

Validation concerns for network simulation models arise in the following contexts<sup>1</sup>:

- **Simulator validation:** Has the simulation tool or language (e.g., OPNET, NS, PARSEC, ...) been validated? This is similar to the question of whether a particular C compiler from a given vendor is bug free.
- **Protocol validation:** Does the simulation model of a given network protocol faithfully replicate details of the protocol based on its specification, implementation, or both? For instance, is the TCP model provided by the PARSEC Glomosim library or the NS library correct with respect to actual TCP implementations?
- **System validation:** A model simulates the use of one or more physical resources. The results of the model are significantly affected by the accuracy with which these resources have been simulated in the model. For wired networks, depending on the specific model, this might include a model of the CPU and Network Interface card (NIC) to simulate processing delays and overheads and a delay model for the network connecting the source to its destination. For wireless networks, this includes a model of the CPU for processing delays and appropriate models of the radio and the signal propagation in the air (channel models) to account for different types of transmission losses (e.g., multipath fading, shadowing, etc.)
- **Scenario validation:** Have the traffic, mobility, and other parameters used in the simulation experiments been validated against real life situations in which the corresponding technology is likely to be deployed? What is the degree of sensitivity of the results to minor (or major) variations in the critical parameters of the assumed scenario.
- **Validation of the (simulation) experimental methodology:** The validation of a model along each of the preceding dimensions is necessary though insufficient for accepting the results from a given simulation experiment. Most simulation models use random number generators. To ensure that results from an experiment are statistically valid, it is necessary that the experiment be designed with appropriate attention paid to simulation and analysis methodology to remove statistical bias.

Excellent discussions of methodology validation are available in simulation texts [e.g. 3, 4]. In the remainder of this paper, we address the validation issue for network models in the remaining four contexts.

## 2. Tools:

Under support from DARPA, we are developing two primary tools to address the problem of large scale models:

- A **kernel** for parallel discrete event simulation that encompasses a family of simulation algorithms (*sequential, parallel conservative, parallel optimistic, and hybrid*) and parallel architectures (*shared memory, distributed memory, and clusters*). This kernel has been successfully integrated in a C-based language called PARSEC, and we are currently integrating this kernel with the existing NS library written using C++. The initial target for this integration is a shared memory architectures using conservative simulation algorithms which were found to be most effective for parallel network models[9].
- A **library** of models for scalable wireless and wired networks. This library currently includes detailed models of the protocols listed below. The library has been successfully integrated with multiple models written using NS. In particular, the NS model of Tahoe TCP has been integrated and we are in the process of integrating other TCP models from the NS distribution. We have also successfully integrated the DSDV routing model written in NS and are in the process of integrating other wireless routing models developed in NS [8].

**Radio/Channel:** Free Space, Rayleigh, Ricean, SIRCIM

**MAC:** CSMA, MACA, IEEE 802.11, FAMA

**Network:**

**Unicast:** IP with Bellman-Ford, OSPF, DSDV (NS), DSR, WRP, DREAM, LAR

**Multicast:** ODMRP, AST

**Transport:** TCP (Free BSD), TCP Tahoe (NS), UDP, RTP

**Application:** Telnet, FTP, NetMeeting, WebPhone, Replicated file system

---

<sup>1</sup> We use the term *validation* to also mean *verification*, which is the term more commonly used in the programming community to address software correctness.

### 3. Validation Techniques and Self assessment

**Simulator validation:** The simulation kernel has been validated by a number of methods. First, the design and implementation of the kernel was accomplished by a small design team of four people and used detailed internal code review for the critical components. Second, the kernel and its PARSEC implementation was tested using a variety of simulation programs (that include network models but also include other applications including VLSI design, queuing networks, parallel programs, and database systems). Many of these programs were previously executed with another simulation kernel in the Maisie language that was developed by a previous generation of graduate students. This allowed the tests to be independent. Third, the simulator has been used to run (without crashing!) very large models containing more than 20,000 wireless nodes on multiple architectures, which lends further credibility to its robustness. The simulator has also been used by a number of students in undergraduate and graduate courses at UCLA both in parallel computing and networking courses. Fourth, the Maisie and PARSEC simulators have been downloaded by over 500 sites worldwide and have been ported to over a dozen different operating systems by its developers and other users. In particular, a group at MIT Lincoln Laboratory is currently porting the simulator to the Mac OS. While none of these guarantee absence of bugs, taken together it provides a relatively high degree of confidence in the correctness of the simulator kernel.

The TCP Tahoe model was recently ported from NS to GloMoSim. During the port, the model itself was not changed in any manner; rather the interface calls were changed to support execution of the model with the PARSEC kernel rather than the NS kernel. These TCP models were executed with a common scenario in which each node transmits packets at a constant rate, and were found to yield same results. We also tested another scenario that uses traffic based on telnet. Although the two results are not identical due to the different random number generators used in the two simulation environments, they were within appropriate statistical bounds.

**Protocol model Validation:** Models for each of the protocols must be correct in the sense that they correctly implement the functionality described in the specification or deduced from an implementation (as many protocols do not have a complete and unambiguous specification); this is more commonly referred to as protocol, or in general, model verification. A number of standard techniques have been used for model verification that include the use of structured programming, detailed code reviews, and testing. Testing includes running a scenario with known results and comparing the model output with the expected result, and running a detailed trace of the event sequence in a model execution to ensure that it follows the expected path. Availability of a graphical display of the event sequence in the context of the physical system being simulated can be of considerable assistance.

A simulator may provide one or more tools to help an analyst in the verification of her model, but unlike simulator kernel validation, model validation is the responsibility of the analyst. For the model library that has been developed, we have used one or more of the preceding software engineering principles for model verification. In most cases, the models were developed by teams of two or more people with internal code reviews, testing with known cases, and event tracing. In many cases, the exercise identified incomplete specification for the corresponding protocol and required multiple interactions with the protocol designers. In at least one case, the model design team uncovered errors in the published specification that were acknowledged and corrected in later versions of the protocol.

In addition to the standard validation techniques discussed above, we have two case studies for specific protocols, where the validation has been quite successful:

- **Direct incorporation of implemented protocol in the model:** A protocol model can be validated trivially against an operational prototype, if the implemented code for the protocol is itself incorporated in the model! This was exactly what was done for the TCP model included in the GloMoSim library. The actual code for the protocol was integrated directly from the protocol stack distributed with Free BSD including the code for the headers, the connection set up, buffer management etc. The only changes made to the code were at the interface in order to integrate it with the rest of the protocol stack. Similarly, the model of RSVP developed in PARSEC was derived for the most part from the actual code from the ISI implementation [6] as part of the same distribution.
- **Comparison of independently developed models for a given protocol:** Another approach to validation is to develop models of a given protocol independently by different groups possibly using different simulation tools. Although this is resource intensive, the degree of confidence in a model that has been validate din this manner tends to be very high. As the integrated library will include NS and GloMoSim models, this will indeed be possible for the TCP and 802.11 models, which are already available in both environments. As these are standard components of many protocol stacks, such a validation will be of substantial benefits to the research community. We have already compared the results from simulation of identical models using both NS and GloMoSim implementations of 802.11. While the traces reveal that the 802.11 models are almost identical, the summary results show some differences –Table 1 presents the results from

a small scenario with 3 nodes that create the hidden terminal effect. These differences appear to be due to differences in the radio models in the two environments which are currently being investigated.

**Table 1: Simulation results with two independently developed 802.11 models**

<u>Scenario</u>	<u>CBR sources</u>	<u>Data Pkts Sent</u>	<u>Data Pkts Lost</u>	<u>RTS Pkts Sent</u>	<u>RTS Pkts Lost</u>	<u>CTS Pkts Sent</u>	<u>CTS Pkts Lost</u>	<u>ACK Sent</u>	<u>ACK Lost</u>
NS	2 x 10pkts/sec	3992	0	4428	436	3992	0	3992	0
GloMoSim	2 x 10pkts/sec	4002	4	4407	405	3998	0	3998	4
NS	2 x 20pkts/sec	7938	0	8794	856	7938	0	7938	0
GloMoSim	2 x 20pkts/sec	8002	4	8845	843	7998	0	7998	4

The size of packet: 512 bytes.

**System validation:** This is perhaps the hardest problem in validating simulation models of wireless networks. For wired networks, where the packet transmission issues are relatively well understood and validated analytical models exist for many of the common networks, this is less of an issue. However, the sources of signal loss in a wireless environment are numerous and although the causes are well understood, there are relatively few validated models for wireless transmission in indoor or outdoor scenarios. The various types of models that have been used for wireless transmission media include *free space*, *statistical impulse response*, *trace based*, and *ray tracing*. Free space is the most common model where the received signal strength is computed assuming a perfect obstacle free environment, where transmission losses due to multipath fading, shadowing, etc. are ignored. Another common approach is to use a statistical impulse response model which has been derived from a set of measurements in a given scenario. In particular, we have compared message loss statistics using a free space model and the SIRCIM[5] model. Depending on the size of the networks, traffic patterns and other configurations, the results predicted by the two models may be within 5% or differ by more than 200%.

**Scenario validation:** A scenario consists of the deployment of various communication devices which includes both the number and initial placement of each device, the mobility pattern of each device, and the application level message traffic (both the number of generated packets and their distribution in the network) generated by the model. From the military perspective, programs such as NETWARS[1] and SEAM-LSS [7] are developing the capability to define standard formats to represent the communication devices, their mobility and communication patterns using a set of tools including MODSAF (MODular Semi Automated Forces) and communication threads. The former is used to generate doctrinally correct force movement pattern whereas the latter specifies doctrinally correct exchange of messages among a group of related objects participating in a military exercise. GloMoSim provides to the analyst the ability to take a scenario as specified by the preceding set of tools and run a simulation experiment where the specific radio and the communication protocol stack is composed from appropriately selected models at each layer. This capability allows an analyst to compare protocol alternatives in the context of realistic scenarios rather than those based on random distributions.

#### 4. Key Difficulties

We identify three major difficulties in validating simulation models of networks: first, there is lack of detailed specifications on the protocol implementations for *both* commercial and research radios. In the case of commercial radios, this is often proprietary information; for research efforts the details change sufficiently rapidly that published papers are typically out of date. Direct communication among the modelers and system designers is thus essential. Second, models of the channel propagation are perhaps the hardest to validate; the widely used free space models can be quite inaccurate and the more accurate models (e.g., SIRCIM) are very resource intensive. We have found that parallel execution can give substantial performance benefits *particularly* for the computation intensive models like these. Last and perhaps most important, it is much easier to validate individual protocol models than the entire communication device that includes radio, propagation models, and protocol models. Achieving this objective requires a sustained effort and interaction among the device designers, modelers, and the propagation medium modeling community.

#### References

1. Y. Atamna, "NETWARS: toward the definition of a unified framework for modeling and simulation of joint communication systems," In proceedings of SPIE, Vol. 3393, pp.162-169, April 1998.
2. R. Bagrodia, R. Meyer et al., "PARSEC: A Parallel Simulation Environment for Complex Systems," IEEE Computer, October 1998.
3. R. Jain, "The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling," John Wiley & Sons, April 1991
4. A. M. Law and W. D. Kelton, "Simulation Modeling and Analysis," McGraw-Hill, 1991.

5. T. S. Rappaport, S. Y. Seidel and K. Takamizawa, "Statistical Channel Impulse Response Models for Factory and Open Plan Building Radio Communication System Design," IEEE Transactions on Communications, Vol. 39, No. 5, May 1991.
6. RSVP: ReSerVation Protocol, <http://www.isi.edu/div7/rsvp/>.
7. SEAM-LSS: Simulation and Evaluation of Adaptive Mobile Large Scale Network Systems, <http://www.seamlss.com/>.
8. Wireless and Mobility Extensions to ns-2, <http://www.monarch.cs.cmu.edu/cmu-ns.html>.
9. X. Zeng, R. Bagrodia and M. Gerla, "GloMoSim: A Library for Parallel Simulation of Large-Scale Wireless Networks," 12<sup>th</sup> Workshop on Parallel and Distributed Simulation, May 1998.