Proceedings of the 2006 Winter Simulation Conference
L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, eds.

MULTI-SCALE INTEGRATED INFORMATION AND TELECOMMUNICATIONS SYSTEM (MIITS): FIRST RESULTS FROM A LARGE-SCALE END-TO-END NETWORK SIMULATOR

Roman Waupotitsch

Decision Applications Division
Los Alamos National Laboratory
Mail Stop K557, Los Alamos, NM 87545, U.S.A.

Stephan Eidenbenz
James P. Smith

Computer and Computational Sciences
Los Alamos National Laboratory
Mail Stop M997, Los Alamos, NM 87545, U.S.A.

Lukas Kroc

Department of Computer Science
Cornell University
4130 Upson Hall, Ithaca NY 14853, U.S.A.

ABSTRACT

Performing realistic simulations of Internet packet traffic on a national or global level is a daunting task from both the modeling and the computational perspective. We present the MIITS (Multi-scale Integrated Information and Telecommunications System) tool that implements a novel approach to network simulation and report first scaling results from a realistic Internet scenario. MIITS' end-to-end approach to network simulation relies on modules for (i) accurate network topology and capacity representation, (ii) realistic communication session generation based on the activities of an agent population that is statistically equivalent to the population in a large metropolitan area, (iii) the actual scalable packet-level network simulation that is based on distributed event–driven technology, and (iv) analysis of large amounts of simulation output data. We present a sample simulation of a Los Angeles network as an intermediate step toward the vision of national-level simulation.

1 INTRODUCTION

Packet-level communication network simulation is computationally intense as a single agent or network node can generate multiple sessions at the same time that each create hundreds of data packets that then in turn need to traverse the network via a potentially large number of intermediate hops. A large number of network simulation tools has been developed. A non-exhaustive list includes SSFNET (SSFNET), ns-2 (NS2), QualNet (QUALNET), GTNetS (Riley 2003), GloMoSim (Zeng et al. 1998), OPNet (OPNET), pdns (PDNS), USSF (Rao et al. 1999), DaSSFNet (DASSFNET), SWAN (Liu et al. 2001). Each simulator was designed with a different focus (e.g., SWAN is for wireless simulation; OPNET and QualNet are commercial tools with a focus on accurate representation of hardware-specifics; SSFNet, GTNetS, and pdns are aimed at simulating large-scale Internet traffic).

We believe the Grand Challenge in network simulation is to simulate on a packet-level the global communication traffic generated by 5 billion people using and instructing a similar number of diverse electronic communication devices; see (Fujimoto et al. 2003) for a first attempt. Meeting this challenge requires changes not only in simulation technology, but maybe even more in traffic and network modeling concepts. Existing simulators have all focused on a few aspects of the grand challenge: for instance, OPNET allows for excellent level of detail, while SSFNet aims at optimizing packet-flows in Internet settings, ns-2's wireless simulation focuses on accurate (and thus expensive) physical layer representation. The MIITS tool that we introduce takes new aspects into account and expands on well-established aspects.

A first key aspect of MIITS is an end-to-end approach to simulation. Traditionally, a network simulator is considered to be given network topology and capacity information as input, maybe in addition to a few parameters (e.g., mean value) that characterize the distribution according to which traffic gets generated. MIITS' approach to network simulation puts a lot of emphasis on creating realistic network topology and realistic communication sessions between end users by individually modeling end user devices and behavior. It does so by leveraging existing socio-

technical simulation technology developed at Los Alamos National Laboratory (TRANSIMS), (Barret et al. 2002), and (Barret et al. 2004).

A second key aspect is an integrated approach across various types of networks. This requires integration of packet-switched and circuit-switched networks including conversions between the two technologies, it requires representation of end devices that can access different types of networks (e.g., some Blackberry devices operate on 802.11, Bluetooth, and cell phone networks). Traditionally, simulations of, say PSTN (Public Switched Telephone System), cell phone networks, and the Internet have been executed separately from each other. With the advent of VoIP, and increasing substitution of PSTN through cell phones, such separate simulation ignores increasingly important network-interdependency aspects.

A third key aspect is scalability. Scalability is largely recognized to be hard to achieve in network simulation. Several network simulation tools focus on scalability. MIITS' approach to scalability is an uncompromising distributed simulation design. All key MIITS modules, in particular Session Generation and Network Simulation, rely on distributed discrete-event simulation. We have found that even the session generation module has memory requirements, when simulating a metropolitan area, that greatly exceed those of a single processor machine.

The objective of this paper is to introduce the MIITS approach with a focus on the actual network simulation technology and to report first simulation results for a large-scale study. In Section 2.1 we introduce SimCore the simulation interface, in Section 2.2 we present NetSim, a packet-level simulation system, both developed at Los Alamos National Laboratory. In Section 3 we present some data on the performance of the system, in Section 4 we present result obtained on the Los Angeles network, and finally in Section 5 we conclude with some evaluative remarks and the path forward.

2 SYSTEM OVERVIEW

The system presented here is a module of MIITS a scalable, end-to-end simulation environment for representing and analyzing extremely large, complex communication networks of any type, including cellular networks, public switched telephone networks (PSTNs), the Internet, and ad hoc mesh networks. MIITS in turn is a module of the Urban Infrastructure Suite (UIS) developed at Los Alamos National Laboratory. UIS is a set of seven interoperable modules that employ advanced modeling and simulation methodologies to represent urban infrastructures and populations.

MIITS employs state-of-the-art software engineering principles in order to achieve its goal of being the world's fastest, largest-scale, most accurate, packet-based, general-purpose and integrated network simulator for use in large-scale urban simulation. MIITS is a unique tool that models any communication network if there is data available on how the network operates. It offers network representation in several resolutions, ranging from packet-level simulation to flow-based approaches, giving the user the possibility to choose a level of detail in the simulation that is suitable to his/her specific needs when making the trade-off decision between accuracy and reasonable turn-around time is the key design criterion for MIITS.

Figure 1 shows the types of networks that are covered by MIITS. Traditional wireless and wire-line networks such as the public switched telephone network (PSTN), the cell phone network and the Internet are covered by MIITS; in addition, emerging technologies, including ad hoc networks, sensor networks, WLANs are accommodated too.

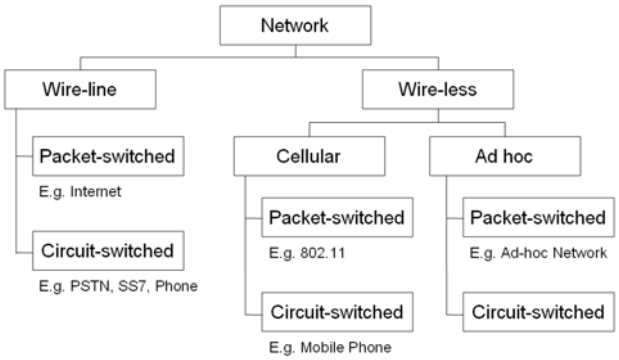

Figure 1: Types of Networks Covered by MIITS.

Figure 2 shows an overview of the architecture of MIITS. Each box represents a module of the system. Data flows from left to right, with the Core Simulation Framework providing the core libraries and architecture on which all other modules are based (discussed in detail in section 2.1). The System Integration & Applications layer represents specific integrated applications build on these modules.

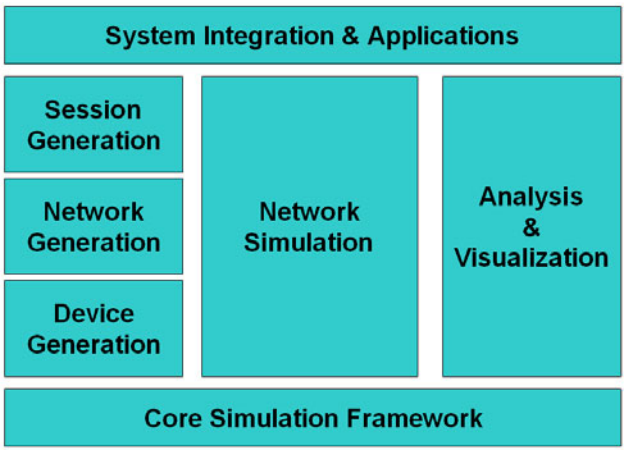

Figure 2: MIITS Architecture

The three data generation modules create the input for the Network Simulator. They receive their input from other modules of the UIS that create that data based on survey data. Module Network Generation generates the higher order elements of the communication network, for example routers connecting through media such as wire-lines or the ether. Module Device Generation creates the communication devices and assigns the devices to individual users, possibly varying assignments over time. Module Session Generation generates communication sessions between network devices. Module Network Simulation performs the actual simulation and is discussed in detail in Section 2.3.

2.1 End-To-End Simulation

One of the main objectives in the development of MIITS was the ability to perform end-to-end simulations of population communication. By this we mean that all types of communication as depicted in Figure 1 are simulated for the entire population on a city, country, and global level. To achieve this admittedly ambitious goal various system components needed to be developed. One was the simulation system designed to be scalable to run in parallel on thousand of computing units. We have developed a core simulation framework, SimCore, which provides the foundation of the modules. This framework is shown as part of the overall system in Figure 2 and is discussed in detail in section 2.2.. In this document we focus on one system that was built on this framework, Netsim, the simulation of packet-switched traffic. This system will be discussed in detail in Section 2.3.

Second, there is the data. End-to-end simulation requires that the communication patterns of every person in the scenario on every communication device be simulated. This includes all the supporting infrastructures and supporting devices, and requires analyzing statistical data and survey data to create a synthetic population that closely resembles the real population on an individual level. Examples are the populations mobility patterns, their communication patterns, and also the devices used for communication: phone, cell phone, email, web pages, etc. This data is created using the systems and methods developed at Los Alamos over the last 15 years and creation of this data may represent by itself one of the most time-consuming aspects of MIITS.

And finally, one needs the computing infrastructure to execute the simulation, which in the case of MIITS consist of grids of single or multi CPU computers.

MIITS is still in the early stages of development, but has already proofed an indispensable tool in such analyses as hurricane impact studies, and infrastructure vulnerability analysis. An example of how a real-world network could be analyzed is provided in Section 4.

2.2 SimCore

SimCore is a generic discrete event simulation framework that provides an *entity-service* level interface on which specific simulation systems can be built. Its first application was an end-to-end packet routing simulation system, Netsim. However, in the meantime its versatility has been recognized and it is being considered in applications that are not internet related. A second application for which it has already been used is the Session Generator depicted in Figure 2.

SimCore isolates the implementation of simulation systems from the simulation engine details or logical processes (LPs). Simulations are implemented in terms of entities that receive packets, handle or service those packets, and send packets to other entities, where they are handled in a similar fashion. The details of the actual location of the receiving entity (on which LP, and on which CPU) are hidden from the implementation. This is facilitated by an API level specification of the interface based on a generic programming paradigm, and a generic data input mechanism that can be used for any combination of entities and services.

SimCore provides an application interface (API) on which to develop discrete event simulation applications. It also provides an API for the simulation engine. Currently, SimCore uses DaSSF (DASSF) as the underlying simulation engine. However, any engine that supports a conservative synchronization mechanism could be used. SimCore provides a generic set of base classes from which all components in a specific simulation application can be derived. Finally, SimCore provides the I/O mechanisms for the system. Figure 3 shows the conceptual layout of the System.

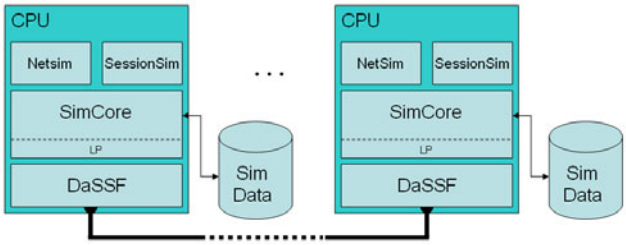

Figure 3: Layers of Components of the Software System.

A visual representation of how events (called Info packages) are sent between the services of entities is shown in Figure 4 and Figure 5. Figure 4 shows the flow of an Info Package from a source entity to a target entity with the package created at time *t1* and delivered at time *t2*, the specified arrival time. As described earlier, events need only provide the address of the destination entity and the address of the service on that entity and leave to SimCore the actual sending of the event through the system. Figure 5 shows the detailed flow of an Info Package: the Info Package is encoded into an EventInfo package by SimCore

and is sent by DaSSF to the destination. At the destination a corresponding unpacking sequence is performed. Figure 5 also shows that the time at which the EventInfo package is sent is not explicitly defined and is up to the simulation engine. On the other hand the time when the Info package is sent to and received by SimCore is explicitly known.

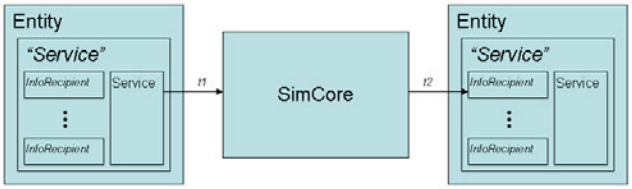

Figure 4: Flow of an Info Package from a Source Entity to a Target Entity

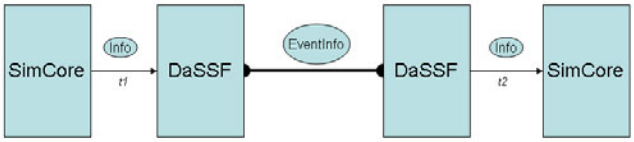

Figure 5: Detailed Flow of an Info Package

2.3 Netsim

Netsim is an implementation of a packet switching network. It is built on SimCore and provides implementation of the entire protocol stack as well as the devices involved in the simulation. As such it implements the physical layer, the MAC layer, the network layer, the transport layer, and the application layer. It also implements representations of physical devices, *Devices*, *Interfaces*, and *Media*, that host the layers. Devices are the simulation representation of the physical devices hosting the higher level protocols such as the application layer. Interfaces are the simulation representation of the interface card(s) that facilitate communication for the Device. Finally, Media are the simulation representation of the media that transport the data, either wired or wireless.

Netsim provides base implementations for all the protocols and specific implementations of particular protocols. Various physical devices are implemented using parameterizations of the object representing the device.

Physical devices are organized in a hierarchy. See Figure 6. Devices contain their Interfaces, and Interfaces contain their (unique) Medium. Physical devices are implemented as subclasses of Entities. As such they maintain a list of "Services" that they provide. The application layer, the transport layer, and the network layer are associated with Devices. The MAC layer and the physical layer are associated with the Interface layer. Media provide connectivity information.

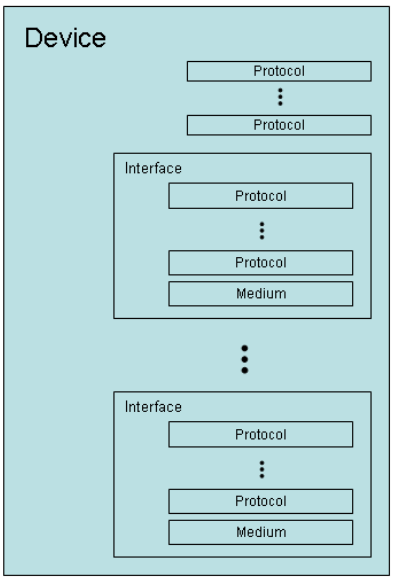

Figure 6: Hierarchical Organization of Devices and Protocols within Netsim

2.4 Routing

Routing is one of the crucial elements in any network simulation system. Netsim employs a generic mechanism to implement routing. This allows different routing mechanisms to be used in different simulation scenarios. It allows routing to be implemented locally, i.e. on the same host as the protocol that requires routing information, or it allows routing to be implemented on specialized routing servers. Routing may be implemented using a shortest path approach or a hierarchical approach depending on the requirements of the simulation. It may be implemented with the routing computations distributed over multiple hosts. Finally, routing is implemented in such a fashion that only those nodes that actually participate in the routing of packets are included in the representation of the routing network. Due to the open design of NetSim, alternative methods, such as neighbor-index vectors (Riley et al. 2000), can be implemented with minor implementation effort.

3 SYSTEM PERFORMANCE

3.1 Memory

This section presents performance characteristics of the system. It includes performance that is tied to the Session Generator and Netsim, and to the underlying simulation engine DaSSF.

Memory requirements for devices is approximately 3000 bytes. This includes data for the interfaces and protocols, an indication of the overhead that is incurred when implementing a system using a true object oriented or generic implementation paradigm.

Typically one would expect there be significantly more messages involved in a simulation than there are devices. As an example the simulation presented in the next section which simulates a network with 1 million devices and 1 million sessions on that network sends 2.255 billion messages. Messages come in two flavors, control messages and actual IP packets sent in the system. Not all of these messages, but at times a large portion are stored in the simulation engine, which at a size of 200 bytes per message can make up the bulk of the used memory.

3.2 Speed

As mentioned earlier, the system was developed to be run in a parallel environment. We ran the system on a Linux cluster of 60 1.4Ghz Dual CPUs running MPICH2.

Figure 7 shows that parallel execution generated a significant speedup. Our experiments show that this is mostly due to the parallelism created for the routing computation. If the time to compute routes is ignored the speedup is less pronounced and the simulation overhead may cause the walltime on fewer CPUs to exceed that on more CPUs in scenarios with sparse packet density (i.e. packets per simulation time). We believe that in these cases the efficiency of the parallelism resulting from distributing the simulation over many CPUs is negligible and is outweighed by the overhead resulting from the communication through the network and the increased overhead for the simulation engine.

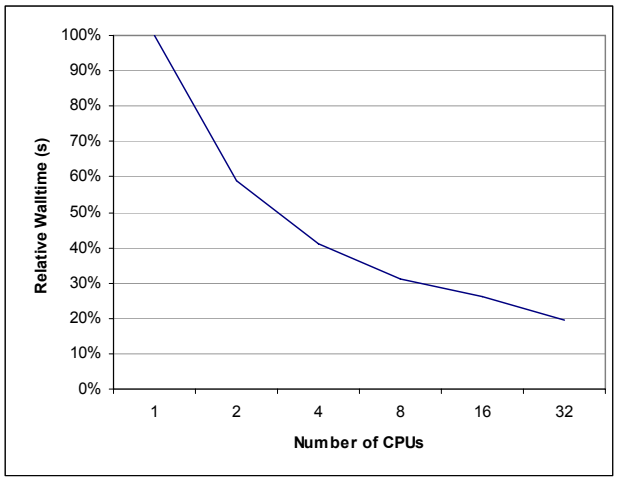

Figure 7: Relative Walltime of Simulations Run with Different Numbers of CPUs

The typical performance of the simulations presented in the following sections was a CPU time of 602639 seconds (or 167.4 CPU hours) for 1 million sessions (with an average of 72 packets).

4 LOS ANGELES NETWORK SIMULATION

4.1 Network

Local Data of Los Angles was collected using trace route from various national locations. This resulted in a network with 100k nodes. This network contained a significant number of nodes with degree 2. We collapsed these nodes resulting in a smaller network of 11225 nodes, which we refer to as the core network. In the collapsing process the overall structure of the network was maintained. To ensure connectivity of the core network we introduced bridges that connected the individual components. In our simulation core nodes are the nodes which perform the routing and can be considered the local backbone of the Los Angeles network.

Connected to the core network are stub lines that connect to end nodes which represent the end-devices. End devices are divided into clients and servers. Clients represent the end-users, servers represent sites such as Web servers and Email servers. Connections between core node and core nodes, and servers and core nodes are simulated using a bandwidth of 10Mbit/sec, client to core node connections are simulated using lines with a bandwidth of 1 Mbit/sec. Our simulation connects 90 end-devices through stub lines to every node of the core network, resulting in a total network size of 11225*90 = 1010250, i.e., roughly one million nodes.500 of the devices are identified as servers, 5000 are identified as clients. All traffic in the simulation is between these identified clients and servers.

4.2 Scenario

We uniformly chose data downloads using UDP as the model of sessions to disentangle network capacity from the particularities of the protocol. The packet size is also chosen uniformly at 60Kbits per packet. The distribution of sessions was chosen such that 30% of the session send 100 packets and the overall average is 72 packets per session. This is equivalent to an average of 540kbyte per session.

To eliminate the overhead incurred in routing we chose to compute all shortest paths at the startup and store this data in a lookup table. For the different simulation runs the average separation in time between sessions was 1.5, 0.5, 0.1, 0.01, 0.001, and 0.0001 seconds, with the maximum being twice this number and the minimum being 0.

A sample movement of a session with a single packet through the network is shown below. Sessions were numbered starting at 1000000, and the trace shown is the movement of the first packet of the first session.

```
Time        EntityId    Service     Type    Info
0.929887    (d 87569)   eProt_FTP   1   1000000
0.93        (i 96910)   eProt_FIFO  3
0.930101    (i 96910)   eProt_UDG   1   60
0.936201    (i 96910)   eProt_FIFO  4
0.936201    (i 12115)   eProt_FIFO  3
```

```
0.937174    (i 12115)   eProt_UDG   1   60
0.943274    (i 12115)   eProt_FIFO  4
...
0.943274    (i 19084)   eProt_FIFO  3
0.943455    (i 19084)   eProt_UDG   1   60
0.949555    (i 19084)   eProt_FIFO  4
...
0.998147    (i 19961)   eProt_FIFO  3
0.999045    (i 19961)   eProt_UDG   1   60
1.00515     (i 19961)   eProt_FIFO  4
1.00515     (i 213027)  eProt_FIFO  3
1.00516     (i 213027)  eProt_FIFO  4
1.00516     (d 103759)  eProt_FTP   2   1000000
```

4.3 Results

Differences in session spacing allows us to analyze how the network performs for scenarios with different network loads. For example the packet shown above moves through the network in less than 1 second, with an average of 0.07 seconds per hop. This is approximately what one would expect. We measure the average time to finish a session (with an average of 72 packets as described above). With sufficient spacing the time required is 0.7 seconds to 0.8 seconds. As session spacing decreases the time increases to 17 seconds.

The reason of the slowdown becomes apparent when analyzing the load on nodes in the network. There is a small set of nodes which handle a large portion of the traffic. As session spacing decreases the load on these node increases dramatically. We measured the average lengths of the queues in the MAC layer. For sessions with spacing larger than 0.01 seconds the load on the queues is trivial with the load on the node with maximum load being 11. The queue lengths become interesting for the scenario with an average session spacing of 0.001 seconds. The queue lengths for the 3 nodes with maximum average queue lengths are shown in Figure 8.

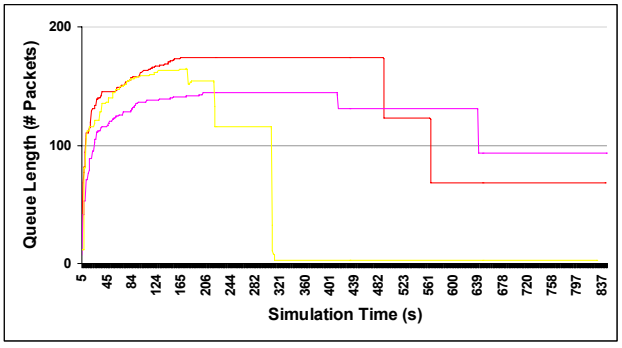

Figure 8: Profiles of Average Queue Lengths of Node 4500 (blue), Node 4538 (red), and Node 7300 (yellow)

The maximum queue length was set to 512 packets. We recorded dropped packets in the system. This record is shown in Figure 9. This figure shows that only a small number of packets is lost for a session spacing of 0.001 seconds. The actual traffic through the nodes shown are shown in Figure 10.

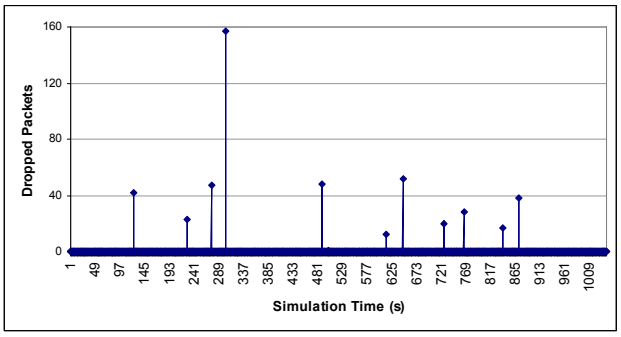

Figure 9: Number of Dropped Packets in the Network for Session Spacing of 0.001 Seconds

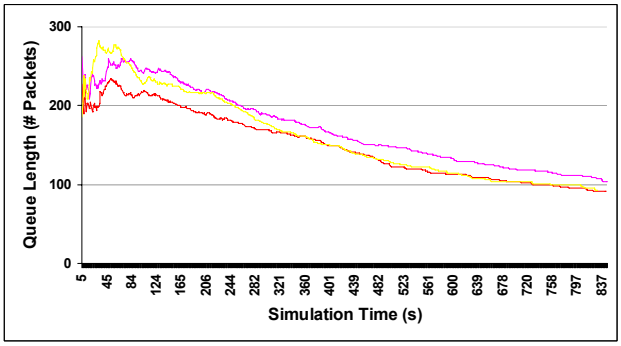

Figure 10: Number of Packets per Second through Node 4500 (blue), Node 4538 (red), and Node 7300 (yellow) for Session Spacing of 0.001 Seconds

The distribution of the loads of the nodes which have an average of at least 10 packets is shown in Figure 11 (for session spacing of 0.001 seconds). As can be seen 485 nodes, or 4%, of the core network have a queue buildup of more than 10 packets. Once the session spacing is reduced further by an order of magnitude, the network is unable to handle the load. Figure 12 shows the trace of the dropped packets in that scenario for the first few seconds of simulation time.

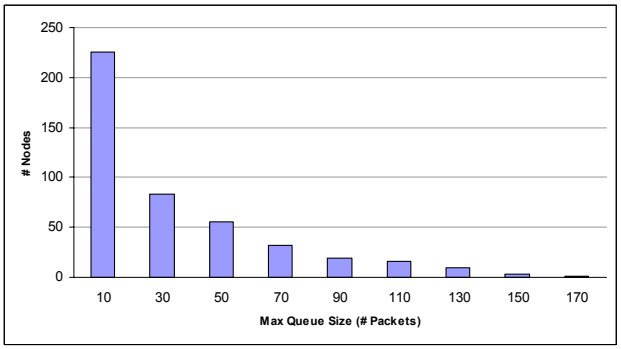

Figure 11: Histogram of Queue Sizes of Network Nodes

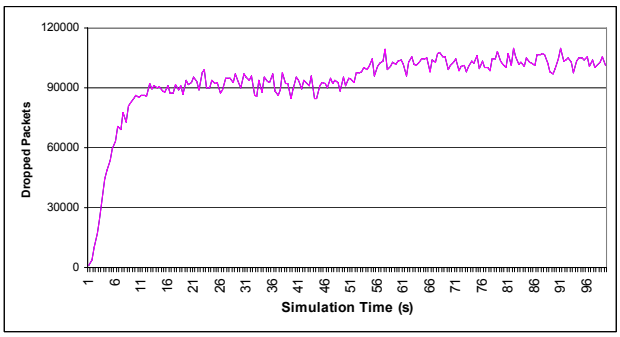

Figure 12: Number of Dropped packets in Network (Session Spacing of 0.0001 Seconds)

5 CONCLUSION AND FUTURE WORK

We present a truly scalable system to simulate internet traffic for large local and the national network. We demonstrated proof feasibility by simulating a large local network on a network topology that, if not exactly, but closely resembles that of Los Angeles.

Ongoing and future development include improving the memory footprint of the system and an improved distribution strategy for the device nodes to reduce inter CPU message traffic. We expect to implement a routing scheme that takes the hierarchical nature of the internet topology into account. And finally we are working on improving the fidelity of our internet topology representation.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the numerous and substantial contributions that have flown into MIITS. In particular, we would like to thank the whole MIITS team, which has included about 25 individuals over the course if its life. We also acknowledge the contributions of MIITS' predecessor projects and in particular the contributions of Madhav Marathe and Chris Barrett.

REFERENCES

Barrett, C. L., M. Drozda, M V. Marathe, S. S. Ravi, and J P. Smith. 2004. A mobility and traffic generation framework for modeling and simulating ad hoc communication networks. *Scientific Programming* 12(1): 1-23.

Barrett, C. L., M V. Marathe, J. P. Smith, S. and S. Ravi. 2002. A mobility and traffic generation framework for modeling and simulating ad hoc communication networks. *Proceedings of the ACM Symposium on Applied Computing (SAC 2002)*, pp. 122-126.

DASSF DaSSFNet: A High-Performance Network Simulator. Available via <http://www.crhc.uiuc.edu/~jasonliu/projects/ssfnet/index.html> [accessed June 21, 2006].

Fujimoto, R. M., K. Perumalla, A. Park, H. Wu, M. H. Ammar, and G. F. Riley. 2003. Large-Scale Network Simulation: How Big? How Fast?. *Proceedings of the 11th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems. (MASCOTS '03)*, Orlando, FL, USA, pp. 116-123.

Liu, J., L. F. Perrone, D. M. Nicol, C. Elliott, and D. Pearson. 2001. Simulation Modeling of Large-Scale Ad-hoc Sensor Networks. *Proceedings of the European Simulation Interoperability Workshop 2001* (EUROSIW).

NS2: The Network Simulator – ns-2. Available via <http://nsnam.isi.edu/nsnam/index.php/User_Information> [accessed June 21, 2006].

OPNET: OPNET Technologies, Inc. Modeler. Available via <http://www.opnet.com/products/brochures/Modeler.pdf> [accessed June 21, 2006].

PDNS: PDNS Parallel/Distributed NS. Available via <http://www.cc.gatech.edu/computing/compass/pdns> [accessed June 21, 2006].

QUALNET: Scalable Network Technologies, Qualnet. Available via <http://www.scalable-networks.com> [accessed June 21, 2006].

Rao, D. M., and P. A. Wilsey. 1999. Simulation of ultra-large communication networks. *Proceedings of the 7th IEEE/ACM Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'99)*, College Park, MD, USA, pp. 112-119.

Riley, G. F. 2003. Large-scale network simulation with GTNetS. *Proceedings of the 2003 Winter Simulation Conference*. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Riley, G. F., M. H. Ammar, R. Fujimoto. 2000. Stateless Routing in Network Simulations. *Proceedings of the IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems. (MASCOTS '00)*, San Francisco, CA, USA, pp. 524-531.

SSFNET. Available via <http://www.ssfnet.org> [accessed June 21, 2006].

TRANSIMS: Los Alamos National Laboratory, TRANSIMS. Available via <http://transims.tsasa.lanl.gov> [accessed June 21, 2006].

Zeng, X., R. Bagrodia, and M. Gerla. 1998. Glomosim: a library for parallel simulation of large-scale wireless networks. *ACM SIGSIM Simulation Digest*, 28(1):154-161, July 1998.

AUTHOR BIOGRAPHIES

ROMAN WAUPOTITSCH is a Technical Staff Member at Los Alamos National Laboratory in the Decisions Applications division. Previous positions include Vice President of Engineering at Geometrix, Inc, leading the development of systems for 3D object reconstruction and for 3D facial recognition, and Technical Manager at Raindrop Geomagic. He received his Diploma in Technical Mathematics from Graz University of Technology, and a M.S. and a Ph.D. in Computer Science from the University of Illinois at Urbana-Champaign. His email address is <rowa@lanl.gov>.

STEPHAN EIDENBENZ received his Ph.D. in Computer Science from the Swiss Federal Institute of Technology, Zurich in 2000. He obtained a Bachelor's degree in business administration from GSBA in Zurich in 1999. He is a technical staff member in the Discrete Simulation Sciences group (CCS-5) at LANL, where he leads the MIITS project that models and simulates large-scale communication networks. Stephan's current research interests are in wire-line and wireless networking, selfishness in networking, infrastructure modeling, discrete event simulation, and algorithms. Stephan has worked and published in a large number of fields, including approximation algorithms and inapproximability, visibility problems in polygons and terrains, error-modeling in computational biology, and network protocol design. He is also an active reviewer and member of various program committees (e.g. ACM MobiHoc, Infocom). His email address is <eidenben@lanl.gov>.

LUKAS KROC earned his M.Sc. degree in Computer Science at Charles University in Prague, Czech Republic. After his M.Sc. studies he worked as a Graduate Research Assistant at the Basic and Applied Simulation Science group at Los Alamos National Laboratory, where he took part in developing tools for large scale socio-technological simulations. He is currently a Ph.D. student at Cornell University. His email address is <kroc@cs.cornell.edu>.

JAMES P. SMITH is the NISAC (National Infrastructure Simulation and Analysis Center) project leader. He has also been a team leader and group leader in the Discrete Simulation Science Group of the Computing and Computational Sciences Division at Los Alamos National Laboratory, the technical lead for the NISAC EpiSimS project (Epidemiological Simulation System) and the NISAC population mobility project, and a technical staff member contributing to the NISAC telecommunications capability. His current research applies to modeling and simulation of socio-technical systems in national infrastructure, especially public health, telecommunication/computing, and transportation. He has scientific experience in high performance computing and parallel processing applied to large-scale microscopic simulations, including original software design and debugging of very large, evolving systems of inter-operable computational systems, and efficient analysis and synthesis of massive data produced by multi-scale complex environments. He also worked for a short time in nuclear theory, for several years in experimental biophysics at the Pennsylvania Muscle Institute, and started a company working in analytic finance. His graduate work included theoretical and experimental fusion and space plasma physics research. He has publications in biophysics, analytic finance, education, space plasma physics, communications, transportation, mathematical biology and computer science, and is a co-inventor on the TRANSIMS patent. He has a Ph.D. in Theoretical Plasma Physics from the University of Texas at Austin. His email address is <jpsmith@lanl.gov>.