Simulation of Large-Scale Heterogeneous Communication Systems

Lokesh Bajaj, Mineo Takai, Rajat Ahuja, Rajive Bagrodia
Computer Science Department
University of California, Los Angeles
Los Angeles, CA 90095

Abstract- **Large-scale hybrid networks that include wireless, wired, and satellite-based communications are becoming common in both military and commercial situations. This paper describes a scalable simulation environment that effectively utilizes parallel execution to reduce the simulation time of detailed high-fidelity models of large communication networks. The paper also presents a set of case studies that evaluate the performance of large wireless networks with thousands of nodes [*] and compares the impact of different lower layer protocols on the performance of typical applications.**

I. Introduction

High-level design problems for the digital communication infrastructure in the battlefield environment, as envisaged for DOD programs like the Warfighter Information Network (WIN), are extremely challenging in a number of dimensions: the scale is large; network traffic is a mix of voice, data, and imagery; connectivity can change dynamically in unpredictable ways; and the quality of service requirements are often severe. Fig. 1 presents a sample military communication scenario that includes wireless, wireline, satellite, and airborne communication assets. The total number of communication devices in such scenarios is often in the thousands, and for even modest deployments it can scale up to the tens of thousands. For joint force exercises, having a hundred thousand communication units is not unrealistic.

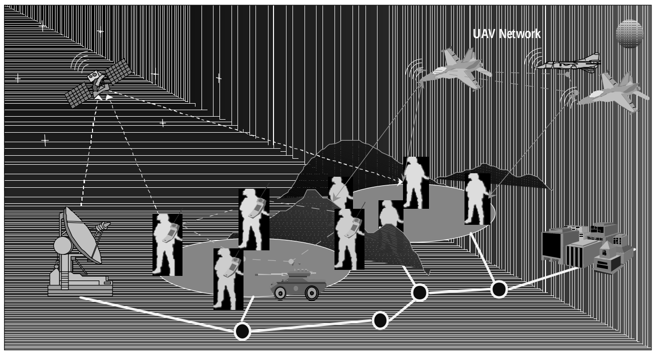

Fig. 1: Warfighter Information Network (WIN)

The multitude of proposed solutions at each protocol layer for wireless, wireline, and satellite networks has led to an explosion in possible design choices for these networks. The size of these networks makes experimentation and measurement prior to deployment impossible, yet the risks of deploying these new technologies in critical situations require assurance that they will work. Detailed, high fidelity simulation of the communication infrastructure can provide invaluable insights to help the DOD make appropriate choices. Under current funding from DARPA, we are developing a scalable simulation facility whose objective is to simulate networks with up to a hundred thousand nodes linked by a heterogeneous communications capability that includes multicast, asymmetric communications using direct satellite broadcasts, multi-hop wireless communications using ad hoc networking, and traditional Internet protocols. The scalability of the simulator to very large networks will be achieved primarily by exploiting parallelism on state-of-the-art parallel computers. We have already demonstrated the feasibility of using parallel model execution to achieve dramatic reductions in execution times of such models. For instance, we have been successful in running a detailed simulation of a large wireless network with 10,000 mobile radios. Using parallel execution, it was possible to reduce the execution time sufficiently such that a model with 10,000 wireless nodes could be simulated on six processors in less time than a network with half as many nodes using purely sequential execution.

A number of network simulators, both commercial and university research projects, have been developed [6, 8]. Well known commercial simulators include BoNeS from Cadence, COMNET from CACI, and Opnet from Mil3 [2]. None of these tools have been used for the large-scale simulations described in this paper. Widely used public domain simulators include NS developed at LBNL, and this is being used to develop an Internet simulator called VINT at ISI [8]. NS is basically a transport-level simulator that supports several flavors of TCP (include SACK, Tahoe and Reno) and router scheduling algorithms. Models can be described using a variation of the Tool Command Language, Tcl. The VINT effort is a recent start that aims to develop a comprehensive simulator for the Internet; it does not address the use of parallel execution, and to the best of our knowledge has not been used for detailed high fidelity simulation of large networks.

The next section gives an overview of the parallel library of network models that is being developed and presents the results of a few selected experiments to demonstrate the scalability of the simulator to very large networks. The following sections present case studies in the use of the simulator to evaluate the performance of a real-world application – replicated file systems in a wireless environment.

This research was supported by ARPA/CSTO under Contract DABT-63-94-C-0880, and by ARPA under Contract DAAB-07-97-C-D321.

II. GloMoSim Library

GloMoSim (for Global Mobile Information System Simulator) is a scalable simulation library for wireless network systems [5]. It is built on top of the PARSEC simulation environment [3] that provides parallel discrete-event simulation capability. In contrast to existing network simulators such as OPNET and NS, GloMoSim has been designed and built with the primary goal of simulating very large network models that can scale up to a million nodes using parallel simulation to significantly reduce execution times of the simulation model.

As most network systems adopt a layered architecture, GloMoSim is being designed using a layered approach similar to the OSI seven-layer network architecture. Simple APIs are defined between different simulation layers. This allows the rapid integration of models developed at different layers by different people. Actual operational code can also be easily integrated into GloMoSim with this layered design, which is ideal for a simulation model as it has already been validated in real life and no abstraction is introduced. For example, a TCP model was implemented in GloMoSim by extracting actual code from the FreeBSD operating system. This also reduces the amount of coding required to develop the model.

A common API between every two neighboring models on protocol stacks is predefined to support their composition. These APIs specify parameter exchanges and services between neighboring layers. For example, interfaces between the Data Link (MAC) layer and the network layer are defined as messages with the following formats in the simulation library:

Packet Handling APIs:
Data packet from network to MAC layer for transmission:
Packet_network_to_mac (destId, payload, packetSize)
Data packet from MAC to network layer on reception:
Packet_mac_to_network (sourceId, payload, packetSize)

For outgoing packets sent from the network to the MAC layer, the *destId* field refers to the next hop for this particular packet. For incoming packets sent from the MAC to the network, the *sourceId* field refers to the previous hop on which this packet arrived. The *payload* and *packetSize* fields refer to the actual data and the size of the data that is being received or sent for all packets. Each protocol module at a given layer is required to comply with the APIs defined for that layer.

Table I lists the GloMoSim models currently available at each of the major layers. GloMoSim also supports two different mobility models. Nodes can move according to a model that is generally referred to as the "random waypoint" model [7]. A node chooses a random destination within the simulated terrain and moves to that location based on the speed specified in the configuration file. After reaching its destination, the node pauses for a duration that is also specified in the configuration file. The other mobility model in GloMoSim is referred to as the "random drunken" model. A node periodically moves to a position chosen randomly from its immediate neighboring positions. The frequency of the change in node position is based on a parameter specified in the configuration file.

TABLE I
MODELS CURRENTLY IN THE GLOMOSIM LIBRARY

Layer	Models
Physical (Radio propagation)	Free space, Rayleigh, Ricean, SIRCIM
Data Link (MAC)	CSMA, MACA, MACAW, FAMA, 802.11
Network (Routing)	Flooding, Bellman-Ford, OSPF, DSR, WRP, LAR, DREAM, Fisheye
Transport	TCP, UDP
Application	Tcplib, Synthetic, Replicated File System

In PARSEC, a simple approach to designing a network simulation model is to create each network node as an entity. Although this approach is easy to understand, it has scalability problems. If an entity has to be instantiated for each node, the memory requirements would increase dramatically for a model with a large number of nodes because each entity requires additional memory to work as an independent process. The performance of the simulation would also degrade due to context switching overheads among many entities. Hence, initializing each node as a separate entity inherently limits the scalability and performance of the simulation.

To circumvent these problems, node aggregation was introduced into GloMoSim. With node aggregation, a single entity can simulate several network nodes in the system. A separate data structure representing the complete state of each node is maintained within the entity. When the simulation code for a particular node is being executed, it does not have access to the data structures of other nodes in the simulation. The node aggregation technique implies that the number of nodes in the system can be increased while maintaining the same number of entities in the simulation. In fact, the only requirement is that we need only as many entities as the number of processors on which the simulation is being run. Hence, a sequential simulation needs only one entity in the simulation. With the node aggregation technique, the memory and context switching problems are eliminated.

In GloMoSim, each entity represents a geographical area of the simulation. Hence, the network nodes that a particular entity represents are determined by the physical position of the nodes. For example, suppose we specify a geographical area of 100 by 100 meters in the simulation and set the number of x and y partitions to be 2 for a particular simulation. There would be four partitions in the simulation, where each partition is represented by a single entity. Fig. 2 shows how the terrain would be divided into the four partitions. One particular partition in the simulation would encompass the area represented by the coordinates (0, 0), (49, 0), (0, 49), and (49, 49).

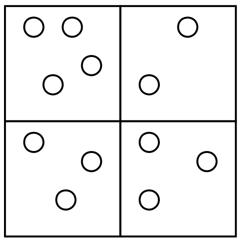

Fig. 2: A simulation with geographical area of 100 by 100 meters, which is divided into four partitions.

In a basic usage of GloMoSim, each entity represents a regular rectangular region (partition). Thus, a partition can have, at most, eight neighboring partitions. When a network node sends out a message, the message has to be sent to at most the eight neighboring entities in the simulation. This is much simpler than the simple design mentioned previously. If each entity represents a single network node, broadcasting a message from a node is very difficult. The first option to implementing broadcasting is that each entity constantly keeps track of the other entities that are within its power range. This option is difficult since the network topology would constantly change as mobility is introduced into the simulation. The second option is that when a node sends a message, it would be sent to all the other entities in the simulation. The receiving entity will accept the message as long as it is in the power range of the sender. This becomes highly inefficient as the number of nodes in the simulation increases. Hence a simple message transmission becomes very complicated when node aggregation is not used. With node aggregation, each entity can examine which node can receive a packet within the entity and send messages only to the neighboring entities where the packet can be reached.

The node aggregation technique gives another benefit to the simulation performance. As each entity needs to examine packet receptions only for the nodes located in the region it is simulating, using many partitions reduces the total search space for packet delivery. In Fig. 3, for instance, if a packet sent by a node located in Partition (0, 0) cannot be reachable to the boarder of the partition, no message needs to be sent to the other partitions. Therefore, the other partitions do not have to examine the reception of the packet, which reduces the region to be examined for the packet by a factor of four compared to using single partition. Fig. 3 shows the impact of multiple partitions for the models with 2500 and 5000 wireless nodes. Both simulation models consist of wireless nodes running CSMA at the MAC layer, each of which is randomly placed in a 2000 x 2000m free-space region. As seen in Fig. 3, the executions for both models become faster as the number of partitions increases. The effect of multiple partitions is larger for the model with 5000 nodes as the reduction in the execution time is related to the number of wireless nodes to be examined for each radio transmission.

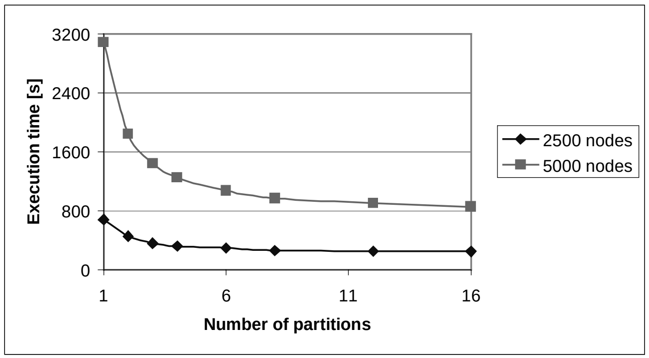

Fig. 3: Execution times against the number of partitions.

It might appear that the addition of node aggregation would cause difficulty for protocol developers who are only interested in developing the simulation model for their particular model. But this is not the case, as we have created several layers of abstractions in GloMoSim. For the most part, the developer writes pure C code. The presence of the PARSEC runtime and interactions with the runtime are completely hidden from the user. In the experience of our own group, modelers prefer to work within our structured environment of node aggregation rather than the alternative.

GloMoSim is aimed at simulating models that may contain as many as 100,000 mobile nodes with a reasonable execution time. GloMoSim has already been used to simulate 10,000 nodes up to the MAC layer using parallel execution of the model on shared memory architectures. Fig. 4 indicates parallel performance of GloMoSim on a Sun SPARCserver 1000. Speedup rates are calculated based on the sequential execution of each model. The same configuration as the experiments on multiple partitions is used with a different number of wireless nodes. Twelve partitions are used for all the executions to balance the workload of each processor. As shown in Fig. 4, GloMoSim achieved better parallel performance for models with a higher number of wireless nodes because more activities occur concurrently in those models, which increases the parallelism of models.

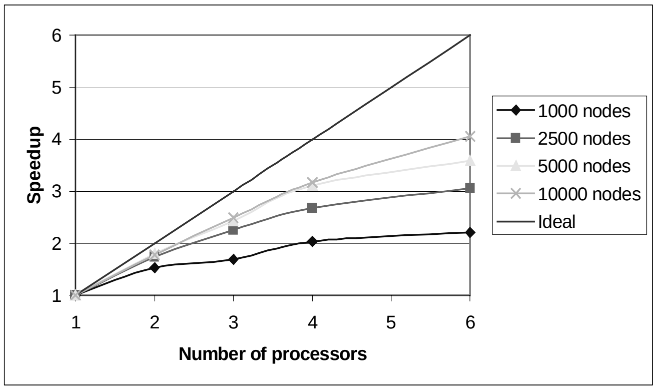

Fig 4: Parallel performance with different number of nodes.

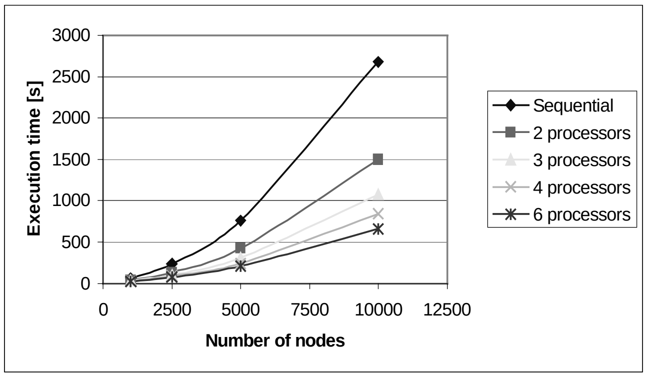

Fig 5: Execution times with varying number of nodes.

Users, especially those who need to simulate large-scale models, can benefit from this parallel simulation capability of GloMoSim. Fig. 5 shows increases of execution times against the number of mobile nodes in the model. With the same number of processors, the execution time increases dramatically as the number of mobile nodes in the model increases. However, the execution time with six processors for the 10000 node model is shorter than the sequential execution for the 5000 node model. This implies that the user can run a simulation model consisting of twice the number of mobile nodes in the same amount of time with six processors.

Parallel simulation requires synchronization of the simulation clock among multiple processors. The PARSEC simulation environment provides four variations of conservative protocols and an optimistic protocol for the synchronization. The current GloMoSim kernel has parallel execution directives for conservative protocols and will be capable of executing models using optimistic protocols in the future. The experiments done in this section used the null message protocol, which is one of the most basic conservative protocols widely used for many applications.

III. Case Study: Simulation Of A Replicated File System

Optimistically replicated file systems have been suggested as an effective method to provide users access to shared files, even when they are temporarily disconnected [10]. Such file systems work on the following principles: replicas, or copies, of a file are stored on multiple computers. Using optimistic replication, a user’s read or write operations are directed to the local replica. This can lead to file consistency problems when concurrent updates are made to multiple replicas of a given file. A process called reconciliation, which involves two replicas on separate computers, addresses such inconsistencies. During reconciliation, replicas are compared and updates are propagated between the two computers. When a server generates a reconciliation request, it will also specify the target server based on the reconciliation topology, which is distinct from the physical topology in which the servers may be organized. Examples of reconciliation topology include tree, ring, and star topologies.

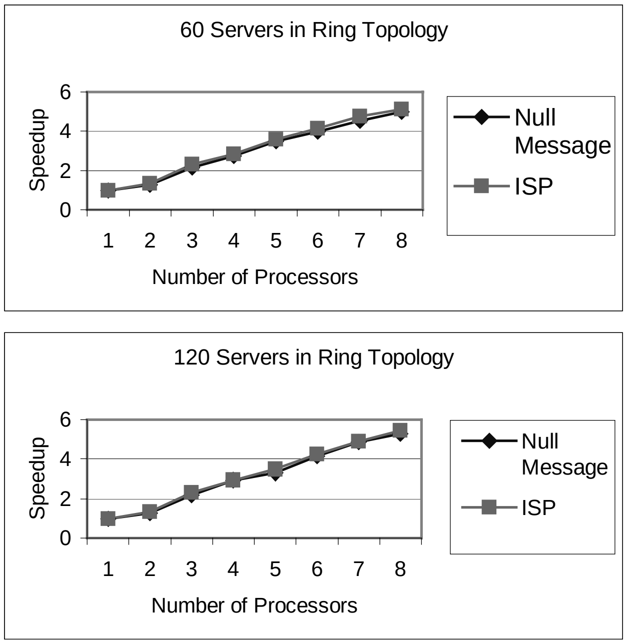

Fig. 6: The speedups with ring topology for 60 and 120 servers, in comparison to the global event list algorithm.

The design of a scalable replicated file system involves a number of considerations. At the fundamental level, it is necessary to identify the costs paid due to greater availability and reliability, effects of file access patterns on performance, impact of data exchange method, data propagation topology, data synchronization interval, and lastly, how all these metrics change as the system is scaled up to hundreds of replicas. Some of the common metrics that have been proposed to evaluate the quality of service offered by a replicated file system include the stale read and write rate as well as cost measurements of CPU and network bandwidth usage. A stale read or write occurs when a user reads or writes out-of-date data [11]. A simulation model [11] for an optimistically replicated file system called Rumor [9] was initially developed using the PARSEC simulation environment by researchers in the OS group to investigate design alternatives for a small number of replicas. As the system was scaled up to hundreds of replicas and a detailed model of the network stack replaced the abstract network model, the execution time of the model was found to increase dramatically.

The Rumor simulation with an abstract model of the network was subsequently parallelized in an attempt to reduce its execution time [4]. The speedup for the ring reconciliation topology with respect to the global event list algorithm is shown in Fig. 6. The results with the null message protocol, as well as the Ideal Simulation Protocol (ISP), are presented. ISP is provided in PARSEC to measure the parallelism available in a simulation model. It does this by first taking a complete program trace. ISP provides a lower bound on the execution time of a model by executing it with perfect knowledge about messages that are safe to process [2]. The partitioning strategy for a ring topology is simple: a contiguous subset of the servers is placed on each processor. With eight processors, speedup of close to 5 is achieved for both the 60 and 120 server case. The system also scales nicely, as the speedups for the 60 and 120 server case are very similar. The speedups achieved with the null message algorithm and ISP are very similar. This shows that all the parallelism has been extracted from the model.

IV. Case Study 2: Replicated File Systems Using Detailed Network Models

The previous two sections have shown the scalability of GloMoSim and Rumor simulations in a stand-alone mode. But there is also significant interest in using replicated file services in wireless environments for both military and civilian applications. The communication stack in a wireless ad hoc network is extremely complex. Furthermore, with the large number of protocols designed for the wireless communication at various layers in the protocol stack, there are several possible ways of configuring the communication stack. Most protocols have configurable parameters, further increasing the number of possible network configurations. Thus, unlike the case with wired networks where a reasonable number of validated abstract models have been developed for different scenarios, an abstract network model cannot adequately represent the inherent complexities of a wireless communication stack. To obtain accurate simulation results for data replication services, it is imperative to model in detail all the layers in the communication stack from the physical to the application layer. It has been shown that the performance of the Rumor application is sensitive to the choice of lower level wireless protocols and to the selection of appropriate parameters for a specific protocol (e.g., the size of TCP retransmission window at the transport layer) [1].

Fig. 7 shows the impact of MAC layer protocol choice on the average reconciliation time. In the absence of mobility, the average reconciliation time is almost independent of the specific MAC layer protocol. With the introduction of mobility, there are noticeable differences in the performance of the replication service under different MAC layer protocols. MACA performs the worst, as it does not sense the carrier when it needs to transmit data. If a particular node is receiving data and needs to transmit at the same time, the incoming packet is lost [1].

V. Conclusion

Detailed high-fidelity models of large networks represent a significant challenge for the networking community. As the military moves towards deploying a digital communication infrastructure, it is imperative that the performance of the communication devices be thoroughly studied prior to deployment in order to understand the limits of the network and its ability to handle diverse traffic under stringent operating conditions. This paper presented a simulation library called GloMoSim whose goal is to support accurate performance prediction of large-scale network models using parallel execution on a diverse set of parallel computers. The library has already been used to simulate networks with thousands of wireless nodes and provides a rich set of models for both existing and novel protocols at multiple layers of the protocol stack. It has been used to undertake numerous performance studies among alternative protocols both at UCLA and other organizations. The GloMoSim library is currently available for download at the following Web site: http://pcl.cs.ucla.edu/projects/domains/glomosim.html

Refrences

[1] R. Ahuja, R. Bagrodia, L. Bajaj, and M. Takai, "Evaluation of Optimistic File Replication In Wireless Multiphop Networks," unpublished.
[2] H. Akhtar, "An Overview of Some Network Modeling, Simulation, & Performance Analysis Tools," Proceedings of 2nd IEEE Symposium on Computers and Communications, 1997.
[3] R. Bagrodia, R. Meyer, et al., "PARSEC: A Parallel Simulation Environment for Complex Systems," IEEE Computer, October 1998.
[4] L. Bajaj, R. Bagrodia, and R. Meyer, "Case Study: Paraellelizing a Sequential Simulation Model," Proceedings of the 13th Workshop on Parallel and Distributed Simulations, PADS 1999.
[5] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla, "GloMoSim: A Scalable Network Simulation Environment," Technical Report, UCLA Computer Science Department – 990027.
[6] S. Bhatt, R. Fujimoto, A. Ogieski, and K. Perumalla, "Parallel Simulation Techniques for Large-Scale Networks," IEEE Communication Magazine, August 1998, pp. 42-47.
[7] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," In *Mobile Computing*, edited by Tomasz Imielinski and Hank Korth, chapter 5, pages 153-181. Kluwer Academic Publishers, 1996.
[8] S. McCanne and S. Floyd, UCB/LBNL/VINT Network Simulator – NS (version 2), http://www-mash.cs.berkeley.edu/ns/
[9] P. Reiher, G. Popek, M. Gunter, J. Salomone, D. Ratner, "Peer-to-Peer Reconciliation Based Replication for Mobile Computers," European Conference on Object Oriented Programming, Second Workshop on Mobility and Replication, June 1996.
[10] M. Satyanarayanan, "Coda: A Highly Available File System for a Disconnected Workstation Environment," Proceedings of the 2nd Workshop on Workstation Operating Systems, September 1989.
[11] A. Wang, P. Reiher, R. Bagrodia, and G. Popek, "A Simulation Evaluation of Optimistic Replicated Filing in a Mobile Environment," Proceedings of the 18th IEEE International Performance, Computing, and Communications Conference, February 1999.

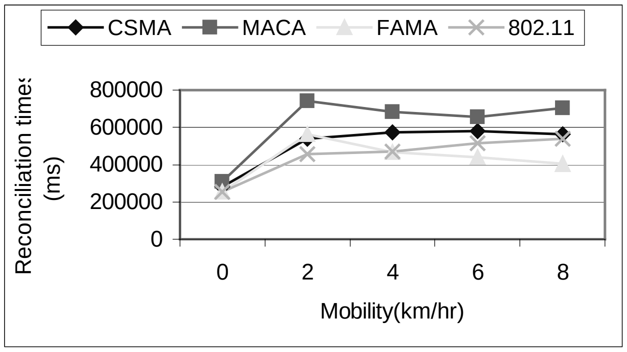

Fig. 7: Average reconciliation time as mobility speed is varied for different MAC layer protocols.