# On Network Model Division Method based on Link-to-Link Traffic Intensity for Accelarating Parallel Distributed Simulation

Hiroyuki Ohsaki, Shinpei Yoshida, and Makoto Imase

Department of Information Networking
Graduate School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita, Osaka 565-0871, Japan
{oosaki,s-yosida,imase}@ist.osaka-u.ac.jp

**Abstract.** In recent years, requirements for performance evaluation techniques of a large-scale network have been increasing. However, the conventional network performance evaluation techniques, such as mathematical modeling and simulation, are suitable for comparatively small-scale networks. Research on parallel simulation has been actively done in recent years, which might be a possible solution for simulating a large-scale network. However, since most existing network simulators are event-driven, parallelization of a network simulator is not easy task. In this paper, a novel network model division method based on link-to-link traffic intensity for accelerating parallel simulator of a large-scale network is proposed. The key ideas of our network model division method are as follows: (1) perform steady state analysis for the network model that is to be simulated, and estimate all traffic intensities along links in steady state, (2) repeatedly apply the minimum cut algorithm from graph theory based on the estimated traffic intensities, so that the simulation model is divided at the link that has little traffic intensities in steady state.

## 1 Introduction

In recent years, demand for a technique to evaluate the performance of large-scale networks has heightened [1, 2] along with the increasing size and complexity of the Internet. The Internet today is a best-effort network, and communication quality between ends is in no way guaranteed. Of course, robustness to some extent has been achieved through use of dynamic routing like OSPF and BGP even with the current Internet. However, the Internet itself is indispensable as society's communication infrastructure, so a technique to evaluate the performance of large-scale networks is in strong demand to ensure the reliability, safety, and robustness of networks, to allow future network expandability and design, and to assess the impact of terrorism and natural disasters.

However, conventional techniques to evaluate the performance of a network such as numerical analysis techniques and simulation techniques are directed toward relatively small-scale networks. As an example, queuing theory [3] as has been widely used in performance evaluation of conventional computer networks is not readily applied to performance evaluation of the large-scale and complex Internet. When strictly analyzing the performance of a network using queuing theory, the number of states for analysis

increases tremendously together with the increase in the number of nodes connected to the network.

Techniques to approximately analyze interconnected networks like Jackson networks have been proposed even in queuing theory [3], although the packet arriving at a node is assumed to be a Poisson arrival. However, TCP/IP, the communication protocol for the Internet, is a complex, layered communication protocol with a complex traffic control algorithm and routing algorithm. As an example, the Internet uses various underlying communication protocols such as Ethernet, FDDI, and ATM, and creation of a rigorous numerical model of a complex system like this is not possible in realistic terms. Of course, numerical analysis techniques are extremely advantageous in terms of calculating time, so their use as a method of complementing other performance evaluation techniques is vital.

Simulation techniques, as opposed to numerical analysis techniques, allow performance evaluation of complex networks [4]. Performance evaluation of medium-scale networks in particular has become possible through the increasing speeds and capacities of computers in recent years. However, communication protocols for the Internet are extremely complex, so massive computer resources are required for simulation of networks, and simulation of large-scale networks is still difficult. The majority of network simulators widely used today simulate behavior at the packet level, so they use an event-driven architecture. A technique for faster speeds of network simulators operating on a single computer has also been proposed [5], although a different approach is needed to simulate a large-scale network.

Research with regard to parallel simulations as technology to allow simulation of large-scale networks has been conducted in recent years [6-8]. Construction of relatively inexpensive cluster computers has become easier through the faster speeds and lower prices of desktop computers and the spread of high-speed network interfaces such as Gigabit Ethernet. In addition, Grid computing using a wide-area network to integrate computer resources around the world has also attracted attention. However, the majority of network simulators have an event-driven architecture, so parallelization of network simulators is difficult.

Thus, this paper proposes division of a network model based on the traffic volume between links in order to run a simulation of a large-scale network at high speeds in a distributed computing environment and evaluate its effectiveness. The basic idea for the proposed division of a network model is as follows:

**(1)** Steady state analysis as proposed in the literature [9] would be performed on a network model (simulation model) to simulate, and the traffic volume passing through links in a steady state would be estimated.
**(2)** The simulation model would be divided by links with a low traffic volume using the minimum cut algorithm in the literature [10] based on the estimated traffic volume.
**(3)** The simulation model would be divided into $N$ portions by repeatedly performing (1) and (2) so that the total traffic volume passing through nodes would be equal.

The simulation model would be divided into $N$ portions via the aforementioned division and respective sub-network models would be run on $N$ computers.

The composition of this paper is as follows. First, Section 2 describes related research regarding parallel simulation of networks. Section 3 explains division of a net-

work model based on the traffic volume between links proposed. Section 4 indicates examples of the proposed division of a network model. In addition, Section 5 describes evaluation via a simple experiment of how much faster the parallel simulation would be through the proposed division of a network model. Finally, Section 6 describes this paper's conclusions and future topics for research.

## 2 Related research

QualNet [11], OPNET [12], and PDNS [13] are typical network simulators that support parallel simulation. QualNet is a commercial simulator from Scalable Network Technologies and can be run on a single SMP (Symmetric Multi-Processing) computer [11]. Division (Smart Partitioning) of a simulation model, load dispersion (Load Balancing) per CPU, and maximized simulation look-ahead (Maximization of Lookahead) are techniques used to increase the speed of parallel simulation. However, it cannot be run on multiple computers such as cluster computers and cannot be used for simulation of large-scale networks.

OPNET is a commercial simulator from OPNET Technologies, and it can be run on a single SMP computer, although it cannot be run on multiple computers like cluster computers [12]. In addition, parallel simulation is only possible for specific modules for wireless networks, and the simulator cannot be used for simulation of large-scale networks.

PDNS [Parallel/Distributed NS] is a network simulator that was developed by the PADS research group at the Georgia Institute of Technology [13]. PDNS is an extension of the ns2 simulator [14] as is widely used in performance evaluation of TCP/IP networks and is run on parallel computers. With PDNS, simulation nodes can be distributed and run on different computers. As a parallel simulator, however, only extremely limited features have been implemented. When simply running a simulation of a large-scale network on multiple computers, simulation speed slows substantially due to overhead from communication between computers performing the simulation, a problem that has been pointed out [13]. Accordingly, performing simulation of large-scale networks is also difficult using PDNS as-is.

## 3 Division of a network model based on the traffic volume between links

An overview of the proposed division of a network model will be explained. Below, the model of the network as a whole to simulate is called the "network model," and the models obtained by division of the network model are called "sub-network models." First, the network model to simulate is expressed in a weighted, undirected graph. The graph's vertices correspond to nodes (routers or terminals) and edges of the graph correspond to links between nodes. The traffic volume passing through a link in a steady state is used as the weight of the graph's edges. The basic idea is (1) to perform steady state analysis as proposed in the literature [9] on a network model (simulation model) to simulate and estimate the traffic volume passing through links in a steady state, (2) to

divide the simulation model with links with a low traffic volume using a minimum cut algorithm in the literature [10] based on the estimated traffic volume, and (3) to divide the simulation model into $N$ portions by repeatedly performing (1) and (2) so that the total traffic volume passing through nodes would be equal.

Specifically, the traffic volume passing through each link in a steady state is first derived using steady state analysis proposed in the literature [9] in instances where a network model to simulate and traffic demands between nodes are given. Moreover, several potential cuts in the network model are determined using the minimum cut algorithm proposed in the literature [10]. Of these, the cuts used were those with a small capacity (traffic volume passing between sub-network models) and simulation calculation time for two sub-network models (estimated by the total traffic volume in sub-network models) that is equal to the extent possible.

Next, a division algorithm like that mentioned above is again applied to a network model of individual sub-network models considered to have the maximum simulation calculation time. $N$ sub-network models are obtained by repeating division like that mentioned above $N-1$ times to have a low traffic volume passing between sub-network models (i.e., slight overhead in parallel simulation) and to have an equal simulation calculation time (i.e., the loads on the computers performing the simulation would be equal) for each sub-network model.

Next, the algorithm for the proposed division of a network model is explained. Preceding an explanation of the algorithm, several forms of notation will be defined. A network model is thought of as undirected graph $G = (V, E)$. Here, $V = \{v_1, v_2, \ldots, v_n\}$ and $E = \{e_1, e_2, \ldots, e_m\}$. The weight of an edge $(v_i, v_j)$ is $w_{i,j}$. Furthermore, the total number of divisions of the network model (the number of sub-network models) is $N$. In addition, the traffic model used in simulation is denoted by traffic matrixes $L = (l_{i,j})$ and $M = (m_{i,j})$. Here, $l_{i,j}$ is the transfer rate for UDP traffic from vertex $v_i$ to vertex $v_j$ and $m_{i,j}$ is the number of TCP connections from vertex $v_i$ to vertex $v_j$. This paper deals with TCP traffic and UDP traffic to continuously transfer data for the sake of simplicity.

The algorithm for the proposed division of a network model is as follows:

1. Derivation of the traffic volume between links by steady state analysis
   Steady state analysis of network model $G$ and traffic matrices $L$ and $M$ are performed, and the traffic volume between links in a steady state is derived. The analysis technique proposed in the literature [9] is used for steady state analysis of the network. Thus, throughput for TCP traffic $T_{i,j}$ in a steady state and throughput for UDP traffic $L_{i,j}$ are determined. Here, $T_{i,j}$ and $L_{i,j}$ are throughput for TCP and UDP traffic passing through an edge $(v_i, v_j)$ in a steady state.
2. Determination of the weight of the edges $w_{i,j}$
   The weight $w_{i,j}$ of an edge $(v_i, v_j)$ is defined as follows:

$$w_{i,j} = \sum_{l} \sum_{m} C_{l,m} T_{l,m} + \sum_{l} \sum_{m} D_{l,m} L_{l,m} \tag{1}$$

   Here, if $m_{i,j}$ passes through edge $(v_i, v_j)$ or edge $(v_j, v_i)$, $C_{i,j}$ is 1; otherwise, it is 0. If, in addition, $l_{i,j}$ passes through edge $(v_i, v_j)$ or edge $(v_j, v_i)$, $D_{i,j}$ is 1;

otherwise, it is 0. Thus, weight $w_{i,j}$ means the sum of the throughput for all traffic passing through edge $(v_i, v_j)$ and edge $(v_j, v_i)$ in a steady state.

3. Initialization of the set $M$ of subgraphs
   The set of subgraphs obtained by division is initialized via network model $G$.

$$M \leftarrow \{G\} \tag{2}$$

4. Model division using a minimum cut algorithm
   The number of divisions of the network model is $N$. The following process is performed repeatedly until $|M| = N$.
   (a) A sum of the weights of the edges $W(E)$ from the set $M$ of subgraphs where the maximum subgraph $G' = (V', E')$ is selected. The sum of the weights of the edges $W(E)$ in a weighted, undirected graph $G = (V, E)$ is defined by the following equation.

$$W(E) = \sum_{(v_i, v_j) \in E} w_{i,j} \tag{3}$$

   (b) A minimum cut algorithm proposed in the literature [10] is run on weighted, undirected graph $G$. Thus, $|V'| - 1$ cuts $(S, \overline{S})$ are obtained. Here, the cut capacity is denoted as the $n$ th small cut $(S_n, \overline{S}_n)$ $(1 \leq n \leq |V'| - 1)$
   (c) Subgraphs with a small cut capacity and equal sum of the weights of the edges to the extent possible are selected from $(S_n, \overline{S}_n)$. Specifically, Subgraphs $S_n$, $\overline{S}_n$ are selected so that

$$\frac{|W(S_n) - W(\overline{S}_n)|}{W(S_n) + W(\overline{S}_n)} \leq \alpha \tag{4}$$

   ($\alpha$ is a constant) is fulfilled and $n$ is a minimum (i.e., a minimum cut capacity). Then, $G'$ in the set $M$ for subgraphs is replaced by $\{S_n, \overline{S}_n\}$.

The division of a network model as proposed in this paper has the following characteristics. First, the algorithm for the proposed division is a heuristic algorithm and uses a minimum cut algorithm in graph theory. In addition, calculations required for simulation of each sub-network model are estimated by calculating the sum of the weights of all edges $W(E)$ during division into sub-network models. Thus, calculations required for simulation of each sub-network model can be expected to be equal, as opposed to division simply using the traffic volume between links $T_{i,j}$ and $L_{i,j}$. The proposed division of a network model assumes steady, continuous TCP and UDP traffic and cannot handle traffic in bursts. In addition, calculations required for simulation of a sub-network model are estimated using weight $W(E)$, although validation of this method of estimation is required.

## 4 Examples of the division of a network model proposed

This section indicates examples of the division of a network model proposed. Here, an example of a network model with 16 nodes and a mean degree of 3 as in Fig. 1 is

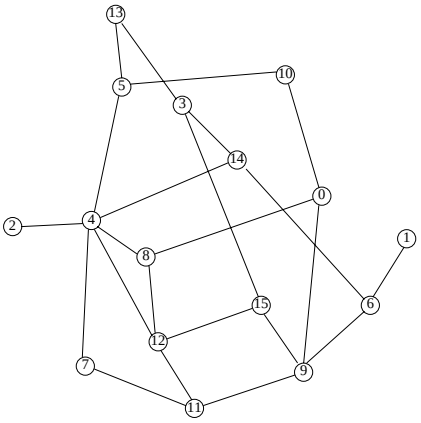


**Fig. 1.** Example of division of a network model (before an algorithm is run)

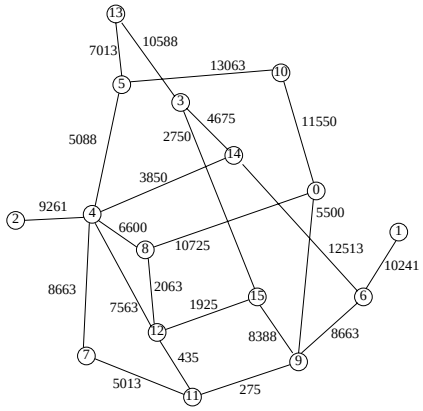


**Fig. 2.** Example of division of a network model (the weight of the edges $w_{i,j}$ is calculated from steady state analysis; the value for each edge is the traffic volume passing through a link [Kbyte/s])

used. The bandwidth for each link is a random value from 1 to 100 [Mbits/s], and the propagation delay for each link is a random value from 10 to 200 [ms]. In addition, the network model's number of divisions is $N = 4$ considering the fact that the simulation was performed on four parallel computers. Here, results are shown for when 1000 TCP connections were generated randomly.

With respect to Fig. 1, steady state analysis from the literature [9] is performed, and the throughput of respective traffic $T_{i,j}$ and $L_{i,j}$ passing through each link in a steady state is derived. Based on this, the weight of each edge $w_{i,j}$ is calculated (Fig. 2) from Eq. (1).

The minimum cut algorithm in the literature [10] is run on the weighted, undirected graph $G' = (V', E')$ in Fig. 2, and $|V'| - 1 = 15$ cuts $(S, \overline{S})$ is obtained. Of these cuts, those for which the sum of the weight of the edges $W(S)$ and $W(\overline{S})$ fulfills Eq. (4) in subgraphs $S$ and $\overline{S}$ those with a minimum cut capacity is applied (Fig. 3). In this example, $(S, \overline{S}) = (\{2, 4, 7, 8, 11, 12\}, \{0, 1, 3, 5, 6, 9, 10, 13, 14, 15\})$ and the cut is applied so that the cut capacity will be 21,863 [Kbyte/s], $W(S)$ = 39,598 [Kbyte/s], and $W(\overline{S})$ = 94,944 [Kbyte/s].

In Fig. 3, $W(S) < W(\overline{S})$, so $\overline{S} = \{0, 1, 3, 5, 6, 9, 10, 13, 14, 15\}$ is further divided into two sub-network models (Fig. 4). In this example, $(T, \overline{T}) = (\{1, 3, 6, 9, 13, 14, 15\}, \{0, 5, 10\})$, and the cut is applied so that the cut capacity will be 12,513 [Kbyte/s], $W(T)$ = 57,818 [Kbyte/s], and $W(\overline{T})$ = 24,613 [Kbyte/s].

Moreover, $N$ (= 4) sub-network models are obtained by repeating the same procedure. Here, $W(T) > W(\overline{T})$, so $T = \{1, 3, 6, 9, 13, 14, 15\}$ is further divided into two sub-network models (Fig. 5). In this example, $(U, \overline{U}) = (\{3, 13, 14\}, \{1, 6, 9, 15\})$, and the cut is applied so that the cut capacity will be 15,263 [Kbyte/s], $W(U)$ = 15,263 [Kbyte/s], $W(\overline{U})$ = 27,292 [Kbyte/s].

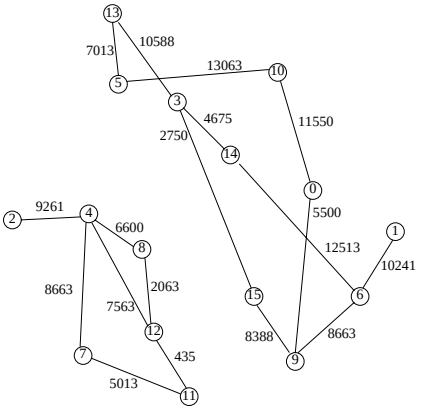


**Fig. 3.** Example of division of a network model (divided into two sub-network models using a minimum cut algorithm. The cut $(S, \overline{S}) = (\{2, 4, 7, 8, 11, 12\}, \{0, 1, 3, 5, 6, 9, 10, 13, 14, 15\})$ is applied)

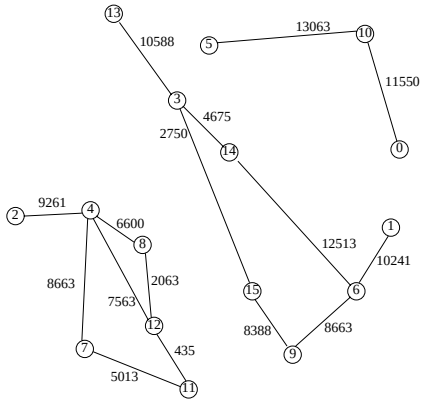


**Fig. 4.** Example of division of a network model ($W(S) < W(\overline{S})$, so $\overline{S} = \{0, 1, 3, 5, 6, 9, 10, 13, 14, 15\}$ is further divided into two sub-network models. The cut $(T, \overline{T}) = (\{1, 3, 6, 9, 13, 14, 15\}, \{0, 5, 10\})$ is applied)

## 5 Evaluation of the division of a network model proposed

This section describes evaluation via a simple experiment of how much faster the parallel simulation would be through the proposed division of a network model. In testing, the running time for parallel simulation (time from the start of simulation until the simulation ended) was measured when a network was divided into several sub-network models using the proposed division method and when a network was randomly divided into sub-network models for balancing the number of nodes in each sub-network model.

In testing, a network model was generated by a random graph of 10 or 100 nodes with a mean degree of 2. Bandwidth for each link in the network model was a random value from 1 to 10 or 100 [Mbits/s], and the propagation delay for each link was a random value from 0.1 to 100 [ms]. In addition, 10 or 100 TCP connections were randomly generated between nodes. Under these conditions, 10 network models were generated, and these were respectively evaluated with regard to when the model was divided into two sub-network models using our proposed division method and when the model was divided simply so that the number of nodes in sub-network models would be equal.

PDNS [13] version 2.27-v1a was used as a parallel network simulator, and simulation was performed for 30 [s]. PDNS version 2.27-v1a's default values were used for the packet length, TCP parameters, router buffer size, and the like. 8 computers with the same performance as shown below were used in testing:

- CPU: Pentium III 1,266 MHz
- Memory: 1,024 Mbyte
- Hard disk: 120 Gbyte

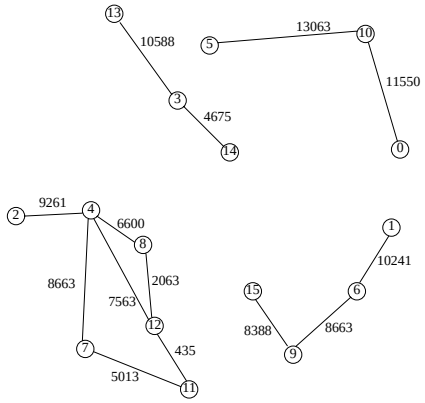


Fig. 5: Example of division of a network model ($W(T) > W(\overline{T})$, so $T = \{1, 3, 6, 9, 13, 14, 15\}$ is further divided into two sub-network models. The cut $(U, \overline{U}) = (\{3, 13, 14\}, \{1, 6, 9, 15\})$ is applied)

– Network: 1 Gbit/s Ethernet
– Operating system: Linux version 2.4.20

Figure 6 shows the total running time required for completing all simulation events for 10 nodes and 1–100 [Mbit/s] link bandwidth. Figure 7 shows the total running time required for completing all simulation events for 100 nodes and 1–10 [Mbit/s] link bandwidth. In these figures, the number of computers running a parallel distributed simulator is changed as 1, 2, 4, 6, and 8, and the parameter $\alpha$ is as 0.1, 0.2, 0.3, 0.4, and 0.5. The results with a random division method are labeled as "random". These figures show with our proposed division method, the total running time becomes about 78%–96% (Fig. 6) and 78%–94% (Fig. 7) of the case with a random division method, indicating significant performance improvement with our proposed division method.

## 6 Conclusions and future topics

This paper proposed division of a network model in order to simulate large-scale networks in a distributed computing environment at high speeds. The proposed division of a network model first derived the traffic volume between links through use of steady state analysis of a network model to simulate. This technique then applies a minimum cut algorithm from graph theory several times in accordance with the traffic volume between links in a steady state and divides a network model into $N$ portions.

Various extensions of the division of a network model for faster parallel simulation as proposed in this paper may be possible in the future. First, this paper dealt with TCP and UDP traffic where data is continuously transferred. Thus, division of a network model can be expanded so as to handle TCP traffic to transfer data in bursts. In addition, this paper dealt with unicast traffic alone, although expansion so as to handle multicast traffic is needed in order to simulate an actual large-scale network.

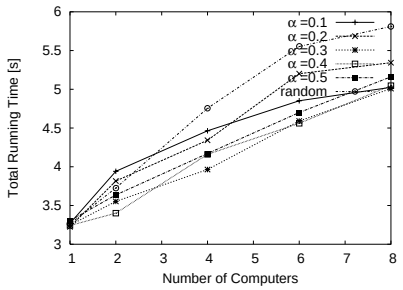


**Fig. 6.** Total running time required for completing all simulation events (10 nodes, degree 2, link bandwidth 1–100 [Mbit/s], link propagation delay 0.1–100 [ms], and 10 TCP connections)

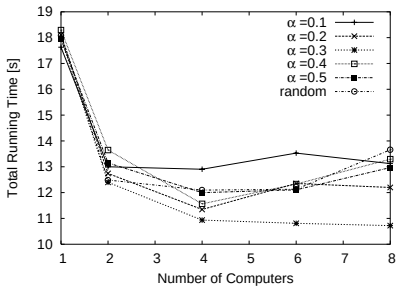


**Fig. 7.** Total running time required for completing all simulation events (100 nodes, degree 2, link bandwidth 1–10 [Mbit/s], link propagation delay 0.1–100 [ms], and 100 TCP connections)

## References


1. Large Scale Networking (LSN) Coordinating Group Of the Interagency Working Group (IWG) for Information Technology Research and Development (IT R&D), *Workshop on New Visions for Large-Scale Networks: Research and Applications*, Mar. 2001. available at `http://www.nitrd.gov/iwg/lsn/lsn-workshop-12mar01/index.html`.
2. S. Floyd and V. Paxson, "Why we don't know how to simulate the Internet," Oct. 1999. available at `http://www.aciri.org/floyd/papers/wsc.ps`.
3. D. Bertsekas and R. Gallager, *Data Networks*. Englewood Cliffs, New Jersey: Prentice-Hall, 1987.
4. A. M. Law and M. G. McComas, "Simulation software for communications networks: the state of the art," *IEEE Communications Magazine*, vol. 32, pp. 44–50, Mar. 1994.
5. V. S. Frost, W. W. Larue, and K. S. Shanmugan, "Efficient techniques for the simulation of computer communications networks," *IEEE Journal of Selected Areas in Comuunications*, vol. 6, pp. 146–157, Jan. 1988.
6. H. T. Mouftah and R. P. Sturgeon, "Distributed discrete event simulation for communications networks," *IEEE Journal on Selected Areas in Communications*, vol. 8, pp. 1723–1734, Dec. 1990.
7. G. F. Riley, R. M. Fujimoto, and M. H. Ammar, "A generic framework for parallelization of network simulations," in *Proceedings of the Seventh International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pp. 128–135, Oct. 1999.
8. S. Bhatt, R. Fujimoto, A. Ogielski, and K. Perumalla, "Paralell simulation techniques for large scale networks," *IEEE Communications Magazine*, vol. 38, pp. 42–47, Aug. 1998.
9. D. Dutta, A. Goel, and J. Heidemann, "Faster network design with scenario pre-filtering," in *Proceedings of the International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pp. 237–246, Oct. 2002.
10. M. Stoer and F. Wagner, "A simple min-cut algorithm," *Journal of the ACM*, vol. 44, pp. 585–591, July 1997.
11. Scalable Network Technologies, Inc., "QualNet." `http://www.scalable-networks.com/`.
12. Opnet Technologies, Inc., "OPNET." `http://www.opnet.com/`.
13. PADS (Parallel and Distributed Simulation) Research Group, "PDNS - Parallel/Distributed NS." `http://www.cc.gatech.edu/computing/compass/pdns/`.
14. "The network simulator – ns2." available at `http://www.isi.edu/nsnam/ns/`.