Modeling, Analysis, and Optimal Routing of Flow-Controlled Communication Networks[1]

Simon S. Lam and Ching-Tarng Hsieh[2]

Department of Computer Sciences
The University of Texas at Austin
Austin, Texas 78712-1188

TR-87-24 June 1987

Abstract

Closed queueing networks have been advocated by several authors to be a more desirable model than open queueing networks (Kleinrock's model) for network design. We compare open and closed network models and demonstrate the accuracy of a particular closed network model with experimental results. The proportional approximation method (PAM) is presented for evaluating performance measures of closed queueing networks. PAM algorithms have computational time and space requirements of $O(KM)$, where M denotes the number of queues and K denotes the number of virtual channels in the network. Thus, PAM is the first (and only) method that can be used for solving industrial-strength network design problems using a closed network model.

We formulate the following optimal routing problem: Find a route for a new virtual channel to be added to a network with existing flow-controlled virtual channels. A fast heuristic algorithm is presented. The algorithm uses PAM and exploits the following empirical observation: The route that maximizes the individual throughput of a virtual channel coincides in most cases with the route that maximizes the total network throughput (this is not true in general). We present statistical results from studies of 100 randomly generated networks to demonstrate the accuracy of PAM algorithms and the effectiveness of the optimal routing algorithm. (Exact solutions obtained by the tree convolution algorithm were used as benchmarks in our statistical studies.)

---

[1]Work supported by National Science Foundation under grant no. ECS-8304734 and grant no. NCR-8613338.

[2]Current address: ATT Bell Laboratories, Naperville, Illinois 60566.

1. Introduction

Consider a store-and-forward communication network which accepts data packets from host computers and terminals for delivery to other host computers and terminals. (See Figure 1.) As each packet is routed through the network, from its source to its destination, it joins queues inside network switching nodes for transmission over communication links. A queueing network model is most appropriate for the analysis and design of such communication networks.

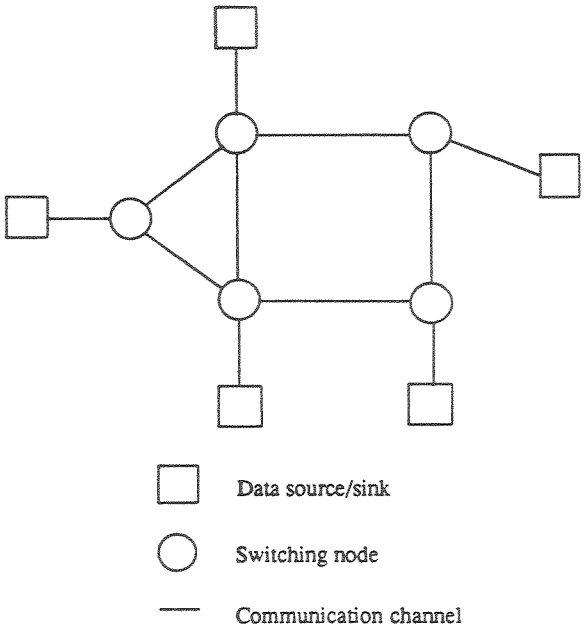

Figure 1. A store-and-forward communication network.

Analytic tools for the design of packet networks have been based primarily on the model of Kleinrock [6, 7], which is an open network of M/M/1 queues. Kleinrock's model will be referred to as the open network model. It is *open* because sources are assumed to generate new packets according to Poisson processes at constant rates and *all* such Poisson arrivals are assumed to be accepted by the communication network without any input control. In practice, most packet networks implement window protocols for flow control across logical connections between pairs of communicating sources and destinations [1, 2, 18]. In this paper, we shall refer to such logical connections as *virtual channels*.[3] A window protocol limits the number of packets, belonging to a virtual channel, that can be traversing the network at the same time. Such a limit is called the *window size* of the flow-controlled virtual channel.

Flow-controlled packet networks can be more accurately modeled by a *closed* queueing network model than by an open queueing network model. Consider the implementation of a window protocol with window size N_k by placing N_k permits initially at the source of virtual channel k. Whenever the number of permits at the source is nonzero, each packet generated is given a permit and accepted into the network. When the packet reaches its destination, its permit is then carried back to the source by an ack message from the destination to the source. Thus, the N_k permits circulate in the network, carried alternately by data packets and by acks, just like the circulating customers of a closed routing chain in a queueing network [13]. Whenever there is no permit at the source, it is assumed that the source is blocked from

[3] In practice, such flow-controlled logical connections can also be identified in the so-called datagram networks.

generating new packets. (Equivalently, new packets are generated by Poisson processes at constant rates but any new packet that finds no permit at the source is dropped.)

Use of a closed network model for the analysis and design of flow-controlled packet networks has been advocated by Reiser [15], Kobayashi and Gerla [8], Schwartz [18], Lam and Lien [10, 14], and Lam and Hsieh [9], and others. The open network model is accurate for networks that are lightly utilized (when flow control mechanisms have little or no effect on network performance). As a network's offered load increases, the open network model cannot be used to predict the throughputs of virtual channels since it does not model flow control; it also significantly overestimates network delays. In Section 2, we present experimental results to illustrate these observations about the open network model and to demonstrate the accuracy of some closed network models.

A serious drawback of closed network models has been the large computational time and space needed for calculating network performance measures, i.e., throughputs and mean delays of virtual channels. Consider a model of a network with M queues and K virtual channels. Let N_k be the window size of virtual channel k. Both the convolution and MVA computational algorithms have a time requirement of $MK\prod_{k=1}^{K}(N_k+1)$ [16, 17]. Thus the time complexity grows exponentially with K, the number of virtual channels.[4] (The space complexity also grows exponentially with K.) For example, if the window size is 8 for all virtual channels, these algorithms are not applicable to the analysis of networks with more 6 or 7 virtual channels. The tree convolution algorithm was designed to exploit routing information and the sparseness of routes in a communication network and it has been used to solve numerically many network examples with 32 to 50 virtual channels [12]. (The number of virtual channels that can be handled for a specific network depends upon window sizes and the sparseness of routes in the network.) However, it is still not applicable to the analysis of many real-life wide area networks which can be quite large. Even for those networks that can be analyzed, the algorithm is too slow for use in network design and optimization procedures.

We present in Section 3 the *proportional approximation method* (PAM) for analyzing closed multichain queueing networks [5]. PAM is based upon the MVA algorithm and the idea used in the derivation of proportional bounds [4]. Two PAM algorithms are presented. They have time requirements of $(3M+1)K$ and $(5M+2)K$ multiplications (we count multiplications and divisions only and refer to them both as multiplications.) Given these time requirements, networks with thousands of queues and thousands of virtual channels can be analyzed. The accuracy of the PAM algorithms was investigated by randomly generating 100 networks which were solved exactly by the tree convolution algorithm. We found the PAM algorithms to be very accurate (see Section 3). Additional experimental studies of the accuracy of PAM as well as a comparison of PAM algorithms with other approximate solution methods can be found in [5]. We note that all other approximate solution methods of closed queueing networks perform an iterative solution of some modified MVA equations. Many iterations may be needed for a solution to converge. PAM algorithms are noniterative.

In Section 4, the following optimal routing problem is formulated. Consider a network with a fixed topology, known channel capacities, and a set of existing virtual channels; the routes and flow-control window sizes of the virtual channels are also known. Find a route for a new virtual channel to be added to the network between a given pair of source and destination. The objective is to maximize the total throughput of the network including the new virtual channel. We present a fast heuristic algorithm for the

---

[4]Time requirements are counted in number of multiplications and divisions only.

above optimal routing problem. The network throughput and virtual channel throughputs are evaluated using a PAM algorithm. The key idea used in the design of the heuristic is the following: The route that maximizes the individual throughput of a virtual channel coincides in most cases with the route that maximizes the total network throughput (even though this is not true in general for *all* cases). The effectiveness of the heuristic algorithm was examined using the 100 randomly generated networks and exact solutions provided by the tree convolution algorithm. Statistics from our experimental study show that the heuristic algorithm either found an optimal route or a route with a throughput close to the maximum almost all the time.

2. Comparison of Open and Closed Network Models

In a queueing network model of packet networks, communication channels and nodal processors are modeled as FIFO servers with exponentially distributed service times.[5] We assume that the packet network has adequate buffers so that buffer overflow at a node has negligible probability. The *independence assumption* of Kleinrock [6] is needed for both open and closed network models.

Let M denote the number of servers and K the number of virtual channels. For simplicity, we assume that each virtual channel has a loop-free fixed route from its source to its destination. (Actually, the routing behavior of a class of customers in queueing networks is specified by a Markov chain, which can be used to specify probabilistic bifurcated routing, i.e., the actual route of a packet is chosen probabilistically from a set of routes [13].)

In the open network model, new packets arrive to the source node of virtual channel k according to a Poisson process at a constant rate of γ_k packets per second. It is assumed that all arrivals to a virtual channel are accepted into the network without any input control.

Consider the closed network model in Figure 2. The route of each virtual channel is *closed* by the addition of two servers outside the communication network boundary. The generation of external arrivals to a virtual channel is modeled with a FIFO server with exponentially distributed service times, called the *source server* of the virtual channel. The service rate of the source server for virtual channel k is γ_k. Imagine that the N_k permits for flow control to be N_k customers circulating in the closed queueing network. The source generates new packets at the rate of γ_k only when the source has permits (i.e., the source server works only when its queue is nonempty). When there is no permit at the source, generation of new packets is blocked (i.e., the source server does not work when its queue is empty).

We model the delay of a permit's return from the sink to the source by a delay server (also called an infinite-server service center in the queueing networks literature). The mean service times of a delay server can be different for different virtual channels. The delay server is an abstraction of end-to-end acks that are sent in a real packet network. It is not important to model the storing and forwarding of end-to-end acks explicitly because these acks are either encoded in data packets or, if they are sent separately, are very short. Thus they consume relatively small amounts of channel capacities, which can be neglected or accounted for separately.

---

[5] In our examples, nodal processors are not modeled because processor delays are typically very much smaller than the communication channel delays in these examples.

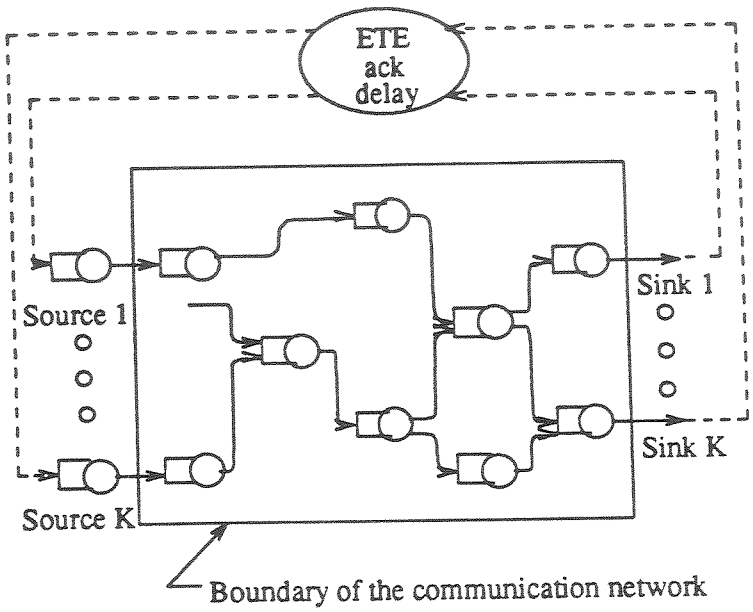

Figure 2. A closed queueing network model of a store-and-forward communication network.

The mean delays of end-to-end acks are not known and can only be estimated. In the experimental study to be presented next, we modeled the end-to-end ack delay of a virtual channel by

(*number of channels in route*)τ

where τ is the average packet transmission time. We found that performance measures of this network example are rather insensitive to the mean end-to-end ack delays. Increasing them all by 50%, or decreasing them all by 50%, varies the performance measures by a maximum of 10%.

We next compare performance predictions of the open network model and the closed network model (described above) with predictions given by a network simulator. The simulator implements most of the important elements of a real network [11]. (See Table 1.) In particular, it implements a data link protocol (similar to Arpanet's). The storing and forwarding of end-to-end acks, both standalone and piggybacked, are explicitly simulated. Virtual channels have a window mechanism for flow control. The length of a packet is sampled from a general probability distribution and remains constant for the entire journey of the packet through the network, i.e., no independence assumption. Key differences between the models are summarized in Table 1. Note that the closed network model, like the open network model, requires the independence assumption and the assumption of exponentially distributed service times. As was observed by Kleinrock many years ago for the open network model [6], we found that the accuracy of throughputs and mean delays of virtual channels predicted by the closed network model does not depend strongly on these two assumptions.

	Simulator	Closed Model	Open Model
Link-level acks	yes	no	no
End-to-end acks	yes	abstraction	no
Window flow control	yes	yes	no
Independence assumption	no	yes	yes
Packet length distribution	Exponential or General	Exponential	Exponential

Table 1. Comparison of model features.

The network used for the comparison of models has 8 switching nodes and 20 full-duplex links. It has 18 virtual channels, in 9 symmetric pairs, randomly chosen between node pairs. The window size of each virtual channel is equal to the number of links in its route. The capacity of each communication link is 10,000 bits/second and the average packet length is 1,000 bits.[6]

The performance predictions of seven models are compared in Figures 3-6. The open network model is labeled OPEN. The closed network model described above is labeled FIXED. The closed network model labeled ITERATIVE employs an iterative solution method; it is described in [9]. The four sets of simulation results correspond to four packet length distributions used by the simulator. EXP denotes the exponential distribution. The other three distributions are specified below:

HYPER a hyperexponential distribution with coefficient of variation equal to 1.40

BURST1 a distribution with 0.3 probability at 100 bits, 0.3 probability at 1900 bits, and 0.4 probability uniformly distributed between 100 bits and 1900 bits; the coefficient of variation is equal to 0.77

BURST2 a distribution with 0.4 probability at 50 bits, 0.4 probability at 1950 bits, and 0.2 probability uniformly distributed between 50 bits and 1905 bits; the coefficient of variation is equal to 0.88

Note that the exponential distribution has a coefficient of variation equal to 1. Each virtual channel in the network example has the same packet arrival rate of γ at the source. Thus each source server in closed network models has a mean service time of $1/\gamma$. The exponential distribution was used in the simulator to generate interarrival times of packets to sources, the same as what is assumed in the queueing network models (i.e., Poisson arrivals).

Figure 3 shows the network throughput as a function of γ. The closed network models and the four simulation models give predictions that are very close to each other. The open network model assumes that the throughput of each virtual channel is γ. At γ=4, one of the servers in the network saturates, i.e., its packet arrival rate exceeds its service rate. In plotting the throughput curve labeled OPEN, we have assumed that γ can still be increased for those virtual channels that do not visit a saturated server.

[6]The reader is referred to Chapter 3 of [3] for a full description of the network, for numerical values of performance measures shown in Figures 3-6, as well as for a study of confidence intervals of the simulator's predictions.

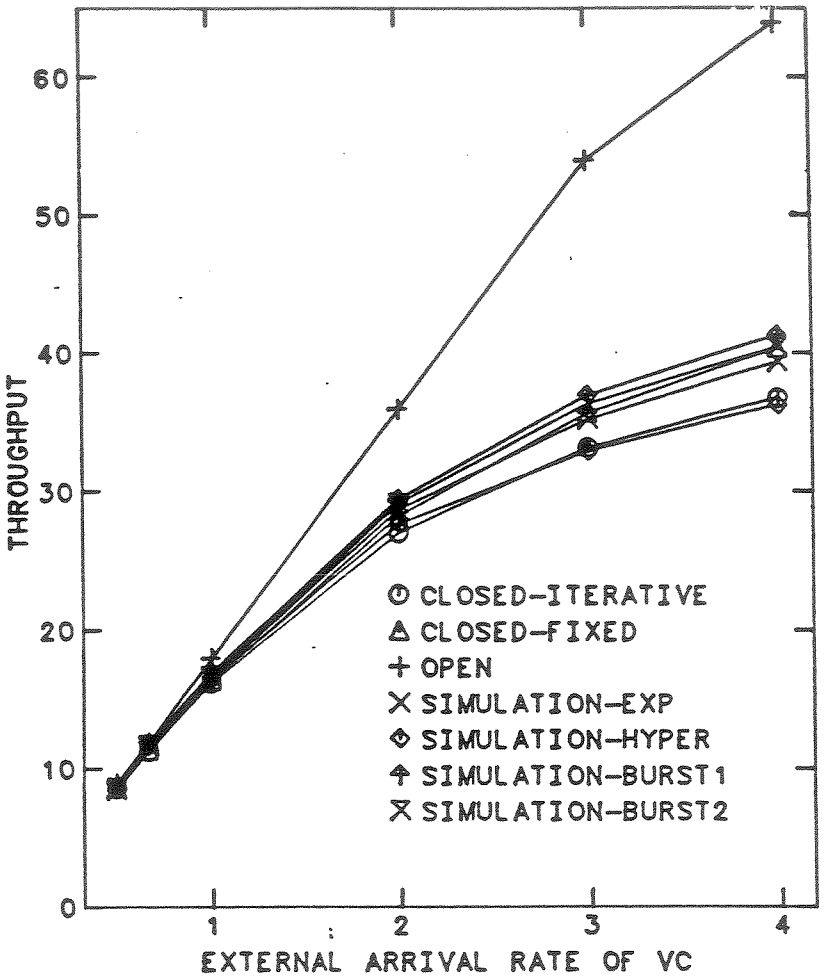

Figure 3. Network throughput versus γ for different models.

Figure 4 shows the average network delay as a function of γ. Again, the predictions of the closed network models and simulation models are very close to each other. The open network model gives predictions that are substantially higher than those of other models as γ increases. At γ=4, it predicts infinite average delay since one of the servers is saturated. Have we treated the open network model unfairly since its average delay is calculated using virtual channel throughputs that are too high as shown in Figure 3 ? To be fair, we have plotted an additional delay curve in Figure 4 labeled OPEN-IDEAL. The delay values of this curve are calculated by the open network model using virtual channel throughputs obtained by the best available closed network model (FIXED). In practice, these virtual channel throughputs may be measured in a real network. Figure 4 shows that the IDEAL open network model gives delay predictions that are still higher than all of the simulation predictions for $\gamma \geq 3$. The model is said to be ideal because it requires another model to supply it with virtual channel throughputs.

Figures 5 and 6 show the throughputs and average delays of individual virtual channels predicted by the seven models for γ=3 packets/second.

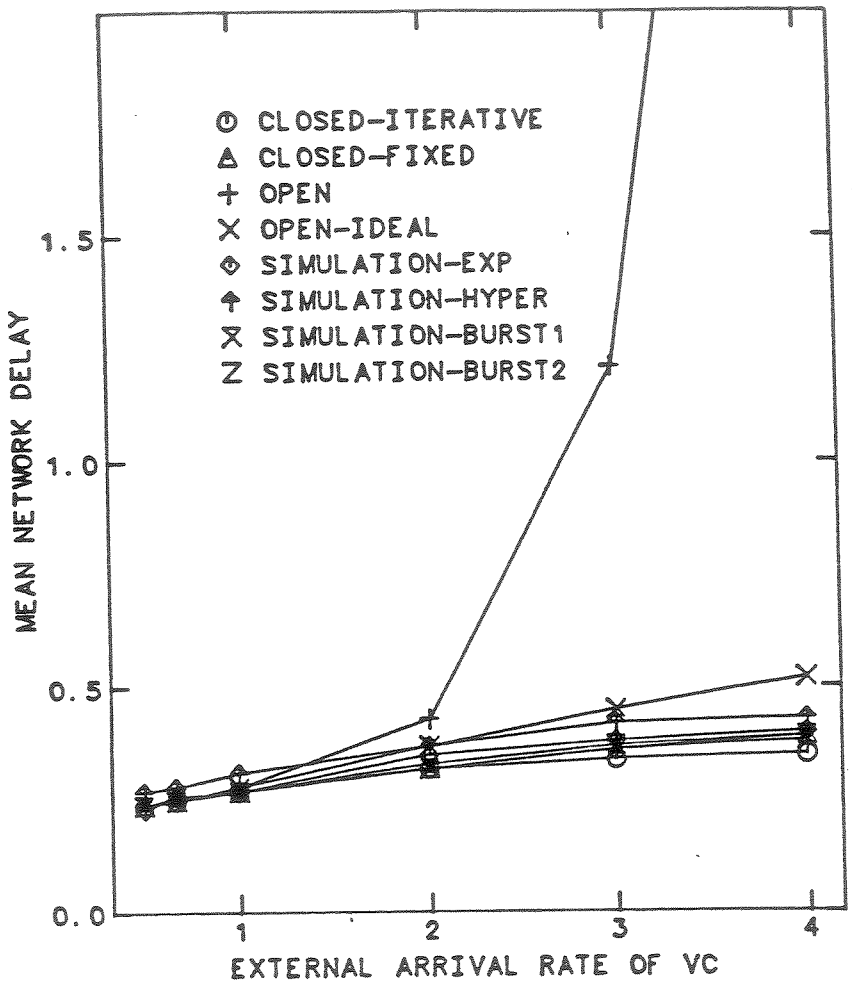

Figure 4. Mean end-to-end delays versus γ for different models.

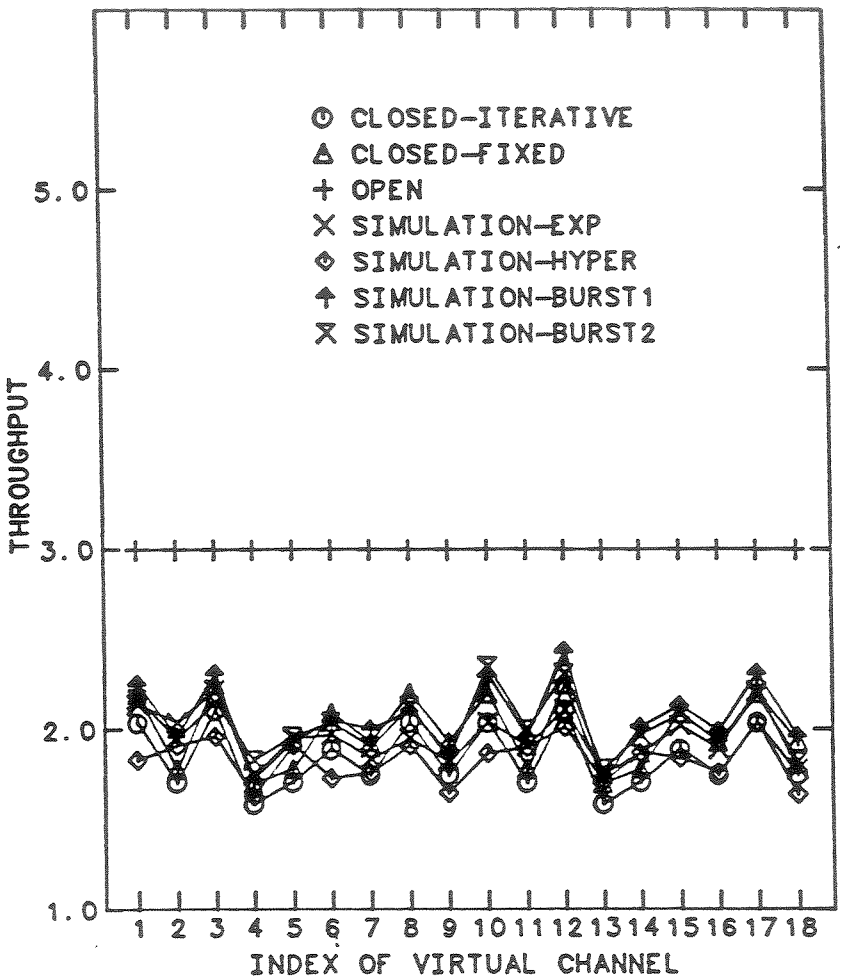

Figure 5. Virtual channel throughputs given by different models at γ=3 packets/second.

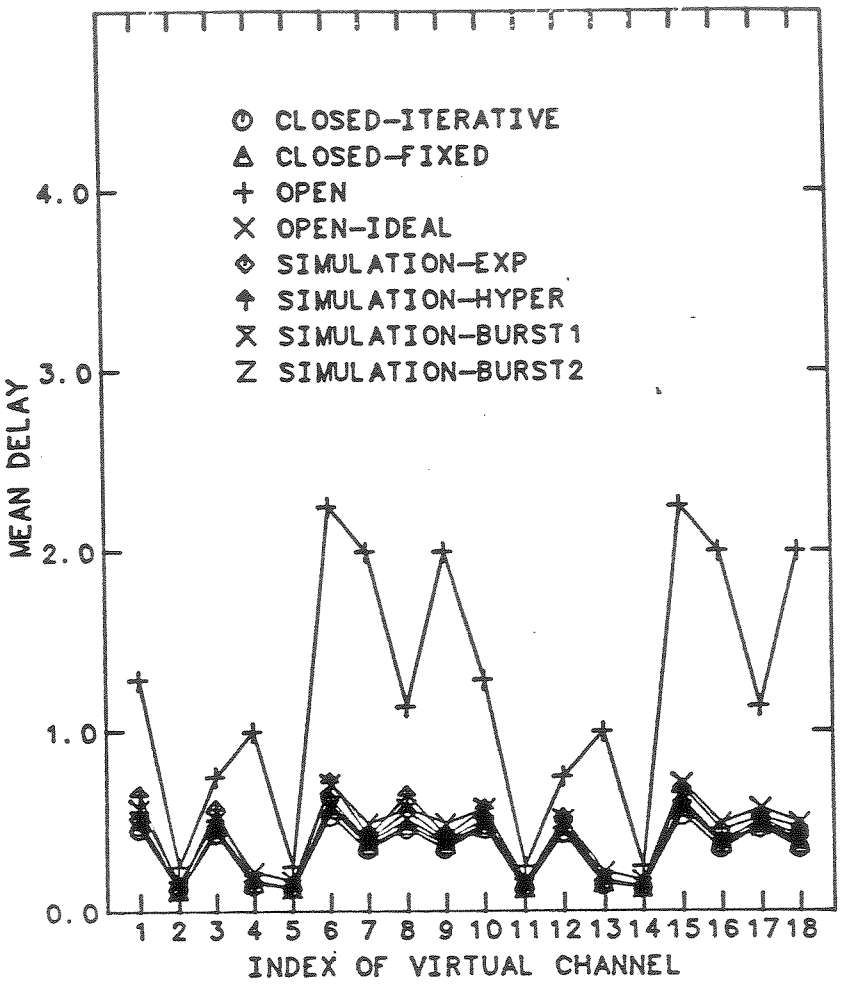

Figure 6. Mean delays of virtual channels given by different models at γ=3 packets/second.

The simulation results indicate that throughput and average delay predictions of the closed network model are quite accurate despite the independence assumption and the delay-server abstraction of end-to-end acks. Examining the numerical values of throughputs and average delays given by the four simulation models, we found that these performance measures do depend to some extent on the coefficient of variation of the packet length distribution. Not surprisingly, network performance is generally better (i.e., larger throughputs and smaller average delays) for packet lengths with a small coefficient of variation than for packet lengths with a large coefficient of variation. For the range of coefficients considered, 0.77 to 1.40, we conclude that the closed network model (FIXED) is quite robust. Its predictions for *individual* virtual channels differ from simulation results (for all four packet length distributions) by less than 10 percent in most cases and by less than 15 percent with only a few exceptions.

Using the EXP curves obtained from simulations as benchmarks, we found that, between the two closed network models, the FIXED model is more accurate than the ITERATIVE model, in addition to being computationally faster.

3. Algorithms for Analyzing Closed Queueing Networks

Consider a closed queueing network with k customer classes (or chains). We refer to $\underline{N}=(N_1,N_2,\ldots,N_K)$ as the *population vector* of the queueing network, where N_k denotes the population of the k^{th} chain. If N_k in $\underline{N}$ is greater than or equal to 1, then $\underline{N}-\underline{1}_k$ refers to the population vector $\underline{N}$ with a chain k customer removed. Let τ_{mk} denote the mean service time of chain k customers at server m, for $k=1,\ldots,K$ and $m=1,\ldots,M$. There are two kinds of servers: delay servers with no queueing and fixed-rate servers with queueing. We use $D_{mk}(\underline{n})$ to denote the approximate mean delay of chain k customers at server m, $q'_{mk}(\underline{n})$ to denote the approximate mean queue length of chain k customers at server m, and $T_k(\underline{n})$ to denote the approximate throughput of chain k customers, in a network whose population vector is $\underline{n}$.

PAM algorithms are based upon MVA formulas [17]. However, the iterations from population vector $\underline{0}$ to population vector $\underline{N}-\underline{1}_k$ in the MVA algorithm are skipped. Instead, mean queue length approximations $q'_{mh}(\underline{N}-\underline{1}_k)$ at population vector $\underline{N}-\underline{1}_k$ are obtained by distributing the population N_h of chain h, over the servers on its route, proportional to τ_{mh}. This approach was the basis of proportional upper bounds for single-chain networks in [4]. In multichain networks, this approach leads to approximations rather than upper bounds. We refer to it as the proportional approximation method.

We present below two algorithms based upon the proportional approximation method [5]. The first algorithm, PAM_BASIC, calculates throughputs of individual chains. The second algorithm, PAM_IMPROVED, calculates throughputs of all chains and the utilizations of all servers. The algorithm then checks server utilizations to see if any utilization exceeds 1. (This is possible because the chain throughputs are approximations.) The throughputs of those chains that visit a server whose utilization exceeds 1 are then scaled down.

Algorithm PAM_BASIC

Step 1: Calculate proportional approximations of mean queue lengths from

$$q'_{mh}(\underline{N}-\underline{1}_k)=\begin{cases}\dfrac{\tau_{mh}N_h}{\sum_{i=1}^{M}\tau_{ih}} & \text{if } h\neq k\\[2ex] \dfrac{\tau_{mh}(N_h-1)}{\sum_{i=1}^{M}\tau_{ih}} & \text{if } h=k.\end{cases}$$

for $m = 1,2,\ldots,M$, $k = 1,2,\ldots,K$, and $h = 1,2,\ldots,K$.

Step 2: Calculate approximate mean delay of chain k at server m and approximate throughput of chain k from the following MVA formulas:

$$D_{mk}(\underline{N})=\begin{cases}\tau_{mk}\left(1+\sum_{h=1}^{K}q'_{mh}(\underline{N}-\underline{1}_k)\right) & \text{if } m \text{ is a fixed-rate server}\\[2ex] \tau_{mk} & \text{if } m \text{ is a delay server}\end{cases}$$

for $m = 1,2,\ldots,M$ and $k = 1,2,\ldots,K$, and

$$T_k(\underline{N}) = \frac{N_k}{\sum_{m=1}^{M} D_{mk}(\underline{N})} \qquad \text{for } k = 1,2,\ldots,K.$$

Total throughput of the network, if needed, is equal to the summation of the chain throughputs. (Note that Step 1 can be implemented with MK multiplications and MK divisions.)

Algorithm PAM_IMPROVED

The first two steps of this algorithm are the same as those of Algorithm PAM_BASIC.

Step 3: Calculate server utilizations from the following formula:

$$U_m(\underline{N}) = \sum_{k=1}^{K} \tau_{mk} T_k(\underline{N})$$

for $m = 1,2,\ldots,M$, where $U_m(\underline{N})$ is the utilization of server m at population vector $\underline{N}$.

Step 4: Find the largest utilization S_k among the fixed-rate servers visited by chain k,

$$S_k = \max_{\substack{m \text{ in} \\ \text{chain } k \text{ and} \\ m \text{ is a fixed-rate server}}} U_m(\underline{N})$$

for $k=1,2,\ldots,K$.

Step 5: (Scale down throughputs of individual chains if necessary.)
If $S_k > 1$ then $T_k(\underline{N}) := T_k(\underline{N}) / S_k$.

Step 6: Calculate total throughput, and recalculate server utilizations if the throughput of any chain has been scaled down,

$$T(\underline{N}) = \sum_{k=1}^{K} T_k(\underline{N})$$

and

$$U_m(\underline{N}) = \sum_{k=1}^{K} \tau_{mk} T_k(\underline{N}), \text{ for } m = 1,2,\ldots,M.$$

Because of the additional steps, the approximate throughputs calculated by PAM_IMPROVED are, in general, more accurate than those calculated by PAM_BASIC. Another algorithm, PAM_TWO, is presented in [5]. It is more accurate than PAM_IMPROVED but it has a computational time requirement of $O(MK^2)$ instead of $O(MK)$ for the two algorithms presented above; the space requirement is $O(MK)$ for all three PAM algorithms.

In the above algorithms, we have assumed that the virtual channels in a communication network have loop-free fixed routes. If the routing behavior of a closed chain is specified by a first-order Markov chain, then τ_{mk} in the above algorithms should be interpreted as the load, or traffic intensity, of chain k at server m rather than as the mean service time [17].

We have conducted two sets of experiments to examine the accuracy of PAM algorithms [5]. In the first, 100 networks with characteristics of models of communication networks were generated randomly; these networks have fixed-rate servers only.[7] The accuracy of PAM_BASIC and PAM_IMPROVED was studied by comparing their approximate results with exact results given by the tree convolution algorithm (TCA). In the second set of experiments, 500 networks with characteristics of models of computer systems were generated and the accuracy of PAM_IMPROVED and PAM_TWO was studied by comparisons with exact results given by the MVA algorithm and approximate results given by three iterative approximate solution methods. These networks have delay servers and fixed-rate servers.

We shall present some statistical data from the 100-network experiment. Before doing so, we consider a network example with 30 fixed-rate servers and 20 chains and compare the approximate throughputs computed by PAM_BASIC and PAM_IMPROVED with the exact throughputs computed by TCA. (The network example is completely specified in [5].) The network is fairly heavily utilized. The maximum server utilization, computed by TCA, is 0.990. The average server utilization, among those with nonzero traffic, is 0.398. The accuracy of PAM_BASIC is illustrated in Figure 7. The accuracy of PAM_IMPROVED is illustrated in Figure 8. Chain throughputs computed by PAM_IMPROVED are very accurate for this example. The maximum percentage error is 6% for chain 10. Throughputs computed by PAM_BASIC and PAM_IMPROVED are the same for those chains that do not visit a highly utilized server. For those chains that visit a highly utilized server (chains 3, 7, 14, 18), throughputs computed by PAM_BASIC have much larger percentage errors than PAM_IMPROVED. The proportional approximation tends to underestimate mean queue lengths at bottleneck servers. Thus PAM_BASIC overestimates the throughputs of chains that visit such bottlenecks. While PAM_IMPROVED checks server utilizations and compensates for these errors, PAM_BASIC does not.

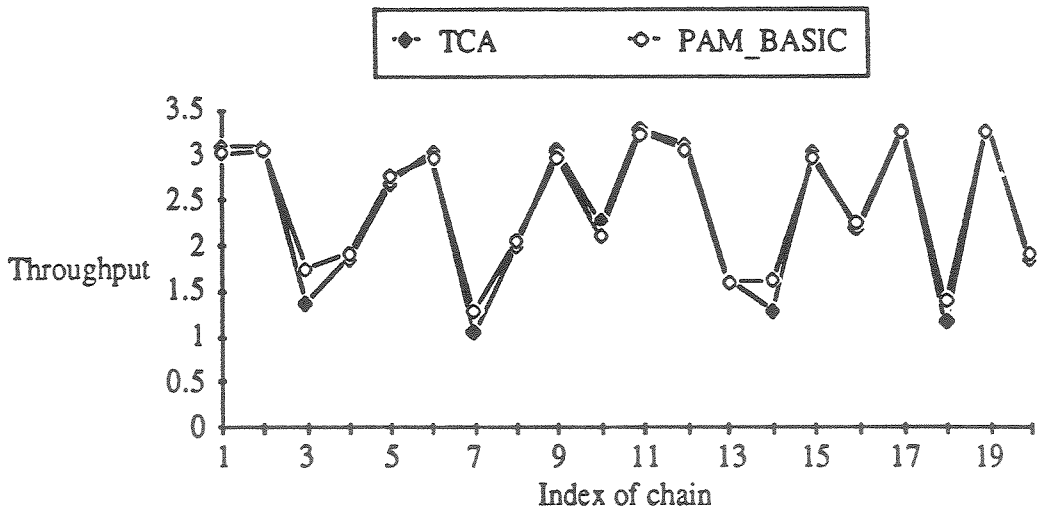

Figure 7. Throughputs of chains calculated by TCA and PAM_BASIC.

[7] In other words, mean end-to-end ack delays are assumed to be zero.

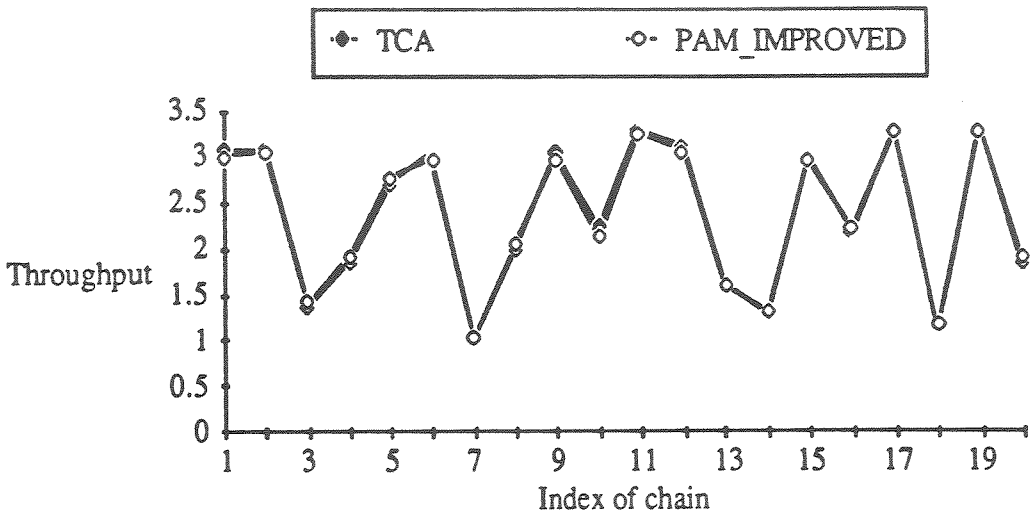

Figure 8. Throughputs of chains calculated by TCA and PAM_IMPROVED.

We next present some results of the 100-network experiment. The 100 networks were generated randomly (see [3, 5] for details). Statistics on the parameters of the 100 networks are shown in Table 2. In Table 2, #packets denotes the summation of all chain populations. In Figures 9-11, we show distributions of percentage errors of total network throughput, chain throughputs, server utilizations calculated by PAM_IMPROVED compared to exact solutions of TCA. A summary of the error statistics is shown in Table 3. Note that the maximum percentage error of total network throughput calculated by PAM_IMPROVED is 3.55% for the 100 networks. The average percentage error of chain throughput is 2.67%. Given its modest computational time requirement, we consider PAM_IMPROVED to be highly accurate.

	Maximum	Minimum	Average
1. # nodes	25	7	15
2. # links	37	9	20
3. # queues	74	18	40
4. # chains	43	8	23
5. # packets	107	19	57
6. Ave. queue util.	0.546	0.291	0.419

Table 2. Statistics of 100 test networks.

	Statistics of percentage errors				
	#samples	Minimum	Maximum	Mean	Variance
Total throughput	100	0.009	3.554	0.824	0.002
Throughput of chain	2255	0.000	20.503	2.667	7.167
Utilization of queue	3104	0.000	14.769	2.064	4.263

Table 3. Statistics of approximation errors of total throughput, chain throughputs, and server utilizations calculated by PAM_IMPROVED for the 100 networks.

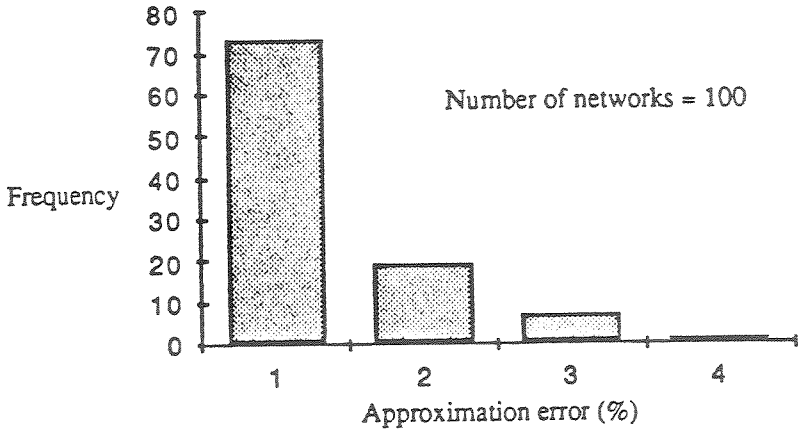

Figure 9. Distribution of approximation errors of total throughputs calculated by PAM_IMPROVED for the 100 networks.

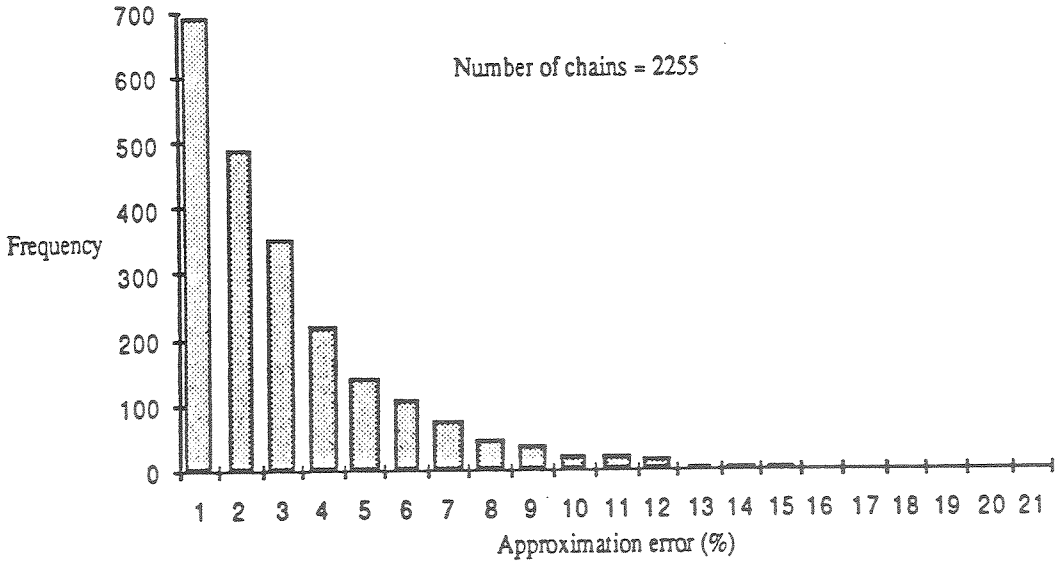

Figure 10. Distribution of approximation errors of throughputs of individual chains calculated by PAM_IMPROVED for the 100 networks.

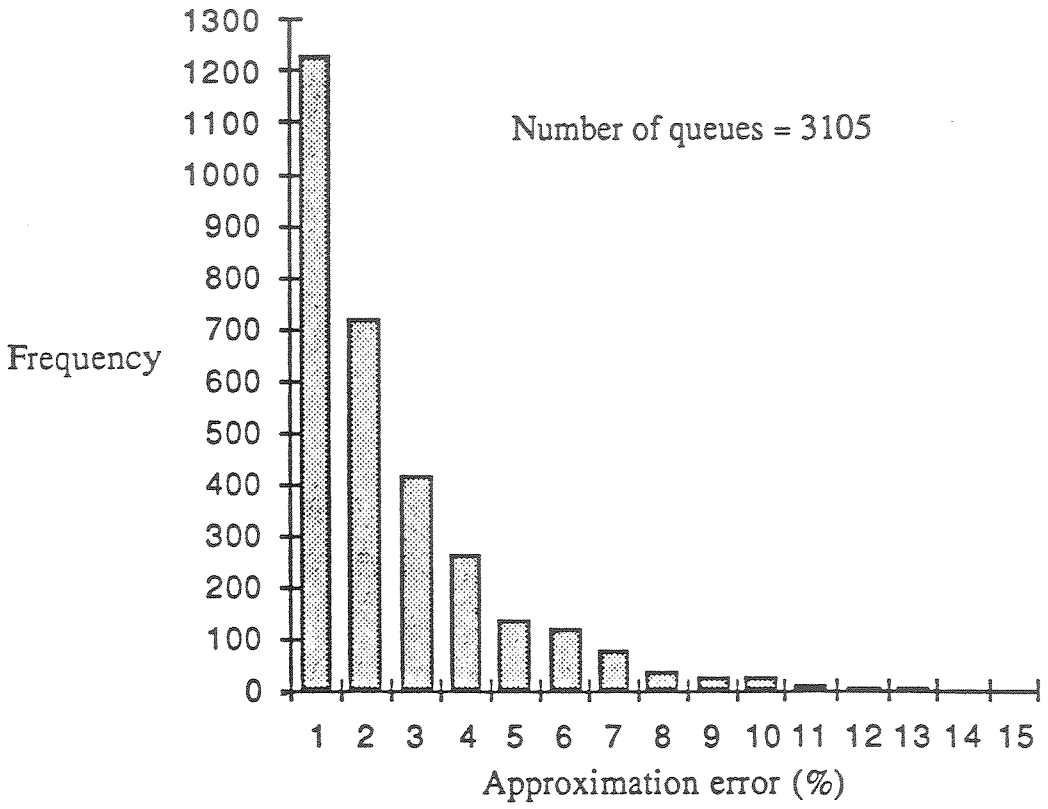

Figure 11. Distribution of approximation errors of server utilizations calculated by PAM_IMPROVED for the 100 networks.

We also studied the correlation between the percentage error of the throughput of a chain calculated by PAM_BASIC and the maximum utilization among servers visited by the chain. We selected one chain from each of the 100 networks in the above experiment and calculated its throughput with PAM_BASIC. The percentage error in throughput and the corresponding maximum server utilization are plotted in Figure 12. Note that throughput errors of PAM_BASIC and maximum server utilizations are highly correlated. Percentage errors are large only for chains that visit servers with utilizations very close to 1. This is a very useful observation! It suggests that PAM_BASIC, being the fastest of the three PAM algorithms, can be used in many network design procedures. The general objective of optimal routing, for example, is to route traffic in such a way that no bottleneck exists in the network. While PAM_BASIC has relatively large errors for those chains that visit bottlenecks during an intermediate step of an optimal routing procedure, it can still be effectively used to select good routes which generally do not include bottlenecks. And after bottlenecks have been eliminated from a network, PAM_BASIC provides accurate throughputs.

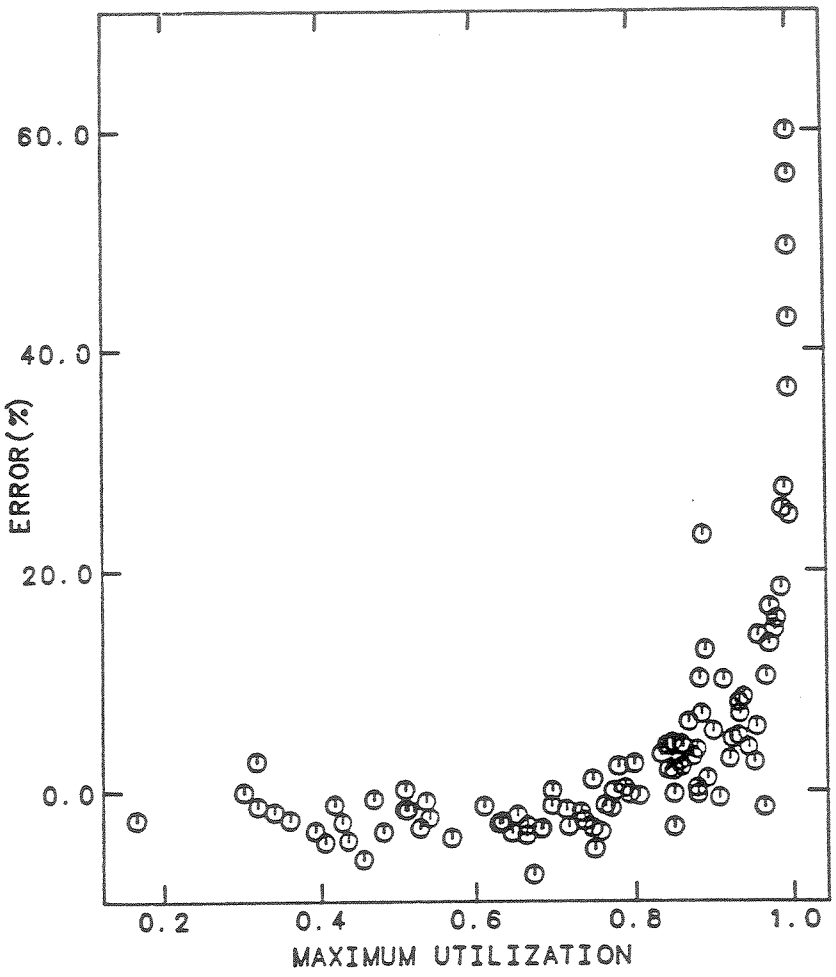

Figure 12. Correlation between percentage error of chain throughput calculated by PAM_BASIC and the maximum utilization of the servers visited by the chain.

4. Optimal Routing

Optimal routing plays a central role in network design problems. In the process of designing a network, numerous modifications are usually made before a satisfactory network is obtained. After each modification, optimal routing has to be performed to adapt to any change in network topology or traffic. An optimal routing algorithm to be used in network design should be as fast as possible. Also the algorithm that is used for evaluating network performance within the optimal routing algorithm has to be very fast.

We present an algorithm to solve the following problem for the closed network model: Given a network with a set of flow-controlled virtual channels, find the optimal route for a virtual channel to be added to the network to maximize the total throughput of the network including the new virtual channel. This algorithm can be used to add virtual channels to a network one at a time until all virtual channels have been included in the network. If further improvement in the network throughput is desired, then some virtual channels can be selected for rerouting.

Most optimal routing algorithms are based upon finding the shortest path between a pair of source

and destination; they differ primarily in their link metrics. When a closed network model is used for network design, however, a link metric is not easily defined. The two basic steps of our optimal routing algorithm are the following:

(1) Find a set of routes with the smallest end-to-end mean delays for the virtual channel being added to the network.

(2) Use PAM_BASIC to find a route, from the set in step (1), that gives the highest network throughput.

The mean delays in step (1) are calculated using the proportional approximation. However, the mean queue lengths of the virtual channel for which a route is to be found, say virtual channel k, are not included since virtual channel k does not yet have a route. This is equivalent to treating virtual channel k as having a window size of one so that $q'_{mk}(\underline{N}-\underline{1}_k)=0$ for all m. This is the key assumption that makes possible the use of mean delay as a link metric and transforms step (1) into a shortest path problem that can be solved very efficiently.

A route selected in step (1) with the smallest end-to-end mean delay for the new virtual channel does not necessarily maximize the total throughput of the network in a closed queueing network. Thus in step (1) we employ one metric to select a set of good candidates and in step (2) employ a better metric to pick the best route from the set of candidates. Note that the metric in step (2) cannot be evaluated efficiently because each possible route has to be evaluated separately by solving a closed network model. By using different performance metrics at different stages of our search heuristic we reduce computational time very substantially.

The effectiveness of our algorithm to find an optimal or suboptimal route depends upon the accuracy of the proportional approximation as well as the following empirical observation: The route that maximizes the individual throughput of the new virtual channel coincides frequently with the route that maximizes the total network throughput (this is not true in general). Note that the route that maximizes the throughput of a closed chain in a queueing network also minimizes its mean delay according to Little's formula. In fact, only an approximation of the new virtual channel's mean delay is evaluated in step (1). Nevertheless, statistics from our 100-network experiment show the following heuristic optimal routing algorithm to be very effective.

Algorithm HRA1 *(To find an optimal route for chain k)*

Step 1: Calculate approximate mean queue length $q_{mh}(\underline{N}^k)$ by the following formula:

$$q_{mh}(\underline{N}^k) = \frac{\tau_{mh}}{\sum_{i=1}^{M} \tau_{ih}} N_h \qquad \text{for } m=1,2,\ldots,M, \qquad h=1,2,\ldots,K \text{ and } h\neq k$$

where $\underline{N}^k$ is the population vector of the network with chain k removed.

Step 2: Compute the approximate delay using the MVA formula,

$$D_{mk}(\underline{N})=\tau_{mk}\left(1+\sum_{\substack{h=1\\h\neq k}}^{K} q_{mh}(\underline{N}^k)\right) \qquad \text{for } m=1,2,\ldots,M$$

Step 3: Create a routing tree for chain k. The root node of the tree is the source and the leaf nodes are the destination of chain k. Select a set R of shortest delay routes from the routing tree

using the approximate delays calculated in Step 2 as metric.

Step 4: Use algorithm PAM_BASIC to calculate the total network throughput for each route in R. Select the route in R that yields the largest network throughput.

With a slight modification, Algorithm HRA1 can be used to find a route to maximize the individual throughput of chain *k*. The algorithm, to be called HRA2, is given below.

Algorithm HRA2

The first three steps of the algorithm are the same as those of Algorithm HRA1.

Step 4: Use algorithm PAM_BASIC to calculate the throughput of chain *k* for each route in R. Select a route from R that yields the largest throughput of chain *k*.

The key to a successful application of these algorithms is to find a small set R which includes the optimal route or a near-optimal route. The following three choices of R were used in our experimental study.

I. Only the shortest delay route among all possible routes is selected.

II. Up to three shortest delay routes are chosen.

III. Up to ten shortest delay routes are chosen.

The *Branch-and-Bound* paradigm was used to find the |R| shortest routes for a chain.

Experimental results

The algorithms were tested on the 100 randomly generated networks used for testing the accuracy of PAM. For each network, the source node and destination node of a new chain, for which an optimal route is to be found, were randomly generated. (If the source node and the destination node are adjacent to each other then another node pair is generated.) The parameter values of the new chain were selected in the same way as those of existing chains in the network by the network generator as described in [3, 5]. To be practical, we introduce the following maximum-hop constraint: Each chain can visit up to 9 nodes, or 80 percent of the nodes in the network (including the source and destination nodes), whichever is smaller. Each heuristic routing algorithm was applied to the test networks to find the best routes for the new chains. Exact throughputs for all feasible routes satisfying the maximum-hop constraint were also computed by the tree convolution algorithm as benchmarks. Statistics of results from our experiment using Algorithm HRA1 with different choices of R are given in Table 4. In this table, $\Delta = |T^o - T^h|$, where T^o is the largest exact total throughput that can be achieved among all feasible routes satisfying the maximum-hop constraint and T^h is the exact total throughput achieved by the route chosen by Algorithm HRA1. Both T^o and T^h were calculated using TCA.

	Statistics of deviations from optimal total throughput					
|R|	1		3		10	
	Δ	$\frac{\Delta \times 100\%}{T^o}$	Δ	$\frac{\Delta \times 100\%}{T^o}$	Δ	$\frac{\Delta \times 100\%}{T^o}$
Minimum	0.0	0.0	0.0	0.0	0.0	0.0
Maximum	0.839	1.072	0.838	1.071	0.412	0.547
Average	0.056	0.062	0.026	0.032	0.018	0.022
Std. dev.	0.161	0.185	0.106	0.134	0.068	0.084

Table 4. Statistics of deviations from optimal total throughput for Algorithm HRA1.

A distribution of the percentage differences (with respect to T^o) between T^o and T^h for the routes selected by Algorithm HRA1 is shown in Figure 13. In this experiment, Algorithm HRA1 examined only the shortest delay route for the new chain (|R|=1). Table 4 and Figure 13 show that the heuristic algorithm HRA1 found the optimal route most of the time! Even when it failed to find the optimal route, the exact total throughput of the network for the route chosen by HRA1 is very close to the optimal total throughput. Table 4 also shows that increasing the number of routes examined by HRA1 does improve its performance, though not by very much.

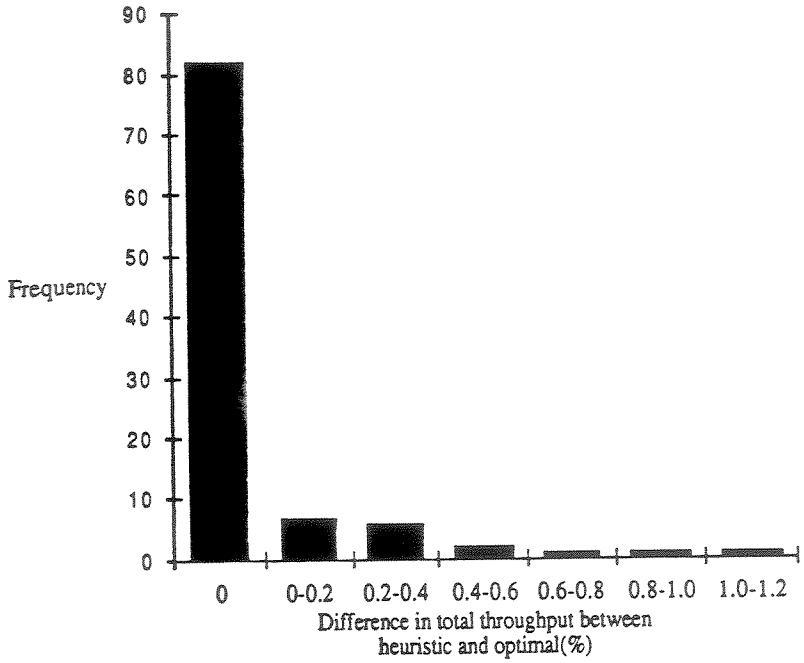

Figure 13. Distribution of percentage difference between the optimal total throughput and the total throughput achieved by the heuristic algorithm.

We also applied Algorithm HRA2 to the 100 networks used to test Algorithm HRA1. Again, the value of |R| was one. In Table 5, Δ is $|T_k^o - T_k^h|$, where T_k^o is the largest throughput of chain k (calculated by TCA) among all feasible routes satisfying the maximum-hop constraint, and T_k^h is the exact throughput of chain k (calculated by TCA) whose route was chosen by Algorithm HRA2.

	Maximum	Minimum	Average	Std. dev.
Δ	0.202	0.0	0.003	0.022
$\frac{\Delta \times 100\%}{T_k^o}$	13.732	0.0	0.214	1.442

Table 5. Statistics of results from the heuristic routing algorithm HRA2.

Of the 100 networks tested, Algorithm HRA2 actually found a route maximizing the throughput of chain k in 95 networks. Among these 95 networks, 80 of them also have the largest total throughputs that can be achieved among all feasible routes of chain k (subject to the maximum-hop constraint).

The network used in the next example has 21 nodes, 30 links (60 queues), and 30 virtual channels. (See Tables 6 and 7.) The optimal route for a new virtual channel from node 20 to node 3 is to be selected. We found that both heuristic algorithms gave the same ten shortest delay routes for the new

chain (because the first three steps of the algorithms are the same). Exact throughputs and approximate throughputs of the new chain for these ten routes, calculated by TCA and PAM_BASIC respectively, are shown in Figure 14. In this figure, the order of the best seven routes is the same whether the routes are ordered by exact chain throughput or approximate chain throughput. The exact total throughput and approximate total throughput of the network resulting from assigning each of the ten routes to the new chain are shown in Figure 15. In this figure, the two curves have the same trend for all ten routes.

Link	Capacity (bits/second)
(1,19)	2400
(14,6)	2400
(17,12)	9600
(20,12)	2400
(1,5)	4800
(7,10)	2400
(14,9)	9600
(17,20)	9600
(18,15)	4800
(1,11)	9600
(2,21)	2400
(3,18)	2400
(4,13)	2400
(5,16)	2400
(7,13)	4800
(8,9)	2400
(10,4)	2400
(14,8)	2400
(17,11)	4800
(19,5)	4800
(20,6)	9600
(1,16)	9600
(2,10)	9600
(3,14)	2400
(7,18)	9600
(12,19)	2400
(15,21)	4800
(17,8)	2400
(20,19)	2400
(21,16)	2400

Table 6. Links and their capacities in network example.

Chain	Population	Mean service time at source server	Route (in node sequence)
1	2	0.1	11 1 16
2	3	0.3	17 20 6 14
3	3	0.3	17 20 19 5
4	2	0.2	3 14 8
5	2	0.2	14 6 20 19
6	2	0.3	16 1
7	2	0.3	9 14 3
8	2	0.3	8 17 12 19
9	2	0.3	4 10 7
10	3	0.2	7 10
11	3	0.3	13 7 10 2
12	2	0.2	9 14 6 20 17 12
13	2	0.3	18 3 14 8
14	3	0.3	17 11 1 5
15	2	0.3	17 12 19 5
16	2	0.2	3 18 7 10
17	3	0.2	21 15 18 7 13
18	3	0.1	3 18 7 13 4
19	2	0.1	20 17 11 1 16
20	2	0.2	13 4
21	3	0.3	2 21 15
22	2	0.3	4 10 2 21
23	2	0.2	13 4
24	4	0.3	17 8 9
25	2	0.2	18 15 21
26	4	0.1	15 21 2 10 4
27	2	0.1	4 13 7 18 15
28	2	0.1	18 3
29	2	0.3	9 8 17 12
30	2	0.3	18 15 21 2

Table 7. Chain populations, mean service times at source servers, and routes of chains in network example.

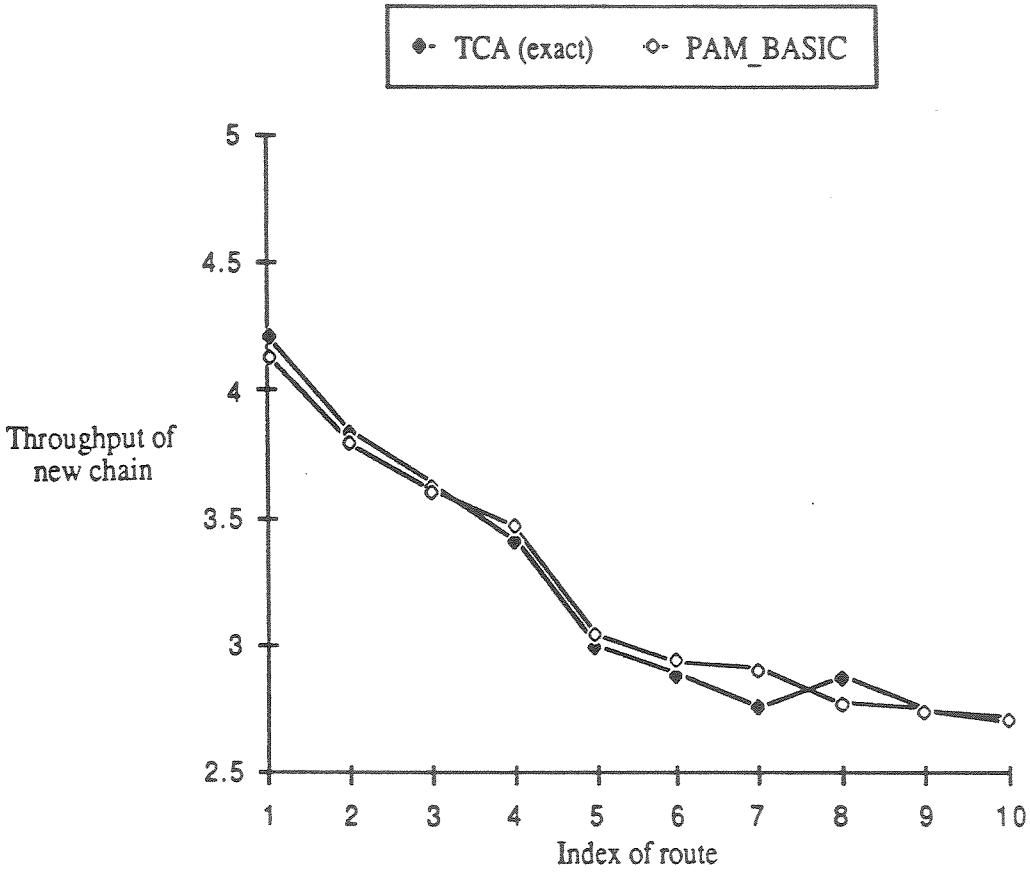

Figure 14. Exact and approximate throughputs of a chain using different routes.

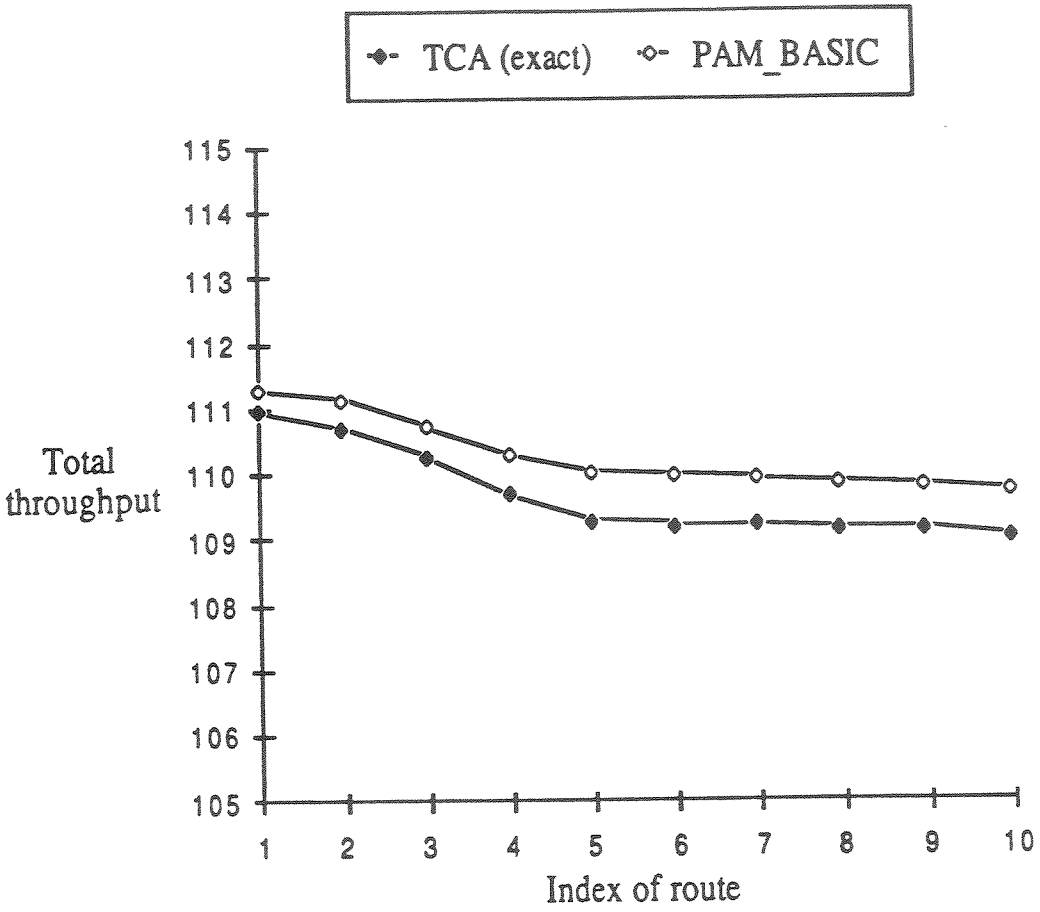

Figure 15. Exact and approximate total network throughputs for different routes of a particular chain.

What we have demonstrated using the above example is that approximate throughputs calculated by PAM_BASIC are adequate as a metric for comparing routes. This is because bottleneck queues are usually not visited by an optimal route (or a near-optimal route). We have shown in Section 3 that the approximate throughputs of individual chains calculated by PAM_BASIC are very accurate if they do not visit bottleneck queues. Note also, from Figures 14 and 15, that the first seven routes that yield the largest throughputs for the new chain also yield the largest total throughputs for the network. This gives further evidence that it is effective to search for a route maximizing total throughput from among a set of routes that maximize the throughput of the new chain.

5. Conclusions

We have described a closed queueing network model for the analysis and design of flow-controlled communication networks and demonstrated its accuracy and robustness with experimental results. It gives a network designer more information about the performance of a flow-controlled communication network than an open queueing network model which cannot be used to predict throughputs of flow-controlled virtual channels.

In applying the closed network model to network analysis and design, some engineering judgment is needed to determine the "effective window sizes" of virtual channels. There are two sources of uncertainty. First, the implementations of some sliding window protocols are more complicated than the one described in Section 2, e.g., SNA pacing [1, 18]. Second, the mean delays of acks travelling from destinations to sources need to be estimated. Suppose we let the end-to-end ack delay of each virtual channel be zero in the closed network model. In order for a virtual channel in the communication network being modeled to achieve the throughput predicted by the closed network model, its window size should be larger than the chain population N_k in the model by $\alpha_k T_k$, where T_k is the chain throughput predicted by the model and α_k is virtual channel's average end-to-end delay of acks. By Little's formula [7], $\alpha_k T_k$ is the average number of permits that are en route from the destination to the source. Alternatively, one might estimate this number directly instead of estimating α_k first and then rounding off $\alpha_k T_k$ to an integer.

We have presented two algorithms based upon the *proportional approximation method (PAM)* for evaluating performance measures of closed queueing networks. These PAM algorithms have computational time and space requirements of $O(KM)$, where K denotes the number of virtual channels and M the number of queues; they can be used to solve models of large communication networks. Statistical studies comparing the predictions of PAM algorithms with exact solutions calculated by the tree convolution algorithms show that the PAM algorithms are very accurate.

We have also illustrated the use of the closed queueing network model and the PAM algorithms for network design by solving the following optimal routing problem: Find a route for a new virtual channel to be added to a network with existing flow-controlled virtual channels. The total network throughput is to be maximized. (Note that this problem cannot be solved using an open queueing network model.) A fast heuristic solution algorithm based upon PAM is presented. Routes obtained by the heuristic algorithm were compared with optimal routes obtained by an exhaustive search using exact results calculated by the tree convolution algorithm. The heuristic algorithm was very effective in finding an optimal route or a near-optimal route almost all the time.

References

[1] F. D. George and G. E. Young, "SNA Flow Control: Architecture and Implementation," *IBM Systems Journal*, Vol. 21, No. 2, 1982, pp. 179-210.

[2] M. Gerla and L. Kleinrock, "Flow Control: A Comparative Survey," *IEEE Trans. on Commun.*, Vol. COM-28, No. 4, April 1980, pp. 553-574.

[3] C.-T. Hsieh, *Models and Algorithms for the Design of Store-and-Forward Communication Networks*, Ph.D. Dissertation, Department of Computer Sciences, University of Texas at Austin, May 1987.

[4] C.-T. Hsieh and S. S. Lam, "Two Classes of Performance Bounds for Closed Queueing Networks," *Performance Evaluation*, Vol. 7, No. 1, 1987, pp. 3-30.

[5] C.-T. Hsieh and S. S. Lam, "PAM — A Noniterative Approximate Solution Method for Closed Multichain Queueing Networks," Technical Report TR-87-28, Department of Computer Sciences, University of Texas at Austin, July 1987.

[6] L. Kleinrock, *Communication Nets: Stochastic Message Flow and Delay*, McGraw-Hill, New York, 1964.

[7] L. Kleinrock, *Queueing Systems, Vol. 2: Computer Applications*, John Wiley, New York, 1976.

[8] H. Kobayashi and M. Gerla, "Optimal Routing in Closed Queueing Networks," *ACM Transactions on Computer Systems*, Vol. 1, No. 4, Nov. 1983, pp. 294-310.

[9] S. S. Lam and C.-T. Hsieh, "Models and Algorithms for the Design of Store-and-Forward Communication Networks, *Conf. Record International Conf. on Communications*, Chicago, June 1985, pp. 43.1.1-43.1.6.

[10] S. S. Lam and Y. L. Lien, "Modeling and Analysis of Flow Controlled Packet Switching Networks," *Proc. 7th Data Communications Symposium*, Mexico City, October 1981.

[11] S. S. Lam and Y. L. Lien, "Congestion Control of Packet Communication Networks by Input Buffer Limits—A Simulation Study," *IEEE Trans. on Computers*, Vol. C-30, No. 10, October 1981, pp. 733-742.

[12] S. S. Lam and Y. L. Lien, "A Tree Convolution Algorithm for the Solution of Queueing Networks," *Comm. ACM*, Vol. 26, No. 3, March 1983, pp. 203-215.

[13] S. S. Lam and J. W. Wong, "Queueing Network Models of Packet Switching Networks, Part 2: Networks with Population Size Constraints," *Performance Evaluation*, Vol. 2, No. 3, October 1982, pp. 161-180.

[14] Y. L. Lien, *Modeling and Analysis of Flow Controlled Computer Communication Networks*, Ph.D. Dissertation, Department of Computer Sciences, University of Texas at Austin, 1981.

[15] M. Reiser, "A Queueing Network Analysis of Computer Communication Networks with Window Flow Control," *IEEE Trans. on Communication*, Vol. COM-27, pp. 1199-1209, 1979.

[16] M. Reiser and H. Kobayashi, "Queueing Networks with Multiple Closed Chains: Theory and Computational Algorithms," *IBM J. Res. Develop.*, Vol. 21, pp. 283-294, 1975.

[17] M. Reiser and S. Lavenberg, "Mean Value Analysis of Closed Multichain Queueing Networks," *Journal of ACM*, Vol. 27, No. 2, pp. 313-322, April 1980.

[18] M. Schwartz, *Telecommunication Networks: Protocols, Modeling and Analysis*, Addison-Wesley, Reading, Mass., 1987.