MODELING AND ANALYSIS OF FLOW CONTROLLED PACKET SWITCHING NETWORKS*

Simon S. Lam and Y. Luke Lien

Department of Computer Sciences
University of Texas at Austin
Austin, Texas 78712

Abstract

Packet switching networks with flow controlled virtual channels are modeled by closed multi-chain queueing networks. The tree convolution algorithm for an exact analysis of such models is discussed. The algorithm is very efficient when routing chains have sparseness and locality properties that are typical of communication network models. The accuracy of an approximate model of equivalent open chains was investigated. An optimal routing criterion for adding a virtual channel (with a window size of one) to an existing network is explored.

1. INTRODUCTION

The early store-and-forward packet switching networks are mostly datagram networks. In these networks, each packet carries its own source-destination addresses. It is treated as an independent entity with regard to its acceptance into the network and subsequent movement through the network. The current generation of packet switching networks, however, are mostly virtual channel networks [ROBE 78]. In these networks, packets are associated with logical source-destination connections called virtual (or logical) channels. Each packet is identified by its virtual channel ID. Among other attributes, virtual channels are individually end-to-end flow controlled. Examples of end-to-end flow controls are SNA pacing [IBM 75], RFNM in ARPANET [OPDE 74] and various window mechanisms [POUZ 73, CERF 74]. All of them work by limiting the number of packets that a virtual channel can have in transit within the network. (This number will be referred to as the virtual channel window size.) An important function of end-to-end flow controls is the synchronization of the data source input rate to the data sink acceptance rate. They also provide, to some extent, a form of congestion control capability for the network.

We will not dwell upon the details and relative merits of datagram and virtual channel networks [ROBE 78]. Our main interest here is on models for network performance analysis and design. Datagram networks are modeled as an open queueing network given the independence assumption of Kleinrock [KLEI 64]. Such a model forms the basis of extensive studies on the design and analysis of store-and-forward packet switching networks [KLEI 64, KLEI 76, GERL 77, SCHW 77, GALL 77].

* The work was supported by National Science Foundation Grant No. ECS78-01803.

Packet switching networks with flow-controlled virtual channels, on the other hand, are modeled as a queueing network with closed routing chains; each closed chain represents a flow-controlled virtual channel and the chain population size is equal to the virtual channel window size. Although in practice, virtual channel networks are becoming the dominant form of networks, available tools for network analysis and design are still mainly based upon the open queueing network model suitable for datagram networks. The reason for this situation is simple. Open queueing networks have a closed-form solution, while only an algorithmic solution is available for closed multi-chain queueing networks. Furthermore, the time and space complexity of the existing computational algorithms [REIS 75, REIS 80] grow exponentially with the number of chains (virtual channels) thus rendering closed multi-chain network models practically unsolvable except with the use of approximate solution techniques [PENN 75, REIS 79].

Summary of this paper

In Section 2, we shall next describe our queueing network model for packet switching networks with flow controlled virtual channels. The time and space complexity of existing computational algorithms and prior work on approximate solutions are briefly reviewed. A computational algorithm recently developed by us [LAM 81a], called the tree convolution algorithm, is next described in Section 3. It is a general algorithm for the solution of product-form queueing networks. It is very efficient for networks with many queues and many "sparse" routing chains, such as models of communication networks.

Optimization procedures in network design require a relatively fast solution technique for evaluating perturbations in a traffic pattern. In Section 4, the tree convolution algorithm is used to examine the accuracy of the approximation of closed chains by open chains. Conditions for a good approximation are explored. Lastly, in Section 5, we consider the (flow deviation) routing problem of adding a flow-controlled virtual channel with window size equal to 1 to an existing network.

2. THE MODEL AND SOLUTION TECHNIQUES

The model to be considered follows from discussions in [REIS 79, LAM 81b]. The assumptions are summarized below:

CH1694-9/81/0000/0098$00.75 © 1981 IEEE

1. Processing delays within packet switching nodes are ignored since they are typically much smaller than channel delays. This assumption can be easily relaxed as done in [KLEI 76]. We also assume that packet switching nodes have sufficient buffers so that blocking due to buffer overflow has negligible probability. (The problem of buffer overflow was considered in [LAM 76].)

2. The nodes are connected by uni-directional communication channels. Each channel is modeled as a FIFO queue with exponentially distributed service times. The independence assumption of packet transmission time is needed [KLEI 64].

3. There are K uni-directional virtual channels between pairs of nodes. Each virtual channel has a source and a sink both of which are also modeled as FIFO queues with exponentially distributed service times. Packets in the same virtual channel follow a fixed route which may be chosen probabilistically from a finite set of routes between source and sink.

4. The delay for the return of an end-to-end (ETE) acknowledgement from the sink to the source indicating receipt of a packet is modeled by an independent random variable, the distribution of which may be different for different virtual channels. ETE acknowledgements are typically either piggy-backed in data packets or, if sent separately, very short. Thus, they consume relatively small amounts of buffer and channel resources in the network, which may be accounted for separately. Reiser considered models which are more general than the above; his models, however, can only be handled by an approximate analysis [REIS 79].

5. The flow control window size of a virtual channel is the maximum number of packets that it can have in transit within the communication network at the same time. Let N_k denote the window size of virtual channel k, for $k = 1,\ldots, K$. The number of available "slots" in the window is decremented by 1 whenever the source inputs a packet into the network. When the window is full, the data source is quiesced. The number of available slots is incremented by 1 when an ETE acknowledgement is received from the sink. If the window is not full, a new packet is generated by source k for input into the network at the rate γ_k. The physical interpretation of γ_k depends upon the loading on the virtual channels. For a lightly loaded network, γ_k is the external arrival rate of packets to virtual channel k. For a heavily loaded network such that the source queue is non-empty most of the time, γ_k is the reaction speed of the data source to a signal (or message) from the network interface of the virtual channel authorizing new input. (See Fig. 1.)

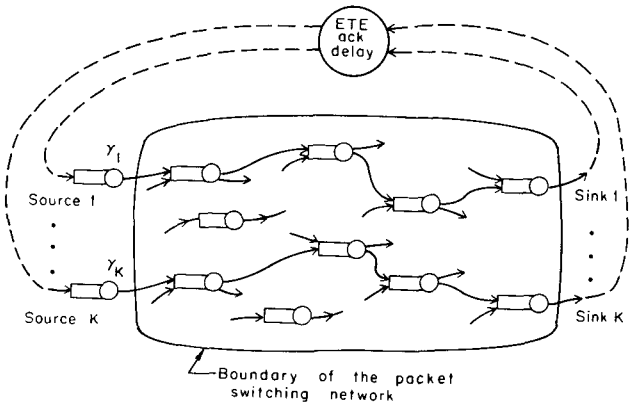

Fig. 1. An illustration of the queueing network model.

In summary, a packet switching network with flow controlled virtual channels is modeled by a closed multi-chain queueing network. Each closed routing chain corresponds to a virtual channel with the chain population size equal to the virtual channel window size. Given the above assumptions, the equilibrium probability distribution of queue lengths in the network is given by the product-form solution [BASK 75]. Let n_{mk} denote the number of chain k customers (packets) at server m. Define

$$\underline{n}_m = (n_{m1}, n_{m2},\ldots, n_{mK})$$
and
$$\underline{n} = (\underline{n}_1, \underline{n}_2,\ldots \underline{n}_M) \quad (1)$$

where M is the total number of servers in the model. (Note that the ETE acknowledgement delay is considered to be a server in the model.) Define

$$\underline{N} = (N_1, N_2,\ldots, N_K) \quad (2)$$

where N_k is the population size of chain k in the network (window size of virtual channel k). The product form solution is

$$P(\underline{n}) = \frac{p(\underline{n})}{G(\underline{N})} = \frac{p_1(\underline{n}_1)p_2(\underline{n}_2)\cdots p_M(\underline{n}_M)}{G(\underline{N})} \quad \text{for } \underline{n} \text{ feasible} \quad (3)$$

where $\underline{n}$ is feasible if $n_{mk} \geq 0$ for all m and k and $\sum_{m=1}^{M} \underline{n}_m = \underline{N}$, $p_m(\underline{n}_m)$ is the (improper) equilibrium probability distribution of queue lengths at server m and $G(\underline{N})$ is the normalization constant obtained by summing $p(\underline{n})$ over all feasible $\underline{n}$.

Note that p_m is a K-dimensional array indexed between $\underline{0}$ and $\underline{N}$, where $\underline{0}$ is a K-vector of all zeros. The convolution of two such functions, say p_1 and p_2, defines a real-valued function, say g_2, over the same domain, as follows

$$g_2(\underline{i}) = \sum_{j_1=0}^{i_1} \cdots \sum_{j_K=0}^{i_K} p_1(\underline{j}) p_2(\underline{i}-\underline{j})$$

$$\text{for } \underline{0} \le \underline{i} \le \underline{N} \qquad (4)$$

In shorthand notation, (4) will be written as

$$g_2 = p_1 \circledast p_2 = p_2 \circledast p_1. \qquad (5)$$

We note that $G(\underline{N})$ is simply an element of the array

$$g_{\{1,2,\ldots,M\}} = p_1 \circledast p_2 \circledast \cdots \circledast p_M. \qquad (6)$$

The convolution algorithm in [REIS 75] solves for $G(\underline{N})$ by performing the convolutions in (6) sequentially as follows

$$g_{\{1,2,\ldots,m\}} = g_{\{1,2,\ldots,m-1\}} \circledast p_m$$

$$\text{for } m = 2,3,\ldots,M \qquad (7)$$

The time requirement to compute the array $g_{\{1,2,\ldots,M\}}$ and hence $G(\underline{N})$ is of the order of

$$(M-1) \prod_{k=1}^{K} \frac{(N_k+2)(N_k+1)}{2}$$

while the space requirement is of the order of $2 \prod_{k=1}^{K} (N_k+1)$. Note that both requirements grow exponentially with K. Given that the servers in the network have fixed service rates (i.e. independent of queue length), the convolution operation can be performed much faster than (4) by the technique of feedback filtering [REIS 75]; the time and space requirements are then of the order of $MK \prod_{k=1}^{K} (N_k+1)$ and $\prod_{k=1}^{K} (N_k+1)$ respectively, which is a substantial improvement, but still growing exponentially with K. The MVA algorithm of Reiser and Lavenberg [REIS 80] bypasses the evaluation of the normalization constant $G(\underline{N})$ and computes the performance measures of mean queue lengths and chain throughputs directly. (It avoids a problem of the convolution algorithm, namely, the occurrence of overflows/underflows [LAM 80].) For fixed-rate servers, its time and space requirements are both of the order $MK \prod_{k=1}^{K} (N_k+1)$ which also grow exponentially with K.

With either the convolution or MVA algorithm (or any one of their variants), the time and space requirements will be beyond the limits of present computers when network models with 10 or more virtual channels are considered. Some approximate solution techniques have been considered. Pennotti and Schwartz [PENN 75] analyzed the model of a single flow-controlled virtual channel with the traffic of all other virtual channels combined and modeled by an open chain. Reiser proposed an efficient heuristic solution technique based upon the MVA algorithm and reported accuracies in the neighborhood of 5% for virtual channel throughputs and 10% for virtual channel transit delays. (The networks considered have 18 communication channels and 42 virtual channels.) [REIS 79].

A computational algorithm, called the tree convolution algorithm, was recently reported by these authors in [LAM 81a]. It is intended for the solution of networks in which routing chains do not visit all servers (or service centers) in the network. In models of communication networks and distributed systems, it is often true that chains visit only a small fraction of all queues in the network (sparseness property). Furthermore, chains are often clustered in certain parts of the network and their routes are constrained by the network topology (locality property). By making use of the routing information of chains, the time and space requirements of the tree convolution algorithm can be made substantially less than those of the (sequential) convolution and MVA algorithms. The number of routing chains that can be handled varies depending upon the extent of sparseness and locality present in their routes. We have solved many numerical examples with 32-50 routing chains. In some cases, the solution of networks with up to 100 routing chains has been found to be possible.

3. EXACT ANALYSIS BY THE TREE CONVOLUTION ALGORITHM

The tree convolution algorithm provides an exact solution of normalization constants and performance measures for product-form queueing networks. It is based upon two ideas. First, we note that the convolution in (6) can be performed in any order to obtain $g_{\{1,2,\ldots,M\}}$. Specifically, in the tree convolution algorithm, the arrays $\{p_m\}$ are placed at the leaf nodes of a tree. (See Figure 2.)

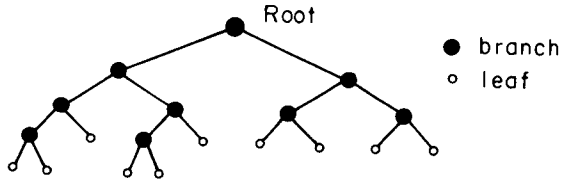

Fig. 2. A tree.

Each node in the tree corresponds to a subset of service centers that are descendants of that node. To compute the array $g_{\{1,2,\ldots,M\}}$, visit all nodes in the tree according to some order of tree traversal. The root node is visited last. A branch node may be visited only after all its sons have been visited. When a branch node is visited, an array g_{SUBNET} is computed for the node from the arrays $g_{SUBNET1}$ and $g_{SUBNET2}$ of its sons by

$$g_{SUBNET} = g_{SUBMET1} \circledast g_{SUBNET2} \qquad (8)$$

where SUBNET = SUBNET1 $\cup$ SUBNET2. If the node has more than two sons, then convolutions are performed sequentially one after the other. Finally, when the root node is visited, the array $g_{\{1,2,\ldots,M\}}$ is obtained.

Both the tree convolution algorithm and the sequential convolution algorithm require M-1

convolutions. In fact, the sequential convolution algorithm is just a special case of the tree algorithm for the tree shown in Figure 3. However, substantial time and space savings can be achieved by the general tree algorithm by making use of the following additional observation.

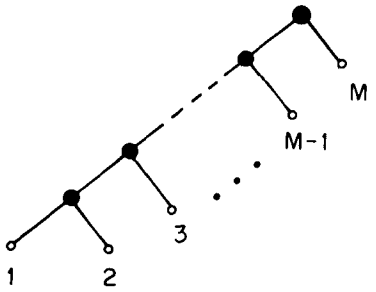

Fig. 3. Tree for the sequential convolution algorithm.

Consider routing chain k. Let CENTERS(k) be the set of service centers visited by chain k. Let SUBNET denote a subset of the M service centers. With respect to SUBNET, chain k is said to be fully covered if CENTERS(k) $\subseteq$ SUBNET; chain k is said to be noncovered if the intersection of CENTERS(k) and SUBNET is null; otherwise, chain k is said to be partially covered.

Partition the set of K chains into the following 3 sets with respect to SUBNET:

σ_{pc} = {k|chain k is partially covered by SUBNET}

σ_{fc} = {k|chain k is fully covered by SUBNET}

σ_{nc} = {k|chain k is noncovered by SUBNET}

Now make the observation that only those elements of g_{SUBNET} corresponding to the following index values are needed for further convolutions to arrive at $G(\underline{N})$,

$$\{i_k = 0,1,2,\ldots,N_k,\ k\epsilon\sigma_{pc};\ i_k = N_k,\ k\epsilon\sigma_{fc};$$
$$i_k = 0,\ k\epsilon\sigma_{nc}\}.$$

Let $|\sigma|$ denote the cardinality of set σ. For the purpose of computing $G(\underline{N})$, it is sufficient to store g_{SUBNET} as an array with dimensionality $|\sigma_{pc}|$ indexed by $\underline{i}_{pc} = \{i_k,\ k\epsilon\sigma_{pc}\}$. Such an array is termed a partially covered array. The amount of space needed for a partially covered array is $\prod_{k\epsilon\sigma_{pc}} (N_k+1)$ locations. (Additionally, a small amount of space is also needed to store σ_{pc}. The time requirement of the convolution in (8) using partially covered arrays is shown in [LAM 81a].)

Given a subset of centers in a network that has many centers and sparse routing chains, it is highly likely that only a few chains will be partially covered by the subset. Thus, for queueing networks with properties of sparseness and locality, the time and space savings from the use of partially covered arrays instead of K-dimensional arrays can be very substantial.

The actual time and space needed for the tree convolution algorithm depend upon the following "tree planting" decisions: the tree configuration, the order of tree traversal and the placement of service centers at leaf nodes. The objective of tree planting is to minimize the overall space and time needed by the algorithm by minimizing the numbers of partially covered chains in subnets associated with branch nodes in the tree.

In our current implementation of the tree algorithm, we employ a preprocessor to plant the tree as well as to calculate the total space and time requirements of that tree prior to actually performing convolutions. Tree planting algorithms are addressed in [LAM 81a, LIEN 81].

In addition to substantial space and time savings, the tree convolution algorithm has several significant advantages over the other algorithms. First, the tree of partially covered arrays employed by the algorithm provides a very flexible data structure for tailoring time-space tradeoffs to individual queueing networks in the calculation of network performance measures. It will also facilitate the solution of very large queueing networks with the help of storage management techniques or by means of parallel computation on a multi-processor machine. Furthermore, the computation of the marginal distribution of queue lengths in a service center is obtained with $(\log_2 M)$-1 convolutions instead of M-1 convolutions needed by a sequential convolution algorithm, where M is the total number of service centers and a balanced binary tree is assumed.

An illustration of the time and space savings of the tree convolution algorithm for an exact solution of a communication network model with 64 queues and 32 closed routing chains (flow-controlled virtual channels) is shown in Table 1. An analysis of the expected time and space requirements of the algorithm as a function of the sparseness of routing chains is presented in [LAM 81c].

4. APPROXIMATE MODELING WITH EQUIVALENT OPEN CHAINS

With the tree convolution algorithm, communication network models with many queues and a relatively large number of sparse routing chains can be solved. (Chains are said to be sparse if the average number of queues visited by a chain is much smaller than the number of queues in the network.) Both throughputs and mean transit delays for individual chains can be calculated exactly with time and space requirements within the limits of present computers. However, these time and space requriements are still fairly large. Hence, the tree algorithm is not very practical for use within optimization procedures for network design or routing assignment. Approximate models are needed for such purposes. The tree convolution algorithm, however, facilitates checking the accuracy of such approximate models. An exact analysis may be performed using the tree convolution algorithm only at various "checkpoints" of the optimization procedures instead of at every perturbation step.

An approximate model that has been considered by Pennotti and Schwartz [PENN 75] as well as Gerla and Nilsson [GERL 80] is that of a queueing network with open chains; the throughput of each open chain is made equal to that of the corresponding closed chain, which can be calculated exactly using the tree convolution algorithm. The question we address next is: how good is such an approximation? We shall measure the accuracy of the approximate model by comparing the mean network transit delays for individual chains given by the tree convolution algorithm and delay estimates calculated using the M/M/1 delay formula for the approximate model of open chains.

One set of such results is shown in Table 2. The network model considered has 64 communication channels and 32 virtual channels. The source arrival rate is assumed to be γ = 1 packet/second for all chains. The service rate for each communication channel is assumed to be μ = 10 packets/second. The mean ETE acknowledgement delay for virtual channel k is assumed to be h_k/μ, where h_k is the number of channels in the route of virtual channel k. The virtual channel window size is N_k = 3 for all k. The throughputs, mean delays and delay estimates of the 32 virtual channels are shown in Table 2. The average utilization of the 64 communication channels is 0.185, with a maximum utilization of 0.281 and a minimum utilization of 0.082 and a standard deviation of 0.066. (The tree convolution algorithm employed an unbalanced binary tree and had a space requirement of 2048 array locations and a time requirement of 66,298 operations (multiplications and divisions).)

The percentage errors in the mean delay estimates are quite small in this case. We next proceed to investigate the effect of varying the relative source and channel speeds γ and μ. We vary γ from 10 to .5 while keeping μ and all the other parameters constant. The results are shown in Table 3. Note that the channel utilizations are highest at γ = 10 and lowest at γ = 0.5. The accuracy of the approximate model is very poor for γ = 10 (same value as μ) and improves as γ decreases. There are two possible reasons for this behavior. First, when the utilization of a M/M/1 queue is high, its delay distribution has a long tail, which gives rise to a poor estimate of the delay in a closed network where the queue lengths are bounded. Second, when $\gamma = \mu$, the bottleneck in a routing chain is at one of the communication channels within the network. On the other hand, when μ is much smaller than γ, the source is the bottleneck in a routing chain and it behaves like a Poisson source at rate γ much of the time (i.e. like an open chain).

We next investigate the effect of varying the virtual channel window size. Window sizes of 2, 3 and 4 are considered. (See Table 4.) All other parameters remain the same as for Table 2. Note that as the window size increases, the accuracy of the approximate model improves, despite increases in the channel utilizations. Our results complement the results of Lavenberg in [LAVE 80a] for a somewhat different approximation technique. He considered only one or two routing chains but with hundreds of customers in each chain. We have considered a large number of chains but only a few customers in each chain.

Finally, we consider the effect of poor route assignments. We took the same network considered above and rerouted some of the chains to give rise to a bigger variance in the chain utilizations. Results for γ = 1 or 2 and a window size of 2 or 3 are shown in Table 5. In these cases, we note that the accuracy of the approximate model suffers from the new route assignment, in particular for the cases of γ = 2.

An important observation from these numerical results is that in almost all cases considered the mean delay estimates are larger than the actual mean delays. (The only exception was observed in the case of $\gamma = \infty$ that gives rise to errors of a few hundred percent, thus rendering the approximate model meaningless.) There are two reasons for the approximate model to overestimate mean delays. First, delay estimates are obtained from M/M/1 queue delay distributions that have long tails. Second, the mean-value analysis shows that the mean delay encountered by a closed chain customer is determined by the mean queue lengths of a network with that customer removed [REIS 80]. The approximate model as described above does not account for this behavior.

In light of the above observation and the results in Table 5, we see that the impact of bottlenecks on chain delays in the approximate model of open chains is exaggerated compared to that in the original model of closed chains. This behavior should be kept in mind when the approximate model is used for route optimization.

5. AN OPTIMAL ROUTING CRITERION

The Flow Deviation Method [FRAT 73, GERL 73, KLEI 76] provides a distance metric for selecting the shortest path for the addition of an infinitesimal amount of traffic into a network. It was shown that if the distance metric of a communication channel is chosen to be the derivative of the mean network transit delay T with respect to the flow in the channel, then the infinitesimal change in delay ΔT caused by the added traffic is minimized.

We next pose a similar problem for networks with flow-controlled virtual channels. We shall consider the perturbation due to the addition of a new virtual channel with a window size of one to an existing network (instead of an infinitesimal amount of flow).

It is obvious that the mean network transit delay T may be calculated using the tree convolution algorithm for the network both with and without the additional virtual channel (given a specific route for it). However, to determine the optimal route would require numerous applications of the tree convolution algorithm and would be very expensive in terms of computational time.

Our approach is to consider a model of the network with equivalent open chains representing the existing flow-controlled virtual channels

(such as described in Section 4). The new virtual channel to be added is represented by a closed chain.

Let there be M_c queues in the model representing communication channels. The aggregate arrival rate of the open chain, representing existing traffic in the network, to channel m is denoted by λ_m packets/second. The service rate of channel m is μC_m packets/second where $1/\mu$ is the average length of a packet in bits and C_m is the channel speed in bits/second. Define

$$\rho_m = \lambda_m/(\mu C_m).$$

The total throughput rate at which open chain packets leave (or enter) the network is γ_o packets/second. The closed chain representing the virtual channel being added has a population size of one (i.e. window size is one), a source rate of γ packets/second and a mean ETE acknowledgement delay of τ seconds. The source and sink nodes of the virtual channel are known but its route is to be determined.

Let Q denote the set of communication channels constituting a route chosen for the new virtual channel. From the arrival theorem [SEVC 79, LAVE 80b], the mean delay encountered by the new virtual channel's packet at channel $m \in Q$ is $1/(\mu C_m - \lambda_m)$. The mean network transit delay of the new virtual channel is

$$T_c = \sum_{m \in Q} \frac{1}{\mu C_m - \lambda_m}$$

Applying Little's formula [LITT 61], the throughput rate of the new virtual channel is

$$\gamma_c = \frac{1}{(1/\gamma) + \tau + T_c} \tag{9}$$

Let T_o be the mean network transit delay and $\bar{n}_o$ be the mean number of packets in the network before the addition of the new virtual channel. The increase in the mean delay due to the new virtual channel is

$$\Delta T = \frac{\sum_{m \in Q} \Delta\bar{n}_{m,o} + \bar{n}_o + \gamma_c T_c}{\gamma_o + \gamma_c} - T_o \tag{10}$$

where $\Delta\bar{n}_{m,o}$ is the increase in the mean queue length of the open chain at channel m due to the new virtual channel, and is given by [PENN 75]

$$\Delta\bar{n}_{m,o} = \frac{\lambda_m}{\mu C_m - \lambda_m} \bar{n}_{m,c} \tag{11}$$

where $\bar{n}_{m,c}$ is the mean number of new packets (belonging to the added virtual channel) at channel m. An application of Little's formula yields

$$\bar{n}_{m,c} = \gamma_c/(\mu C_m - \lambda_m). \tag{12}$$

Finally, we have

$$\Delta T = \frac{\sum_{m \in Q} \frac{\lambda_m \gamma_c}{(\mu C_m - \lambda_m)^2} + \gamma_c T_c - \gamma_c T_o}{\gamma_o + \gamma_c}$$

$$= \frac{\sum_{m \in Q} \left[\frac{\lambda_m}{(\mu C_m - \lambda_m)^2} + \frac{1}{\mu C_m - \lambda_m} \right] - T_o}{(\gamma_o/\gamma_c) + 1}$$

$$= \frac{\sum_{m \in Q} \frac{\mu C_m}{(\mu C_m - \lambda_m)^2} - T_o}{\gamma_o \left(\frac{1}{\gamma} + \tau + \sum_{m \in Q} \frac{1}{\mu C_m - \lambda_m} \right) + 1} \tag{13}$$

To minimize ΔT, the route should be chosen to try to minimize the numerator and to maximize the denominator if possible. Minimizing the numerator implies the choice of a shortest path from source to destination using $\mu C_m/(\mu C_m - \lambda_m)^2$ as the distance metric. Note that this is essentially the same as the distance metric of

$$\frac{C_m}{(C_m - f_m)^2} \qquad \text{where } f_m = \lambda_m/\mu$$

given by the Flow Deviation method.

Maximizing the denominator, on the other hand, implies that the longest path should be chosen with $1/(\mu C_m - \lambda_m)$ as the distance metric.

It seems reasonable to impose a maximum delay bound τ_{max} on the mean delay of the new virtual channel to be added. If such a bound is desired, then the optimal routing problem can be posed as follows:

$$\underset{Q}{\text{Min}} \sum_{m \in Q} \frac{\mu C_m}{(\mu C_m - \lambda_m)^2} \tag{14}$$

subject to

$$\sum_{m \in Q} \frac{1}{\mu C_m - \lambda_m} < \tau_{max}. \tag{15}$$

A straightforward approach to solve the above problem is to find the i^{th} shortest path using $\frac{\mu C_m}{(\mu C_m - \lambda_m)^2}$ as a metric for i = 1,2,... and select the first one that meets the delay bound of (15).

6. CONCLUSIONS

Packet switching networks with flow-controlled virtual channels are modeled as queueing networks with closed routing chains. We discussed the computational difficulties of such models. The tree convolution algorithm recently developed by these authors [LAM 81a] was introduced for the exact solution of such models with a large number of queues and chains. The algorithm derives its efficiency from exploiting the sparseness and locality properties of routes typically present in communication networks.

An approximate model of equivalent open chains was examined and found to be applicable given certain conditions are met. The problem of adding a new virtual channel with a window size of one to an existing network was considered. The optimal routing criterion was found to be akin to the criterion given by the Flow Deviation Method. Additional work on this problem is being done.

REFERENCES

BASK 75 Baskett, F., K. M. Chandy, R. R. Muntz and F. Palacios, "Open, Closed, and Mixed Networks of Queues with Different Classes of Customers," JACM, April 1975.

CERF 74 Cerf, V. and R. Kahn, "A Protocol for Packet Network Intercommunication," IEEE Trans. on Communications, COM-22 (1974), 637-648.

FRAT 73 Fratta, L., M. Gerla and L. Kleinrock, "The Flow Deviation Method: An Approach to Store-and-forward Network Design," Networks 3 (1973) 97-133.

GALL 77 Gallager, R., "An Optimal Routing Algorithm Using Distributed Computation," IEEE Trans. on Commun., Vol. COM-25, Jan. 1977, pp. 73-85.

GERL 73 Gerla, M., "The Design of Store-and-forward Networks for Computer Communications," Ph.D. dissertation, Department of Computer Sciences, UCLA (Jan. 1973).

GERL 77 Gerla, M. and L. Kleinrock, "On the Topological Design of Distributed Computer Networks," IEEE Trans. on Communications, COM-25 (1977) pp. 48-60.

GERL 80 Gerla, M. and P. O. Nilsson, "Routing and Flow Control Interplay in Computer Networks," Proc. Fifth ICCC, Atlanta, Oct. 1980.

IBM 75 IBM Corp., Systems Network Architecture General Information, GA27-3102-0 (Jan. 1975).

KLEI 64 Kleinrock, L., Communication Nets -- Stochastic Message Flow and Delays (McGraw-Hill, New York, 1964).

KLEI 76 Kleinrock, L., Queueing Systems, Volume 2: Computer Applications (Wiley-Interscience, New York, 1976).

LAM 76 Lam, S. S., "Store-and-forward Buffer Requirements in a Packet Switching Network," IEEE Trans. on Communications COM-24 (1976) pp. 394-403.

LAM 80 Lam, S. S., "Dynamic Scaling and Growth Behavior of Queueing Network Normalization Constants," Technical Report TR-148 Department of Computer Sciences, University of Texas at Austin (June 1980), to appear in JACM.

LAM 81a Lam, S. S. and Y. L. Lien, "A Tree Convolution Algorithm for the Solution of Queueing Networks," Dept. of Computer Sciences, University of Texas at Austin, Technical Report TR-165, January 1981.

LAM 81b Lam, S. S. and J. W. Wong, "Queueing Network Models of Packet Switching Networks," Dept. of Computer Science, University of Waterloo, Ontario, Canada, Research Report CS-81-06, Feb. 1981.

LAM 81c Lam, S. S. and Y. L. Lien, "An Analysis of the Tree Convolution Algorithm for Queueing Networks," Dept. of Computer Sciences, Univ. of Texas at Austin, Technical Report TR-166, Feb. 1981.

LAVE 80a Lavenberg, S. S., "Closed Multichain Product Form Queueing Networks with Large Population Sizes," IBM T. J. Watson Research Center, Report RC8496, Sept. 1980.

LAVE 80b Lavenberg, S. S. and M. Reiser, "Stationary State Probabilities of Arrival Instants for Closed Queueing Networks with Multiple Types of Customers," J. Applied Probability, Dec. 1980.

LIEN 81 Lien, Y. L., "Modeling and Analysis of Flow-controlled Computer Communication Networks," Ph.D. Thesis, Dept. of Computer Sciences, University of Texas at Austin, 1981; in preparation.

LITT 61 Little, J. C., "A Proof of the Queueing Formula: $L = \lambda W$," Operations Research 9 (1961) pp. 383-387.

OPDE 74 Opderbeck, H. and L. Kleinrock, "The Influence of Control Procedures on the Performance of Packet-switched Networks," Proc. National Telecommunications Conference, San Diego (Dec. 1974).

PENN 75 Pennotti, M. and M. Schwartz, "Congestion Control in Store and Forward Tandem Links," IEEE Trans. on Communications COM-23 (1975) pp. 1434-1443.

POUZ 73 Pouzin, L., "Presentation and Major Design Aspects of the CYCLADES Computer Network," Proc. Third Data Communications Symposium, St. Petersburg, Florida (Nov. 1973).

REIS 75 Reiser, M. and H. Kobayashi, "Queueing Networks with Multiple Closed Chains: Theory and Computational Algorithms," IBM Journal of Research and Development 19 (1975).

REIS 79 Reiser, M., "A Queueing Network Analysis of Computer Communication Networks with Window Flow Control," IEEE Trans. on Communications COM-27 (1979) pp. 1199-1209.

REIS 80 Reiser, M. and S. Lavenberg, "Mean Value Analysis of Closed Multichain Queueing Networks," JACM 27 (1980) pp. 313-322.

ROBE 78 Roberts, L. G., "The Evolution of Packet Switching," Proc. IEEE, Vol. 66, Nov. 1978.

SCHW 77 Schwartz, M., Computer-Communication Network Design and Analysis, Prentice-Hall, 1977.

SEVC 79 Sevcik, K. C. and I. Mitrani, "The Distribution of Queueing Network States at Input and Output Instants," Proc. 4th Intl. Symp. on Modeling and Performance Evaluation of Computer Systems, Vienna, Austria, 1979.

	Tree convolution	Convolution	MVA
Time (multiplications and divisions)	1,065,008	3.78×10^{22}	7.62×10^{22}
Space (array locations)	1,376	1.84×10^{19}	9.89×10^{21} (upper bound)

Table 1. Time and space requirements for a network example with 64 queues and 32 closed chains.

chain	throughput rate	mean delay	delay estimate	% error
1	9.20e-01	6.02e-01	6.15e-01	2.28e+00
2	9.20e-01	6.02e-01	6.15e-01	2.28e+00
3	9.17e-01	6.25e-01	6.41e-01	2.49e+00
4	9.17e-01	6.25e-01	6.41e-01	2.49e+00
5	9.20e-01	6.03e-01	6.17e-01	2.27e+00
6	9.20e-01	6.03e-01	6.17e-01	2.27e+00
7	9.47e-01	4.96e-01	5.07e-01	2.11e+00
8	9.47e-01	4.96e-01	5.07e-01	2.11e+00
9	9.41e-01	5.36e-01	5.49e-01	2.57e+00
10	9.41e-01	5.36e-01	5.49e-01	2.57e+00
11	9.45e-01	5.06e-01	5.18e-01	2.29e+00
12	9.45e-01	5.06e-01	5.18e-01	2.29e+00
13	8.18e-01	9.91e-01	1.02e+00	2.85e+00
14	8.18e-01	9.91e-01	1.02e+00	2.85e+00
15	9.86e-01	2.73e-01	2.78e-01	1.74e+00
16	9.86e-01	2.73e-01	2.78e-01	1.74e+00
17	9.88e-01	2.44e-01	2.47e-01	1.21e+00
18	9.88e-01	2.44e-01	2.47e-01	1.21e+00
19	9.89e-01	2.33e-01	2.35e-01	1.01e+00
20	9.89e-01	2.33e-01	2.35e-01	1.01e+00
21	9.89e-01	2.32e-01	2.35e-01	1.03e+00
22	9.89e-01	2.32e-01	2.35e-01	1.03e+00
23	9.97e-01	1.23e-01	1.24e-01	7.36e-01
24	9.97e-01	1.23e-01	1.24e-01	7.37e-01
25	9.17e-01	6.25e-01	6.40e-01	2.50e+00
26	9.17e-01	6.25e-01	6.40e-01	2.50e+00
27	9.70e-01	3.79e-01	3.86e-01	1.84e+00
28	9.70e-01	3.79e-01	3.86e-01	1.84e+00
29	8.56e-01	8.46e-01	8.68e-01	2.60e+00
30	8.56e-01	8.46e-01	8.68e-01	2.60e+00
31	8.84e-01	7.57e-01	7.78e-01	2.73e+00
32	8.84e-01	7.57e-01	7.78e-01	2.73e+00

Errors in delay estimates

Average :	2.02e+00
Variance :	4.32e-01
Standard Deviation :	6.57e-01

Table 2. Mean delays and delay estimates for $\gamma = 1$ and a window size of 3.

Case of	utilizations of communication channels				% errors in delay estimates			
	mean	max.	min.	st. dev.	mean	max.	min.	st. dev.
$\gamma = 10$	0.466	0.808	0.134	0.188	40.3	127	15.3	26.3
$\gamma = 2$	0.299	0.469	0.112	0.107	7.34	9.86	4.74	1.37
$\gamma = 1$	0.185	0.281	0.082	0.066	2.02	2.85	0.74	0.66
$\gamma = 2/3$	0.130	0.195	0.061	0.047	0.80	1.39	0.22	0.34
$\gamma = 1/2$	0.100	0.148	0.048	0.036	0.40	0.77	0.09	0.19

Table 3. Channel utilizations and errors in chain delays for different values of γ.

Case of	utilizations of communication channels				% errors in delay estimates			
	mean	max.	min.	st. dev.	mean	max.	min.	st. dev.
window size = 2	0.159	0.248	0.064	0.057	4.02	4.73	2.94	0.46
window size = 3	0.185	0.281	0.082	0.066	2.02	2.85	0.74	0.66
window size = 4	0.197	0.294	0.092	0.070	0.88	1.72	0.18	0.45

Table 4. Channel utilizations and errors in chain delays for different window sizes.

Case of	utilizations of communication channels				% errors in delay estimates			
	mean	max.	min.	st. dev.	mean	max.	min.	st. dev.
γ = 1 and window size = 2	0.159	0.321	0.064	0.064	4.16	5.03	3.35	0.43
γ = 1 and window size = 3	0.185	0.281	0.082	0.074	2.11	2.96	1.01	0.60
γ = 2 and window size = 2	0.236	0.493	0.082	0.098	9.75	16.8	7.02	2.35
γ = 2 and window size = 3	0.297	0.596	0.11	0.12	8.17	13.6	5.30	2.07

Table 5. Channel utilizations and errors in chain delays for a network with some poor routes.