Presented at the International Conference on Communications, Chicago, June 1985

Models and Algorithms for the Design of Store-and-Forward Communication Networks*

Simon S. Lam and Ching-Tarng Hsieh

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

Abstract

The performance predictions of the open queueing network model and two closed queueing network models are compared with results from a realistic simulator of store-and-forward communication networks. The comparisons indicate that closed network models are far superior to the open network model. In particular, one of the closed network models is found to be better than any of the others considered. We then consider network design problems based upon a closed network model and present a solution to the optimal capacity assignment problem. We also describe a heuristic algorithm that makes use of a steepest descent criterion encountered in the optimal solution to search for optimal capacities. Being a heuristic, it can be used for a realistic model, i.e., a model with discrete capacity values, nonlinear cost functions, and upper bounds on the mean delays of virtual channels.

1. Introduction

A store-and-forward communication network consists of a set of switching nodes interconnected by communication channels. Host computers and terminals constitute sources and sinks of data messages to be transported by the network (Figure 1). The basic unit of data transfer within the network is called a packet. Each packet traverses from its source node to its destination node through a series of nodes and communication channels along its path (route). Queues of packets are formed, inside switching nodes, for transmission over communication channels.

Tools for the analysis and design of store-and-forward communication networks have been based primarily on the open queueing network model of Kleinrock [4]. In practice, almost all such communication networks have end-to-end flow control mechanisms for logical connections, to be referred to in this paper as *virtual channels*, between pairs of communicating sources and sinks. Such networks are more accurately modeled as closed queueing networks than as open queueing networks; each routing chain in a closed network model represents a flow-controlled virtual channel with the chain population size equal to the virtual channel window size [5,7,12]. The obstacle that currently prevents closed network models from being widely used by analysts and designers is the large computational time and space it requires to calculate network performance measures (throughputs and mean end-to-end delays of virtual channels). Some progress has been made to reduce these computational requirements and the solution of networks with many virtual channels is feasible [8,10]. Nevertheless, the computational requirements of closed network models are still substantially larger than those of the open network model.

The open network model, however, is unrealistic because it assumes that the number of packets traveling in a network, and belonging to the same virtual channel, is unlimited. In other words, virtual channels have no input control whatsoever. As a result, the open network model is accurate only for networks that are lightly utilized (when input control mechanisms have little or no effect on network performance).

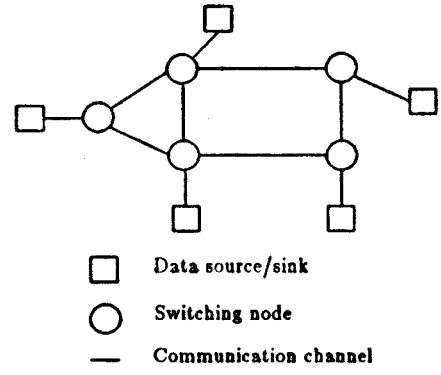

Fig. 1. A store-and-forward communication network.

In Section 2, we give an overview of the open network model and of closed network models. In Section 3, we examine the accuracy of these models by comparing the throughputs and mean delays of virtual channels in an 8-node network given by the models and by simulation. The results indicate that closed network models are far superior to the open network model. In particular, we found a closed network model that is better than any of the others considered. In Section 4, we consider network design problems based upon a closed network model and employing the tree convolution algorithm [8,10] for its solution. In particular, we present a solution to the optimal capacity assignment problem for a closed network model given assumptions of continuous capacities and linear cost functions. However, evaluation of the optimal capacities requires the numerical solution of nonlinear equations. We then describe a heuristic algorithm that makes use of a steepest descent criterion encountered in the optimal solution to search for optimal capacities. Being a heuristic, it can be used for a very realistic model, i.e., a model with discrete capacity values, nonlinear cost functions, and upper bounds on the mean delays of individual virtual channels. The algorithm can also be easily adapted to perform various network topology optimizations (e.g., adding and deleting communication links). On the other hand, the optimal solution is not guaranteed. But empirical results indicate that its capacity assignments are substantially different from, and better than, the square-root capacity assignment in [4].

2. Open and Closed Network Models

In a store-and-forward communication network, communication channels and nodal processors can be modeled as FIFO servers with exponentially distributed service times. We shall assume that the network has sufficient buffers so that blocking due to buffer overflow has negligible probability. (The problem of buffer requirements and loss probabilities has been considered in [6].) Also, the independence assumption of Kleinrock [4] is needed for both open and closed network models.

Suppose that there are K uni-directional virtual channels between source-sink pairs. Packets in the same virtual channel follow a fixed route. (Actually, the models do not preclude choosing such a route

* This research was supported by National Science Foundation Grant No. ECS-8304734.

CH2175-8/85/0000-1362 $1.00 © 1985 IEEE

probabilistically from a finite set of available routes between the source and sink)

Open network model

Each virtual channel is represented as an open routing chain, i.e., the external packet arrivals to the source node of virtual channel k constitute a Poisson process at a known 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. Let the rate of all packet arrivals to a server in the network, say server m, be λ_m packets/second. The work rate of server m is C_m bits/second and the average length of a packet is $1/\mu$ bits. The traffic intensity of server m is defined to be $\rho_m = \lambda_m/(\mu C_m)$. Given that $\rho_m < 1$ for all m, the throughput of each virtual channel is the same as its external input rate. And the mean end-to-end delay of its packets is the sum of the mean delays of the servers along its route. The mean delay of server m is given by the $M/M/1$ mean delay formula and is equal to $1/(\mu C_m - \lambda_m)$. Thus, both the throughputs and the mean end-to-end delays of virtual channels can be obtained very easily for the open network model.

Closed network models

The flow-control window size of a virtual channel limits the maximum number of packets that it can have in transit within the communication network at the same time. Let there be K virtual channels and let N_k denote the window size of virtual channel k, for $k=1,2,\ldots,K$. We model the generation of external arrivals to a virtual channel with an additional FIFO server, called the *source server* of the virtual channel. It has an infinite supply of packets and works at a rate of γ_k packets/second. (See Figure 2.) However, whenever the number of packets in transit within the virtual channel is equal to its window size, its source server is blocked and cannot work. A blocked source server is later unblocked when an end-to-end acknowledgement (ack) returns from the sink indicating receipt of a packet. Thus, the actual input rate of virtual channel k is equal to γ_k when its source server is unblocked and is equal to zero whenever its source server is blocked.

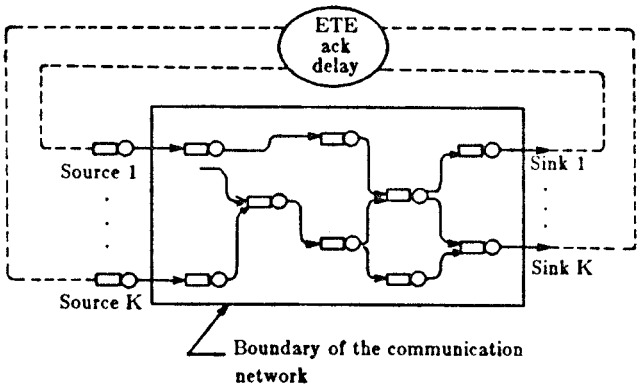

Fig. 2. A closed queueing network model of a store-and-forward communication network

The blocking behavior of a flow-controlled virtual channel is modeled in a queueing network by a closed routing chain with a fixed number of circulating customers. Each customer corresponds to an access token. Initially, N_k tokens are placed at the source server of virtual channel k. Each packet admitted into the network carries a token with it. When there is no more token at the source server, it is blocked. A packet arriving at the sink node of the virtual channel releases its token which is then carried back by an end-to-end ack to the source server to be reused again. Thus, the N_k circulating tokens of a virtual channel correspond to the N_k circulating customers of a closed chain.

We model the delay incurred by the return of an end-to-end ack from the sink of the packet being acknowledged to its source by an infinite-server (IS) service center [7,12]. For such IS servers, only mean delay values need to be specified and these mean values may be different for different virtual channels. It is not really important to model the storing and forwarding of the acks explicitly because these acks either are encoded in data packets travelling in the reverse direction of the virtual channel or, if sent separately, are very short. Thus, they consume relatively small amounts of buffer and channel resources in the network, which may be neglected or may be accounted for separately. The mean delays of end-to-end acks are not known a priori and can only be estimated. Fortunately, these estimates do not have to be highly accurate because the delay of end-to-end acks for a virtual channel impacts the network's performance only some of the time; in particular, it affects the actual input rate of the virtual channel only when its source server is blocked.

How should we estimate the mean delays of end-to-end acks? We considered two alternatives. First, suppose every ack is encoded (*piggybacked*) in a data packet that travels in the reverse direction of the virtual channel. Then, the ack delay is approximated by the delay of the virtual channel in the opposite direction. In this case, the closed network model has to be solved iteratively, i.e., start with some initial estimates of ack delays, calculate virtual channel delays and use them as new estimates for ack delays. We shall refer to this model as the *iterative closed model*. We have tried this iterative approach for several examples. In each case, the iteration converges to a consistent set of delay values. We did not attempt to prove convergence because, for reasons to be said below, we shall not adopt this model for our use.

It is well-known that an ack should be sent as a standalone packet whenever there is no data packet waiting to travel in the reverse direction (to avoid deadlocks). Our simulation results indicate that most end-to-end acks travel as standalone packets. (A somewhat surprising result! Of course, the exact percentage depends on the network traffic load.) Since standalone acks are very short and are given higher priority in our simulation than data packets, mean delays of end-to-end acks used in the iterative closed model are too high in comparison with simulation results. Further, iterative solutions are computationally expensive.

A second approach is to use a fixed estimate for the mean delay of end-to-end acks of each virtual channel. Obviously, this mean delay is proportional to the number of channels in the route of the acks (assuming that all channels have the same capacity). Hence, a reasonable estimate is: (number of channels in route)$\times\tau$, where τ is a parameter to be determined. Obviously, τ must be larger than the transmission time of a standalone ack. We have found experimentally that setting τ equal to the transmission time of a data packet in the closed network model gives quite accurate performance predictions. (See Section 3.) It does specify mean values for ack delays that are too high when the network load is light (when almost all end-to-end acks travel as standalone packets). But in a lightly loaded network, the network performance is highly insensitive to end-to-end ack delays.

To calculate the performance measures of a closed network model (throughputs and mean end-to-end delays of virtual channels), the computational time and space requirements of both the (sequential) convolution algorithm [11] and the MVA algorithm [13] grow exponentially with K; specifically, they are proportional to $\prod_{k=1}^{K}(N_k+1)$. These requirements are beyond the limits of present computers when network models with 10 or more virtual channels are considered.

The tree convolution algorithm, developed by Lam and Lien [8,10], is intended for the solution of networks in which chains do not visit all servers in the network. In models of communication networks, it is often the case 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 closed chains that can be handled varies depending upon the extent of sparseness and locality present in their routes. We have solved numerically many network examples with 32-50 routing chains. In some extreme cases, the solution of networks with up to 100 routing chains has been found to be possible [10].

3. Comparison of Models

In this section, we compare the performance predictions of the open network model and the two closed network models with those given by a network simulator developed by Lam and Lien previously [5]. The simulator implements most, if not all, of the important elements of a real network. In particular, it implements a data link protocol. The storing and forwarding of end-to-end acks, both standalone and piggybacked, are simulated. Virtual channels have a window mechanism for flow control. The length of a packet is sampled from any given probability distribution and remains constant for the entire journey of the packet through the network. Key differences among the models are shown in Table 1 below.

	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 any other	Exponential	Exponential

Table 1. Comparison of model features.

As we can see from Table 1, there remain significant differences between the closed network model and the simulator. Yet preliminary results from our experimental study indicate that the closed network model is quite accurate at all traffic levels. The experimental results for a specific network will be shown next.

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

Simulation results were obtained for three other packet length distributions in addition to the exponential distribution assumed in both open and closed network models.

hyper — a hyperexponential distribution with a coefficient of variation equal to 1.40

$burst_1$ — a distribution with 0.3 probability at 100 bits, 0.3 probability at 1900 bits, and a coefficient of variation equal to 0.77

$burst_2$ — a distribution with 0.4 probability at 50 bits, 0.4 probability at 1950 bits, and a coefficient of variation equal to 0.88

Note that the exponential distribution has a coefficient of variation equal to one. The mean value is the same in all cases. Additional details of the network example and packet length distributions can be found in [9].

The service time of the source server of each virtual channel models the packet generation time of the external source of the virtual channel. We used exponentially distributed packet generation times in our simulations, the same as what is assumed in queueing network models. All virtual channels in the network example have the same packet generation rate, or external packet arrival rate, denoted by γ.

Packet arrival rate	Network throughput						
	closed model		open model	simulation model			
	iter	fixed		exp	hyper	$burst_1$	$burst_2$
$\gamma = 1/2$	8.64	8.66	9	8.81	8.75	8.64	8.80
$\gamma = 2/3$	11.32	11.36	12	11.75	11.68	11.62	11.67
$\gamma = 1$	16.25	16.41	18	16.82	16.39	17.21	17.00
$\gamma = 2$	27.09	28.27	36	29.20	27.14	29.67	29.25
$\gamma = 3$	33.21	35.71	54	36.34	34.08	38.27	37.19
$\gamma = 4$	36.81	40.38	64	40.45	37.26	43.04	41.59

Table 2. Network throughputs given by different models for different values of γ.

Packet arrival rate	Mean end-to-end virtual channel delay							
	closed model		open model	open model (ideal)	simulation model			
	iter	fixed			exp	hyper	$burst_1$	$burst_2$
γ=1/2	0.24	0.24	0.240	0.244	0.245	0.273	0.238	0.246
γ=2/3	0.25	0.25	0.251	0.255	0.262	0.285	0.249	0.242
γ=1	0.27	0.271	0.279	0.281	0.270	0.305	0.266	0.271
γ=2	0.316	0.325	0.426	0.366	0.341	0.383	0.313	0.328
γ=3	0.340	0.362	1.214	0.449	0.368	0.391	0.344	0.359
γ=4	0.353	0.385	∞	0.517	0.387	0.418	0.366	0.379

Table 3. Mean end-to-end delays given by different models for different values of γ.

Packet arrival rate	Average channel utilization						
	closed model		open model	simulation model			
	iter	fixed		exp	hyper	$burst_1$	$burst_2$
$\gamma = 1/2$	0.093	0.093	0.095	0.094	0.093	0.092	0.095
$\gamma = 2/3$	0.122	0.123	0.127	0.127	0.125	0.125	0.123
$\gamma = 1$	0.177	0.178	0.190	0.180	0.176	0.186	0.183
$\gamma = 2$	0.297	0.310	0.380	0.316	0.291	0.321	0.318
$\gamma = 3$	0.363	0.392	0.570	0.391	0.363	0.415	0.402
$\gamma = 4$	0.400	0.441	0.667	0.432	0.392	0.465	0.445

Table 4. Average channel utilizations given by different models for different values of γ.

As γ is increased, the load on the network increases. In Tables 2-4, we compare the various models for γ = 1/2, 2/3, 1, 2, 3 and 4 packets/second. Network throughputs given by the models are shown in Table 2. Mean network delays are shown in Table 3. Average utilizations of communication channels are shown in Table 4. Generally, the predictions of all models are very close to each other when γ is small. As γ increases, the predictions of the open network model begin to deviate significantly from those of the other models (because it does not model flow control). At γ=4, it actually predicts that the mean network delay is infinite because in the open network model, in the absence of input control, some of the communication channels have a traffic intensity of one. In Table 3, there is a model referred to as *open model (ideal)*. It is an open network model where the throughputs of the virtual channels are made equal to the throughputs of virtual channels in the best available closed network model. This is called an ideal model because it is not feasible without solving a closed network model first. Yet, as we can see from Table 3, the ideal open model still overestimates mean network delays significantly for large values of γ.

Let us use the simulation model with exponentially distributed packet lengths as the benchmark for comparison. Comparing the two closed network models with this benchmark, we see that the closed model with fixed estimates for ack delays is excellent. (τ was set equal to the transmission time of a data packet.) It underestimates network throughputs and delays slightly when γ is small because, as explained earlier, it overestimates the ack delays when the network is lightly loaded. The iterative closed network model is not as good and it underestimates network throughputs and delays at all levels of network load (because it generally overestimates ack delays). Referring to Table 1, the above observations indicate that the accuracy of closed network models does not seem to be significantly affected by the independence assumption and their abstraction of the end-to-end acks

Next, let us look at the results in Tables 2-4 given by the simulator for other packet length distributions. We observe that the network performance does depend to some extent on the coefficient of variation of the packet length distribution. Not surprisingly, the network performance is better (i.e., larger throughputs and smaller delays) for a smaller coefficient of variation. For the range of coefficients of variation considered, 0.77 to 1.40, we consider the predictions of the closed network model (fixed) to be sufficiently robust.

Virtual channel	Throughput of virtual channel						
	closed model		open model	simulation model			
	iter.	fixed		exp.	hyper.	$burst_1$	$burst_2$
VC_1	2.04	2.20	3	2.13	2.13	2.44	2.22
VC_2	1.71	1.78	3	2.00	1.91	2.14	1.95
VC_3	2.11	2.25	3	2.23	1.96	2.27	2.29
VC_4	1.59	1.70	3	1.76	1.80	1.88	1.84
VC_5	1.71	1.78	3	2.04	1.95	2.01	1.92
VC_6	1.89	2.09	3	2.03	1.84	2.17	2.14
VC_7	1.75	1.93	3	1.94	1.71	2.09	1.97
VC_8	2.04	2.20	3	2.12	1.95	2.26	2.21
VC_9	1.75	1.93	3	1.81	1.65	2.02	1.89
VC_{10}	2.04	2.20	3	2.14	2.10	2.34	2.19
VC_{11}	1.71	1.78	3	1.96	1.95	2.00	2.01
VC_{12}	2.11	2.25	3	2.41	2.03	2.37	2.33
VC_{13}	1.59	1.70	3	1.80	1.82	1.84	1.89
VC_{14}	1.71	1.78	3	2.04	1.97	1.94	2.06
VC_{15}	1.89	2.09	3	2.05	1.86	2.11	2.10
VC_{16}	1.75	1.93	3	1.99	1.77	2.10	1.96
VC_{17}	2.04	2.20	3	2.09	1.95	2.32	2.27
VC_{18}	1.75	1.93	3	1.79	1.73	1.98	1.97
Total	33.21	35.71	54	36.34	34.08	38.27	37.19

Table 5. Virtual channel throughputs given by different models for γ=3 packets/second.

	Mean end-to-end delay							
	closed model		open model	open model (ideal)	simulation model			
	iter	fixed			exp	hyper	$burst_1$	$burst_2$
VC_1	0.456	0.482	1.286	0.571	0.536	0.529	0.474	0.491
VC_2	0.125	0.127	0.250	0.168	0.125	0.131	0.128	0.125
VC_3	0.430	0.444	0.750	0.512	0.458	0.530	0.468	0.462
VC_4	0.148	0.156	1.000	0.225	0.155	0.162	0.154	0.169
VC_5	0.125	0.127	0.250	0.168	0.122	0.124	0.131	0.131
VC_6	0.529	0.571	2.250	0.705	0.596	0.668	0.544	0.562
VC_7	0.346	0.372	2.000	0.490	0.372	0.402	0.333	0.389
VC_8	0.456	0.482	1.143	0.571	0.523	0.570	0.471	0.487
VC_9	0.346	0.372	2.000	0.490	0.362	0.418	0.365	0.382
VC_{10}	0.456	0.482	1.286	0.571	0.517	0.522	0.471	0.495
VC_{11}	0.125	0.127	0.250	0.168	0.123	0.125	0.122	0.112
VC_{12}	0.430	0.444	0.750	0.512	0.461	0.562	0.432	0.450
VC_{13}	0.148	0.156	1.000	0.225	0.163	0.153	0.147	0.161
VC_{14}	0.125	0.127	0.250	0.168	0.130	0.141	0.124	0.125
VC_{15}	0.529	0.571	2.250	0.705	0.611	0.639	0.545	0.566
VC_{16}	0.346	0.372	2.000	0.490	0.369	0.404	0.328	0.370
VC_{17}	0.456	0.482	1.143	0.571	0.496	0.540	0.456	0.481
VC_{18}	0.346	0.372	2.000	0.490	0.385	0.386	0.346	0.354
Average	0.340	0.362	1.214	0.449	0.368	0.391	0.344	0.359

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

In Tables 5 and 6, we show the throughputs and mean delays of *individual virtual channels* predicted by the various models for $\gamma = 3$ packets/second. The best model, closed model (fixed), gives predictions that differ from simulation results (all four cases) by less than 10 percent in most cases and less than 15 percent with only a few exceptions. (These exceptions arise when we compare it with the hyper and $burst_1$ models having coefficients of variation far away from unity.) For individual virtual channels, we consider such predictions to be very good and the model robust.

4. Capacity Assignment and Network Design

The open network model has a closed-form delay formula which encapsulates the parameters of the model and which plays a central role in network design and optimization procedures [2]. In a closed network model we made the observation that the tree of arrays in the tree convolution algorithm for solving closed network models can play a similar role in network design problems [9]. Due to space limitation, we shall consider only the channel capacity assignment problem in the balance of this paper.

The capacity assignment problem, in general, may be formulated as follows:

Given: Network topology, routes and window sizes of virtual channels

Maximize: Total network throughput $T(\underline{N})$

Design variables: Channel capacities C_i, $i=1,2,..,M$

Subject to: $\text{Cost} = \sum_{i=1}^{M} d_i(C_i)+F_i \leq D_{\max}$

$D_k(\underline{N}) \leq$ maximum allowable delay of virtual channel k, $k=1,2,..K$

$C_i > 0$ for all $i=1,2,...,M$

where F_i is a fixed cost for installing channel i,

$d_i(\cdot)$ is a cost function that is proportional to channel capacity,

$\underline{N}$ is the vector of window sizes,

and $D_k(\underline{N})$ is the mean end-to-end delay of virtual channel k.

The capacity assignment problem based upon the open network model was solved by Kleinrock [4] using the method of Lagrange multipliers. In Section 4.1 we review the product-form solution for closed queueing networks. In Section 4.2 we provide a solution to the capacity assignment problem based upon a closed network model. In Section 4.2, we present a heuristic algorithm for capacity assignment.

4.1. Product form solution for closed networks

In a closed queueing network with M service centers, K routing chains and N_k customers, $k=1,2,...,K$, for chain k, the product-form solution is [1,7]

$$P(\underline{n}) = \frac{p(\underline{n})}{G(\underline{N})}$$

$$= \frac{p_1(\underline{n}_1)p_2(\underline{n}_2)\cdots p_M(\underline{n}_M)}{G(\underline{N})} \qquad \text{for } \sum_{m=1}^{M} \underline{n}_m = \underline{N} \tag{1}$$

where $\underline{N}=(N_1,N_2,\ldots,N_K)$ is the vector of chain population sizes,

n_{mk}, $m=1,2,...,M$, $k=1,2,...,K$, is the number of chain k customers at service center m,

$\underline{n}_m=(n_{m1},n_{m2},\ldots,n_{mK})$, $m=1,2,...,M$,

$\underline{n}=(\underline{n}_1,\underline{n}_2,\ldots,\underline{n}_M)$,

$p(\underline{n})$ is the improper product-form probability density function,

$G(\underline{N})=\sum_{\text{all feasible states}} p(\underline{n})$ is the normalization constant for population vector $\underline{N}$, and the improper probability density function of queue lengths at service center m is given by

$$p_m(\underline{n}_m)=n_m!\prod_{k=1}^{K}\frac{\left(\frac{\lambda_{mk}}{\mu_{mk}C_m}\right)^{n_{mk}}}{n_{mk}!} \tag{2}$$

where $n_m = n_{m1} + n_{m2} + \cdots + n_{mK}$,

λ_{mk} is the relative arrival rate of chain k customers to service center m

C_m is the capacity in bits per second of channel m,

and $\mu_{mk} C_m$ is the service rate of channel m for chain k packets in packets per second.

The chain throughput $T_k(\underline{N})$ in terms of normalization constants is given by

$$T_k(\underline{N}) = \frac{G(\underline{N}-\underline{1}_k)}{G(\underline{N})} \qquad \underline{N} \geq \underline{1}_k, \quad k=1,2,\ldots,K \tag{3}$$

where $\underline{N}-\underline{1}_k$ is the population vector resulting from subtracting one from the kth element of $\underline{N}$. Also, the relative arrival rate of customers to the source server of virtual channel k is set equal to 1 for every k. The network throughput is obtained by summing over all chains.

$$T(\underline{N}) = \sum_{k=1}^{K} \frac{G(\underline{N}-\underline{1}_k)}{G(\underline{N})} \tag{4}$$

From Little's formula, the chain delay is given by

$$D_k(\underline{N}) = \frac{N_k}{T_k(\underline{N})} = \frac{N_k\, G(\underline{N})}{G(\underline{N}-\underline{1}_k)} \tag{5}$$

and the average network delay is

$$D(\underline{N}) = \sum_{k=1}^{K} \frac{T_k(\underline{N})}{T(\underline{N})} D_k(\underline{N}) \tag{6}$$

4.2. Optimality condition for capacity assignment

As in the case of the open network model, the Lagrange multipliers method is used. For the analysis that immediately follows, we assume that channel capacities are available for any positive value as well as linear cost functions $d_i(C_i) = d_i C_i$. Additionally, the end-to-end delays of virtual channels are not constrained. The following expression is to be minimized

$$Y = -T(\underline{N}) + \beta \left[\sum d_i C_i - D_{\max} \right] \tag{7}$$

Take the derivative of Y with respect to C_i

$$\frac{\partial Y}{\partial C_i} = -\frac{\partial T(\underline{N})}{\partial C_i} + \beta d_i \tag{8}$$

Since $T(\underline{N})$ is a function of normalization constants the partial derivative of $G(\underline{N})$ with respect to C_i is first derived. By observation, $G(\underline{N})$ and other performance measures of the closed network model are continuous functions of C_i. Differentiating the normalization constant with respect to C_i, and after some simplifications, we get

$$\frac{\partial G(\underline{N})}{\partial C_i} = \frac{-G(\underline{N})}{C_i} q_i(\underline{N}) \tag{9}$$

where $q_i(\underline{N}) = \sum_{k=1}^{K} q_{ik}(\underline{N})$ is the mean queue length of service center i.

The partial derivative of total throughput with respect to C_i is derived to be

$$\frac{\partial T(\underline{N})}{\partial C_i} = \frac{T(\underline{N})}{C_i} \left[q_i(\underline{N}) - \frac{\sum_{k=1}^{K} T_k(\underline{N}) q_i(\underline{N}-\underline{1}_k)}{T(\underline{N})} \right]$$

$$= \frac{T(\underline{N})}{C_i} \left[q_i(\underline{N}) - \overline{q}_i(\underline{N}) \right] \tag{10}$$

where $\overline{q}_i(\underline{N})$ is the mean queue length of service center i if one customer, from any chain, is removed from the network.

Put $\partial T(\underline{N})/\partial C_i$ into Eq. (7),

$$\frac{\partial Y}{\partial C_i} = \frac{T(\underline{N}) \left[\overline{q}_i(\underline{N}) - q_i(\underline{N}) \right]}{C_i} + \beta d_i = 0$$

$$C_i d_i = \frac{-T(\underline{N}) \left[\overline{q}_i(\underline{N}) - q_i(\underline{N}) \right]}{\beta} \tag{11}$$

Summing Eq. (11) over all channels

$$\sum_{i=1}^{M} C_i d_i = \frac{-\sum T(\underline{N}) \left[\overline{q}_i(\underline{N}) - q_i(\underline{N}) \right]}{\beta} = D_{\max}$$

$$\beta = \frac{-T(\underline{N}) \sum_{i=1}^{M} \left[\overline{q}_i(\underline{N}) - q_i(\underline{N}) \right]}{D_{\max}}$$

Put β into Eq. (11), we obtain the following condition for optimal network throughput

$$C_i = \frac{D_{\max} \left[\overline{q}_i(\underline{N}) - q_i(\underline{N}) \right]}{d_i \sum_{m=1}^{M} \left[\overline{q}_m(\underline{N}) - q_m(\underline{N}) \right]} \tag{12}$$

Note that in Eq. (12), the right hand side is a function of $C_j, j=1,2,..,M$. Thus to get the set of optimal capacities, a numerical solution of the set of equations given by (12) is required. A more detailed derivation of the equations in this section can be found in [9].

4.3. Heuristic algorithm

In the design of a real network, channel capacities are only available in certain discrete values, channel capacity cost functions are not linear, and it is often desirable to upper bound the mean delays of some or all of the virtual channels. We have developed a heuristic algorithm for finding optimal channel capacities. Let $l_i = \frac{T(\underline{N})}{C_i} \left[q_i(\underline{N}) - \overline{q}_i(\underline{N}) \right]$. l_i represents the change in the mean network throughput resulting from a small change in the channel capacity C_i. The algorithm, presented in [9], basically performs a search of the parameter space using l_i as the steepest descent criterion. It includes the following two key steps:

1. Select an initial capacity assignment that is feasible.
2. Repeat the following until no more improvement can be made: Select a pair of channel capacities using l_i as metric; deviate capacity from one channel to another to increase network throughput.

The tree convolution algorithm is used to evaluate l_i and network throughputs.

We considered the assignment of channel capacities to a network example with 6 nodes, 9 full-duplex links, and 16 virtual channels in 8 symmetric pairs (details can be found in [9]). Each virtual channel has a window size of two. In order to compare with the square root capacity assignment [4] , we assumed linear cost functions and continuous capacity values. We started with three different sets of feasible capacities and applied the heuristic algorithm. With each set of initial capacities, the heuristic algorithm found a set of final capacities. The three sets of final capacities are almost identical. (See Table 7.) Furthermore, upon algorithm termination, the l_i values, $i=1,2,..,M$ are approximately equal and the final capacities were found to satisfy the optimality condition in (12). We do not know whether this is the global optimum or just a local optimum. We also calculated the capacities given by the square root capacity assignment. These capacities are shown in column 4 of Table 7 and we were surprised to see how different they are from the capacities given by the method herein. For each set of capacities, the network throughputs and mean delays in the bottom rows of Table 7 were calculated using the tree convolution algorithm.

	1	2	3	4*
Initial capacities				
C_1	27200	3200	200	
C_2	200	3200	200	
C_3	200	3200	200	
C_4	200	3200	200	
C_5	200	3200	27200	
C_6	200	3200	200	
C_7	200	3200	200	
C_8	200	3200	200	
C_9	200	3200	200	
Final capacities				
C_1	2857.15	2855.16	2855.4	2901.67
C_2	5503.12	5501.7	5502.3	4244.17
C_3	4354.17	4350.87	4352.2	4244.17
C_4	2819.35	2819.65	2820.1	2901.67
C_5	994.28	997.6	996.2	2901.67
C_6	994.54	996.97	995.7	2901.67
C_7	2819.45	2820.94	2819.36	2901.67
C_8	2819.64	2820.82	2819.3	2901.67
C_9	5638.3	5635.81	5639.5	2901.67
Total throughput	58.54	58.54	58.54	50.2
Average ETE delay	0.15	0.15	0.15	0.18

* the throughput and average end-to-end virtual channel delay are computed for a closed network model using capacities according to the square root assignment.

Table 7. Total throughputs for different initial capacity assignments

The heuristic algorithm can be very easily adapted to perform various network topology optimizations (e.g., adding and deleting communications links). The adaptation has not been done but is something we plan to do in the future. We are also investigating improvements to the speed of the algorithm by using bounds and approximations for estimating throughputs [3] instead of using the tree convolution algorithm at every step.

References

1. F. Baskett, K.M. Chandy, R.R. Muntz, and F. Palacios, "Open, closed and mixed networks of queues with different class of customers," *Comm. ACM*, vol. 22, pp. 248-260, 1975.
2. M. Gerla and L. Kleinrock, "On the topology design of distributed computer networks," *IEEE Trans. on Communication*, vol. 25, pp. 48-60, 1977.
3. Ching-Tarng Hsieh and S.S. Lam, *Two classes of performance bounds for closed queueing networks*, University of Texas at Austin, Austin, Texas, 1985. Technical Report, in preparation
4. L. Kleinrock, *Communication nets: Stochastic message flow and delay*, McGraw-Hill, 1964.
5. S.S Lam and Y.L. Lien, "Congestion control of packet communication networks by Input Buffer Limits --- A simulation study," *IEEE Trans. Computers*, vol. C-30, no. 10, pp. 733-742, Oct. 1981.
6. S.S. Lam, "Store-and-forward buffer requirements in a packet switching network," *IEEE Trans. Comm.*, vol. COM-24, pp. 394-403, 1976.
7. 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, pp. 161-180, 1982.
8. S.S. Lam and Y.L. Lien, "A tree convolution algorithm for the solution of queueing networks," *Comm. ACM*, vol. 26, no. 3, pp. 203-215, 1983.
9. S.S. Lam and Ching-Tarng Hsieh, *Models and algorithms for the design of store-and-forward communication networks*, University of Texas at Austin, Austin, Texas, 1985. Technical Report, in preparation
10. Y.L. Lien, *Modeling and analysis of flow controlled computer communcation networks*, Ph.D. Thesis, Department of Computer Sciences, University of Texas at Austin, 1981.
11. 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.
12. 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.
13. 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.