A CLUSTERING APPROXIMATION TECHNIQUE FOR QUEUEING NETWORK MODELS WITH A LARGE NUMBER OF CHAINS

Edmundo de Souza e Silva
Steven S. Lavenberg
Richard R. Muntz

April 1984
Report No. CSD-840034

Computer Science Department Report No. CSD-840034
1984

A Clustering Approximation Technique for Queueing Network Models with Large Number of Chains

Edmundo Silva
Steven Lavenberg
Richard Muntz

Computer Science Department
School of Engineering and Applied Science
University of California
Los Angeles

This research was sponsored by the Air Force, AFOSR 820304A and National Science Foundation, 8020300.

A CLUSTERING APPROXIMATION TECHNIQUE FOR QUEUEING NETWORK MODELS WITH A LARGE NUMBER OF CHAINS

Abstract

The past few years have witnessed an increasing number of large distributed computer system implementations based on local area networks. In these systems a number of resources (CPU's, file servers, disks, etc) are shared among jobs originating at different sites. Evaluating the performance of such large systems typically requires the solution of a queueing network model with a large number of closed chains, which precludes the use of exact solution techniques. Therefore, it is important to develop accurate and cost effective methods for the approximate analysis of closed queueing networks with many chains. In this paper we present an approach based on the clustering of chains and service centers. The method is applicable to queueing networks with single server fixed rate, infinite server and multiple server service centers. We present the results obtained when the method is used to solve large queueing network models. Extensive comparison of this method with existing approximation techniques indicates that the approach has better accuracy/cost characteristics.

1. Introduction.

Product form queueing networks have been successfully used in computer performance modeling and analysis. In many applications networks with multiple closed and open chains are used. It is not uncommon for the analyst to be faced with a model that possesses several dozen closed chains. For example, in modeling a distributed computer system different chains might be used to model workloads generated at different sites [GOLD83] and in modeling a packet switching network different chains might be used to model traffic in different virtual routes [REIS79]. However, the exact analysis of such closed queueing networks is usually too costly. For example, networks with more than five chains with a few customers per chain (say 10) may be too costly to solve in a medium size computer. For networks with a special sparseness structure, however, exact solution of networks with many closed chains may be feasible [LAM83].

The two major algorithms that have been widely used to solve exactly product form queueing network models are the convolution algorithm [LAVE83], [REIS76] and the mean value analysis algorithm (MVA) [REIS80]. If only mean values are computed and there are no load dependent service centers present in the network, both algorithms have approximately the same cost requirements. Nevertheless, mean value analysis has two significant advantages over the convolution algorithm: it presents no numerical problems; the equations obtained are extremely simple and have a physical interpretation. When load dependent service centers are present and the MVA algorithm is used, the marginal queue length distributions appear in the equations for these centers. Although the simple physical interpretation is still present, the computational complexity grows combinatorially with the number of load dependent service centers.

Recently, a new approach has been proposed [McKE82], [RAMA82] to handle large closed queueing networks. In this approach lower and upper bounds on performance measures are obtained. However, up to the time this paper was written its applicability is restricted to networks in which all chains visit an infinite server service center, all other centers have utilizations that are not too high, e.g., less than .85, and there are no load dependent service centers. The current cost limitations in the existing exact solution

techniques are the main motivation for the development of accurate and cost effective methods for the approximate analysis of closed queueing networks.

De Souza e Silva et al [DeSo83] presented a summary of an approximation technique based on the clustering of chains and service centers. In this paper, we present the details of the approximation, extensions to queueing networks with multiple server service centers, the heuristic procedure for choosing the clusters and examples which include a non-product form queueing network. In section 2 we present a brief overview of some approximate methods for solving multiple closed chain product form networks. Section 3 contains the outline of the chain-server clustering approximation technique for closed product form queueing networks with single server fixed rate (SSFR) service centers and multiple server (MS) service centers. Section 4 presents a detailed description of the algorithm. Section 5 contains the empirical results obtained after comparing this approximation technique with exact MVA. It also includes an application which requires solving a non-product form network. Section 6 concludes the paper.

2. Overview of Some Approximation Methods for Solving Multiple Chain Closed Queueing Networks.

In this section we briefly describe some of the methods used to handle product form networks. A discussion of methods used to approximately solve non-product form networks can be found in [DeSO83].

Many of the approximation methods for solving product form closed queueing networks are based on the MVA equations due to the simple nature of these equations and the physical interpretation they provide. Below we present the equations for closed product form networks for reference.

Let J denote the number of service centers, the first J_1 of which are single server fixed rate (SSFR), centers J_1+1 to J_2 are load dependent and the remainder are infinite server (IS) service centers. We define the following notation:

J	=	total number of service centers.
K	=	total number of chains.
N_k	=	population of chain k.

$\vec{N}$	=	population vector = $(N_1,\ldots,N_K)$.
$\lambda_k(\vec{N})$	=	mean throughput of chain k with population $\vec{N}$.
θ_{kj}	=	visit ratio of a chain k customer to center j.
$L_j(\vec{N})$	=	mean number of customers at center j with population $\vec{N}$.
$L_{kj}(\vec{N})$	=	mean number of customers of chain k at center j with population $\vec{N}$.
$q_j(\vec{N})$	=	mean number of customers in the queue of center j (excluding customers in service) with population $\vec{N}$.
$q_{kj}(\vec{N})$	=	mean number of customers of chain k in the queue of center j (excluding customers in service), with population $\vec{N}$.
$W_{kj}(\vec{N})$	=	mean waiting time (queueing time + service time) of a chain k customer at center j with population $\vec{N}$.
T_{kj}	=	mean service time of a chain k customer at center j.
μ_{kj}	=	service rate of chain k customers at center j.
a_{kj}	=	relative utilization of a chain k customer at center j, $= \theta_{kj}T_{kj}$.
M_j	=	number of servers at center j.
$P_j(i\|\vec{N})$	=	probability that there are i customers at center j given that the population vector is $\vec{N}$.
$\lambda_k^{j-}(\vec{N})$	=	throughput of chain k in a network with center j removed when the population is $\vec{N}$.
$PB_j(\vec{N})$	=	probability that all servers of center j are busy, when the population is $\vec{N}$.
$\vec{e}_k$	=	k-dimensional vector whose k-th element is one and whose other elements are zero.

For $k = 1,\ldots,K$.

$$\lambda_k(\vec{N}) = \frac{N_k}{\sum_{j=1}^{J}\theta_{kj}W_{kj}(\vec{N})} \tag{2.1}$$

$$L_{kj}(\vec{N}) = \lambda_k(\vec{N})\theta_{kj}W_{kj}(\vec{N}) \qquad j = 1,\ldots,J \tag{2.2}$$

$$W_{kj}(\vec{N}) = \begin{cases} T_{kj}[1 + \sum_{i=1}^{K} L_{ij}(\vec{N}-\vec{e}_k)] & j = 1,\ldots,J_1 \quad (2.3a) \\ T_{kj}(\vec{N}) \sum_{i=1}^{N} \frac{i}{\mu_j(i)} P_j(i-1|\vec{N}-\vec{e}_k) & j = 1,\ldots,J_2 \quad (2.3b) \\ T_{kj} & j = J_2+1,\ldots,N \quad (2.3c) \end{cases}$$

$$P_j(i|\vec{N}) = \frac{1}{\mu_j(i)} \sum_k T_{kj}\lambda_k(\vec{N})\theta_{kj}P_j(i-1|\vec{N}-\vec{e}_k) \tag{2.4a}$$

$$P_j(0|\vec{N}) = \frac{\lambda_k(\vec{N})}{\lambda_k^{j-}(\vec{N})} P_j(0|\vec{N}-\vec{e}_k) \tag{2.4b}$$

Equation (2.3b) reduces to:

$$W_{kj}(\vec{N}) = T_{kj}\left[1 + \frac{q_j(\vec{N}-\vec{e}_k)+PB_j(\vec{N}-\vec{e}_k)}{M_j}\right] \qquad j = J_1+1,\ldots,J_2 \tag{2.3d}$$

for multiple server (MS) service centers, where $PB_j(\vec{N})$, the probability that all servers of MS center j are busy (when the population is $\vec{N}$), is calculated from equation (2.4a), which simplifies to:

$$PB_j(\vec{N}) = \begin{cases} \frac{1}{M_j}\sum_{k=1}^{K} \lambda_k(\vec{N})a_{kj}[PB_j(\vec{N}-\vec{e}_k) + P_j(M_j-1|\vec{N}-\vec{e}_k)] & |\vec{N}|>M_j \\ 0 & |\vec{N}|\leq M_j \end{cases} \tag{2.4c}$$

where $|\vec{N}| = N_1 + \cdots + N_K$.

These equations express an exact relationship between the performance parameters and can be used recursively for an exact solution of the network. For networks with SSFR and IS service centers only, the computational requirements to solve the above equations are $O(J\prod_{k=1}^{K}(N_k+1))$. To solve networks with multiple server service centers or general load dependent service centers, the computational requirements increase considerably due to equation (2.4b). In this case, $2^{\#\text{load dependent centers}}$ networks have to be solved, each with some of the load dependent service centers removed from the network [TUCC82].

It is easy to see then that for networks with a few chains, this recursive computation can be prohibitively expensive. Our experience indicates that networks with more than five chains and 10 customers per chain can be impractical to solve in a medium size computer (say, VAX11/750).

A widely used approximation for product form networks with SSFR and IS service centers proposed by Schweitzer [SCHW79] is:

$$L_{ij}(\vec{N}-\vec{e}_k) = \begin{cases} L_{ij}(\vec{N}) & l \neq k \\ \frac{(N_k-1)}{N_k}L_{kj}(\vec{N}) & l = k \end{cases}$$

This approximation removes the recursion in equation (2.3a) and greatly reduces the computational complexity necessary to exactly solve a model. As we will see in section 6 the errors obtained are typically less

than 20% for mean queue lengths and waiting times and less than 10% for throughputs.

Chandy and Neuse [CHAN82] proposed an approximation called Linearizer that is very accurate but more costly. Linearizer is based on refinements to Schweitzer's approximation and was also proposed to approximately solve networks with SSFR and IS service centers only.

Neuse and Chandy [NEUS81] proposed an algorithm to approximately solve product form queueing network models which can handle load dependent centers. The approach is the same as used in the Linearizer algorithm with additional heuristics for load dependent centers. They chose to estimate the probability density of the queue lengths as:

$$P_j(\lfloor L_j(\vec{N}-\vec{e}_k)\rfloor|\vec{N}-\vec{e}_k) = \lfloor L_j(\vec{N}-\vec{e}_k)\rfloor + 1 - L_j(\vec{N}-\vec{e}_k)$$

$$P_j(\lfloor L_j(\vec{N}-\vec{e}_k)\rfloor+1|\vec{N}-\vec{e}_k) = 1 - P_j(\lfloor L_j(\vec{N}-\vec{e}_k)\rfloor|\vec{N}-\vec{e}_k)$$

$$P_j(i|\vec{N}-\vec{e}_k) = 0 \qquad \text{for } i < \lfloor L_j(\vec{N}-\vec{e}_k)\rfloor \text{ or } i > \lfloor L_j(\vec{N}-\vec{e}_k)\rfloor + 1$$

(where $\lfloor L \rfloor$ is the greatest integer less or equal to L).

The above equations indicate that the probability density of queue lengths at a load dependent service center for population $(\vec{N}-\vec{e}_k)$ will be different than zero at most in two points around the average queue lengths. However, there is no theoretical support for this heuristic choice. Furthermore, as we will see in section 6, the use of the Linearizer algorithm can be prohibitive for networks with a large number of chains.

None of the approximate methods described above gives error bounds for the results obtained. In general the accuracy of the method is evaluated empirically by comparing the results of the approximate algorithms obtained after analyzing a large number of networks, with the results obtained by analyzing the same networks with an exact algorithm.

Recently, however, [McKE82], [RAMA82] proposed a method of obtaining bounds on performance measures in product form queueing networks. The approach is roughly summarized as: (1) substitute Euler's integral for a factorial term in the expression for the normalization constant for a network; (2) make asymptotic expansions of the integrals obtained.

The major features of this approach are:

(a) The algorithm is cheap, $O(J\,K^4)$. This assumes at most 4 terms are used in the asymptotic expansion.

(b) Bounds are obtained for the results.

(c) Second moments can be obtained for queue lengths.

However, up to the date this paper was written, the method is restricted to networks in which all chains visit an IS center, all other service centers have utilizations that are not too large, e.g., less than .85 and there are no load dependent centers.

3. A Clustering Approximation Technique for Product Form Queueing Network Models with Single and Multiple Server Service Centers.

As pointed out in [DeSO83], a variety of iterative methods used to solve queueing network models are based on the analysis of a collection of subnetworks. Informally a subnetwork is a detailed representation of a set of resources. The remainder of the subnetwork (the complement of the subnetwork) is typically represented by an approximation of its effect on the subnetwork. Each subnetwork is analyzed in isolation with a simplified representation of the effect of its complement. The parameters which characterize the complement of a subnetwork are function of the performance parameters of the other subnetworks and are determined by iteration. The main reasons to partition a queueing network into subnetworks are: (1) for product form networks, to obtain subnetworks which are much less costly to analyze than the original networks. Hopefully, the sum of the costs of solving each subnetwork times the number of iterations is significantly less than solving the whole network. (2) for non-product form networks, to obtain subnetworks with product form characteristics, or subnetworks that are otherwise simple enough to be analyzed.

[MARI79] R.A. Marie, "An Approximate Analytical Method for General Queueing Networks", IEEE Transactions on Software Engineering SE-5, 530-538, 1979.

[McKE82] J. McKenna and D. Mitra, "Integral Representations and Asymptotic Expansions for Closed Markovian Queueing Networks: Normal Usage", The Bell System Technical Journal 61, no. 5, 661-683, 1982.

[NEUS81] D. Neuse and K.M. Chandy, "SCAT: A Heuristic Algorithm for Queueing Network Model of Computer Systems", Performance Evaluation Review 10, 59-79, 1981.

[RAMA82] K.G. Ramakrishnan and D. Mitra, "An Overview of PANACEA, A Software Package for Analyzing Markovian Queueing Networks", The Bell System Journal 61, no. 10, 2849-2871, 1982.

[REIS76] M. Reiser and H. Kobayashi, "On the Convolution Algorithm for Separable Queueing Networks", International Symposium on Computer Performance, 109-117, 1976.

[REIS79] M. Reiser, "A Queueing Network Analysis of Computer Communication Networks with Window Flow Control", IEEE Transactions on Communications, COM-27, 1199-1209, 1979.

[REIS80] M. Reiser and S.S. Lavenberg, "Mean Value Analysis of Closed Multichain Queueing Networks", Journal of the ACM 27, 313-322, 1980.

[SAUE83] C.M. Sauer, "Corrigendum: Computational Algorithms for State Dependent Queueing Networks", ACM Transactions on Computer Systems 1, no. 4, 369, 1983.

[SCHW79] P. Schweitzer, "Approximate Analysis of Multiclass Closed Networks of Queues", International Conference on Stochastic Control and Optimization, Amsterdam, 1979.

[TUCC82] S. Tucci and E.A. McNair, "Implementation of Mean Value Analysis for Open, Closed and Mixed Queueing Networks", IBM Research Report, RC 9392, 1982.

[ZAHO81] J. Zahorjan and E. Wong, "The Solution of Separable Queueing Network Models Using Mean Value Analysis", Performance Evaluation Review 10, 80-85, 1981.

[ZAHO83] J. Zahorjan, "Workload Representations in Queueing Models of Computer Systems", ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems, 1983.

4) For a given chain and representation of the complement for this chain, the choice of how the parameters of the complement are determined. For example, for a Poisson arrival representation the arrival rate must be determined. Some of the choices are: (1) set the arrival rate equal to the throughput for that chain that was computed on the previous iteration; (2) set the arrival rate equal to the value that yields the same mean population in the subnetwork for that chain as was calculated on the previous iteration; (3) set the arrival rate for a chain in a center according to some characteristics of that center, and/or to yield the same mean population of that chain in that center as calculated on the previous iteration. Note that that the second and third methods above require an additional iterative step while the first does not.

In this section we present a new iterative approximation method that uses an infinite server representation for some chains and a Poisson arrival representation for others. The method is suitable for very large multiple chain networks. This work was motivated by our interest in analyzing queueing network models of large local area networks, i.e., networks with dozens or even hundreds of sites. As an example we mention the model of the LOCUS local area distributed system presented in [GOLD83]. In that model each site was represented by a "central server model" with terminals linked by a representation of the communication channel. The behavior of customers from different sites was represented by associating one or more closed chains with each site. Due to the large number of chains that can result from this model, an approximate solution is required even if the resulting model is product form and the network has only a few sites. (Schweitzer's approximation was used in [GOLD83]). The problem gets considerably more complex if we introduce multiple server service centers in the model (for instance to model a multiprocessor computer site), since the computational requirements to exactly solve this kind of model grows combinatorially with the number of multiple server centers [TUCC82]. To our knowledge there is no widely accepted approximation technique to handle this problem.

In [DeSO83] we proposed a simple iterative method for very large LOCUS type queueing network models. In the method each subnetwork represented the resources at a site. For a particular subnetwork the customer chains fall naturally into two categories. One category consists of the chains that represent

In addition to the simplified nature of the subnetworks, it is desirable that the parameters of the complement be efficiently calculatable from the analysis of the other subnetworks.

There are several ways to represent the effect of the complement of a subnetwork. For example, Marie [MARI79] used queue size dependent arrival rates; Jacobson and Lazowska used IS centers [JACO82] and reduced customer population [JACO83]; de Souza e Silva et al [DeSO83] used Poisson arrivals and IS centers; Zahorjan [ZAHO83] considered a few representations as we mention below. The accuracy of these methods must be evaluated empirically since so far error bounds have not been obtained.

If a queueing network contains multiple chains, the effect of a complement can be represented differently for each chain. For one chain the effect of the complement can be represented as Poisson arrivals, i.e., the chain is open; for another chain, the complement might be represented as an infinite server and for still another chain as a reduced population. The general framework permits numerous possibilities and little is known about how to utilize these possibilities. Figure 1 illustrates a subnetwork and its complement. The following are some of the choices that are available:

1) The choice of subnetworks. Each subnetwork may contain a disjoint set of resources or they may overlap. (see for example, Marie's method [LAVE83], Jacobson and Lazowska [JACO80], de Souza e Silva et al [DeSO83]).

2) The choice of the way in which the final estimates of the performance parameters of the original network are obtained. This choice arises because the same performance parameters, e.g., a chain throughput or a mean queue size, may be represented in more than one subnetwork. Thus, when the iteration is terminated, there may be more than one value for the same parameter. The final estimates could be a function of these values.

3) The choice of the method used to represent the complement of a subnetwork for each chain. Only some of the possibilities have been mentioned above.

into subnetworks according to the contribution of the chains to the total utilization of the centers. We next present our results for product form networks, first for networks with only SSFR and IS centers and then for networks which also include MS centers.

Product Form Networks with SSFR and IS Service Centers.

In this section we consider product form queueing networks consisting of only SSFR and IS service centers. We divide the network into subnetworks which need not be disjoint, but whose union includes all service centers. For each subnetwork we designate a subset of the chains that visit the subnetwork as local and the remaining chains that visit the subnetwork as foreign.

Our approach consists of: (1) preserving the recursion in the MVA equations involving the effect on the mean queue sizes in a subnetwork of removing a local chain customer; (2) approximating the effect on the mean queue sizes in a subnetwork and its complement of removing foreign chain and local chain customers, respectively. We also assume that: (1) the throughputs of all foreign chains are not affected if an arbitrary number of local chain customers are removed from the network; (2) the mean waiting times of a local chain at any service center in the complement is not affected if an arbitrary number of local chain customers are removed from the network. These assumptions are reasonable if local chain customers contribute little to the utilization of service centers in the complement and if foreign chain customers contribute little to the utilization of service centers in the subnetwork.

Let S denote a particular subnetwork, LC(S) the set of local chains for S and FC(S) the set of foreign chains for S. $\vec{N}(S,\vec{n})$ is the population vector where all foreign chains have the full population and the local chains have population $\vec{n}$. Now, using equations (2.2) and (2.3a) for subnetwork S:

$$L_{cj}(\vec{N}(S,\vec{n})) = \lambda_c(\vec{N}(S,\vec{n}))a_{cj}[1 + \sum_{l=1}^{K} L_{lj}(\vec{N}(S,\vec{n})-\vec{e}_c)] \qquad j = 1,\ldots,J_1,\ j \in S,\ c \in FC(S)$$

Applying assumption (1) about the throughputs of foreign chains and Schweitzer's approximation yields:

customers logged on at the site represented by the subnetwork. These customers tend to have much higher resource utilizations at the site than do customers logged on at other sites. The other category consists of the chains that represent customers logged on at other sites. It is convenient to refer to the two categories of chains by calling the former "local" chains and the latter "foreign" chains. In the approximation the effect of the complement of the subnetwork is represented differently for the two categories of chains. For local chains the complement is represented as an infinite server while for foreign chains the complement is represented by Poisson arrivals.

For illustrative purposes, let us consider a LOCUS distributed system consisting of only two sites (referred as site A and B, respectively) and let us ignore the communication channel. A queueing model of this system is shown in Figure 2a. In this example the behavior of customers from site A (B) is represented by chain A (B). To apply our approximation technique, we divide the model into two subnetworks, labeled A and B. Subnetwork A (B) is formed by all the resources from site A (B). The local chain of subnetwork A (B) is chain A (B) and the foreign chain is B (A). Therefore, the queueing model of subnetwork A (B) will be formed by all the resources from site A (B), one closed chain representing local chain A (B), one IS service center visited by local chain A (B) which represents the complement of local chain A (B), and finally a set of open chains, each one arriving to a center of subnetwork A (B) and visiting only this center. Each open chain in this set represents the effect of foreign chain B (A) on a particular center in subnetwork A (B). Subnetwork A is shown in Figure 2b.

We have extended this method to be a general method for large multiple chain product form queueing networks. In the extension we still have two categories of chains for each subnetwork and corresponding representations for the complement. Although the adjectives "local" and "foreign" have lost some of their original meaning, it is convenient to retain the terminology. We wish to choose the subnetworks and their local and foreign chains so that it remains true that the set of local chains with respect to a subnetwork contributes more to the total utilization at the service centers in the subnetwork than those chains that are foreign with respect to the subnetwork. In other words, chains and servers are clustered

$$\theta_{kj}W_{kj}(\vec{N}) = a_{kj}\left[1 + L_j(\vec{N}) - \frac{L_{kj}(\vec{N})}{N_k}\right] \qquad j = 1,\ldots,J_1,\ j \notin S,\ k \in LC(S)$$

Substituting equation (3.1b) into the above equation yields

$$\theta_{kj}W_{kj}(\vec{N}) = \frac{a_{kj}}{1 + \lambda_k(\vec{N})a_{kj}/N_k}[1 + L_j(\vec{N})]$$

Therefore,

$$\lambda_k(\vec{N}(S,\vec{n})) = \frac{n_k}{\sum_{j \in S} \theta_{kj}W_{kj}(\vec{N}(S,\vec{n})) + D_k} \qquad k \in LC(S) \tag{3.4}$$

where

$$D_k = \sum_{\substack{j \notin S \\ j = 1,\ldots,J_1}} \frac{a_{kj}}{1 + \lambda_k(\vec{N})a_{kj}/N_k}[1 + L_j(\vec{N})] + \sum_{\substack{j \notin S \\ j = J_2+1,\ldots,J}} a_{kj} \qquad k \in LC(S) \tag{3.5}$$

Equations (3.2) - (3.5) and for $j \in S$ equations (2.3a), (2.3c) constitute the set of equations for subnetwork S that we wanted to obtain. When comparing these equations with the exact MVA equations for mixed networks [ZAHO81] it is easy to see that in terms of mean values (1) a foreign chain l effects service center $j \in S$ as if it were an open chain, i.e., Poisson arrivals, with arrival rate to center j given by

$$\lambda_l(\vec{N})\theta_{lj}/[1 + (\lambda_l(\vec{N})a_{lj}/N_l)]$$

and (2) a local chain k customer that leaves the subnetwork "sees" the complement as an infinite server with mean delay D_k. Thus when we solve a particular subnetwork we reduce the network to that shown in Figure 3. In that Figure all foreign chains are represented by open chains and the delay when a local chain customer leaves the subnetwork is represented by an infinite server. Note that the open chain arrival rate for foreign chain l at service center j in the subnetwork is less than the throughput $\lambda_l(\vec{N})\theta_{lj}$ of the corresponding closed foreign chain. As proved in [ZAHO83] if in a single chain closed network the closed chain is replaced by an open chain leaving the same throughput then the mean queue sizes increase, possibly substantially. (This result also appears to hold for multiple chain networks). The reduced arrival rate

$$L_{cj}(\vec{N}(S,\vec{n})) = \lambda_c(\vec{N})a_{cj}\left[1 + L_j(\vec{N}(S,\vec{n})) - \frac{L_{cj}(\vec{N}(S,\vec{n}))}{N_c}\right]$$

from which

$$L_{cj}(\vec{N}(S,\vec{n})) = \frac{\lambda_c(\vec{N})a_{cj}}{1 + \lambda_c(\vec{N})a_{cj} / N_c}[1 + L_j(\vec{N}(S,\vec{n}))] \qquad j = 1,...,J_1, j \in S, c \in FC(S) \tag{3.1a}$$

Similarly we can obtain:

$$L_{kj}(\vec{N}) = \frac{\lambda_k(\vec{N})a_{kj}}{1 + \lambda_k(\vec{N})a_{kj} / N_k}[1 + L_j(\vec{N})] \qquad j = 1,...,J_1, j \notin S, k \in LC(S) \tag{3.1b}$$

We write

$$L_j(\vec{N}(S,\vec{n})) = \sum_{l \in FC(S)} L_{lj}(\vec{N}(S,\vec{n})) + \sum_{k \in LC(S)} L_{kj}(\vec{N}(S,\vec{n}))$$

Substituting (3.1a) into the first sum and (2.2) into the second sum yields

$$L_j(\vec{N}(S,\vec{n})) = U_j[1 + L_j(\vec{N}(S,\vec{n}))] + \sum_{k \in LC(S)} \lambda_k(\vec{N}(S,\vec{n}))\theta_{kj}W_{kj}(\vec{N}(S,\vec{n}))$$

from which it follows that

$$L_j(\vec{N}(S,\vec{n})) = \frac{U_j + \sum_{l \in LC(S)} \lambda_l(\vec{N}(S,\vec{n}))\theta_{lj}W_{lj}(\vec{N}(S,\vec{n}))}{1 - U_j} \qquad j = 1,...,J_1, j \in S \tag{3.2}$$

where:

$$U_j = \sum_{l \in FC(S)} \frac{\lambda_l(\vec{N})a_{lj}}{1 + \lambda_l(\vec{N})a_{lj} / N_l} \qquad j = 1,...,J_1, j \in S \tag{3.3}$$

Using assumption (2) about the mean waiting times of a local chain in the complement and equation (2.1) yields

$$\lambda_k(\vec{N}(S,\vec{n})) = \frac{n_k}{\sum_{j \in S} \theta_{kj}W_{kj}(\vec{N}(S,\vec{n})) + \sum_{\substack{j \notin S \\ j = 1,...,J_1}} \theta_{kj}W_{kj}(\vec{N}) + \sum_{\substack{j \notin S \\ j = J_2+1,...,J}} a_{kj}} \qquad k \in LC(S)$$

Using Schweitzer's approximation and equation (2.3a) yields

Similarly, we obtain the expression for the throughput of local chains:

$$\lambda_k(\vec{N}(S,\vec{n})) = \frac{n_k}{\sum_{j \in S} \theta_{kj} W_{kj}(\vec{N}(S,\vec{n})) + D_k + DM_k} \qquad k \in LC(S) \tag{3.10}$$

where

$$DM_k = \sum_{j=j_1+1,\ldots,j_2} \frac{a_{kj}}{1 + \lambda_k(\vec{N}) a_{kj}/N_k M_j} \left[\frac{q_j(\vec{N}) + PB_j(\vec{N})}{M_j} \right] \tag{3.11}$$

As in the previous section, equations (3.8) - (3.11) have the same form as the exact equations for a mixed network with MS service centers. Therefore, the same interpretation holds, i.e., a foreign chain l effects multiple server center j as an open chain with arrival rate

$$\lambda_l(\vec{N})\theta_{lj} / [1 + (\lambda_l(\vec{N}) a_{lj}/M_j N_l)]$$

and a local chain customer that leaves the subnetwork "sees" the complement as an infinite server with mean delay $D_k + DM_k$.

In order to complete the set of equations for subnetwork S we need an expression for $PB_j(\vec{N}(S,\vec{n}))$, $j \in S$, which is the probability that all servers are busy at MS center $j \in S$ considering all chains in the network. Rather than derive an approximate expression for these probabilities we will solve the above mixed network. The solution is obtained by recursively solving a corresponding closed network with the degraded service rates [LAVE83] [SAUE83]:

$$\mu_j'(n) = \mu_j(n) \frac{\alpha_j(n-1)}{\alpha_j(n)}$$

where

$$\alpha_j(n) = \frac{M_j^{M_j-n-1}}{\prod_{h=n+1}^{M_j-1} \mu_j(h)} \times \frac{1}{(1 - UM_j/M_j)^{n+1}} + \sum_{i=0}^{M_j-2} \frac{(i+n)!}{i!\, n!} UM_j^i \left\{ \frac{1}{\prod_{h=n+1}^{n+i} \mu_j(h)} - \frac{1}{\prod_{h=n+1}^{M_j-1} \mu_j(h)\, M_j^{n-M_j+i+1}} \right\}$$

we use tends to compensate for the larger mean queue sizes that result from replacing foreign closed chains by open chains. Later we will present results of extensive empirical tests that demonstrate the accuracy that is achieved.

Product Form Networks with Multiple Server (MS) Service Centers.

For networks with MS service centers, we apply the same assumptions used in the previous section. The development is similar, but for MS centers equation (2.3d) is used instead of equation (2.3a). We also assume that for MS service centers:

$$q_j(\vec{N}(S,\vec{n}-\vec{e}_l)) + PB_j(\vec{N}(S,\vec{n}-\vec{e}_l)) = q_j(\vec{N}(S,\vec{n})) + PB_j(\vec{N}(S,\vec{n})) - \frac{q_{lj}(\vec{N}(S,\vec{n}))}{N_l} \qquad j \in S,\ l \in FC(S) \tag{3.6}$$

This assumption is similar to Schweitzer's approximation, since $q_{kj}(\vec{N}) = L_{kj}(\vec{N}) - \lambda_k(\vec{N})a_{kj}$, and was found to work well in our empirical tests.

Using equations (2.2) (2.3d) and (3.6) and applying assumption (1) about foreign chain throughputs we obtain:

$$q_{cj}(\vec{N}(S,\vec{n})) = \frac{\lambda_c(\vec{N})a_{cj}}{1 + \lambda_c(\vec{N})a_{cj} / (N_c M_j)} \left[\frac{q_j(\vec{N}(S,\vec{n})) + PB_j(\vec{N}(S,\vec{n}))}{M_j} \right] \qquad j \in S,\ c \in FC(S) \tag{3.7}$$

which is similar to equation (3.1a).

Following the same steps leading to equations (3.2) and (3.3) we obtain:

$$q_j(\vec{N}(S,\vec{n})) = \frac{UM_j PB_j(\vec{N}(S,\vec{n}))/M_j + \sum_{l \in LC(S)} \lambda_l(\vec{N}(S,\vec{n}))\theta_{lj}[W_{lj}(\vec{N}(S,\vec{n})) - T_{lj}]}{1 - UM_j/M_j} \quad j = J_1+1,\ldots,J_2,\ j \in S \tag{3.8}$$

where:

$$UM_j = \sum_{l \in FC(S)} \frac{\lambda_l(\vec{N})a_{lj}}{1 + \lambda_l(\vec{N})a_{lj} / M_j N_l} \qquad j = J_1+1,\ldots,J_2,\ j \in S \tag{3.9}$$

$$\frac{L_{lj}^{(l)} - L_{lj}^{(l-1)}}{N_l} < threshold \qquad j = 1,...,J, \; l = 1,...,K \tag{a1}$$

(where the superscript indicates the number of the current iteration.)

As initial estimates in step 1 we used:

$$L_{kj}(\vec{N}) = \frac{a_{kj}N_k}{\sum_{i=1}^{J} a_{ki}} \qquad j = 1,...,J; \qquad k = 1,...,K \tag{a2}$$

$$\lambda_k(\vec{N}) = \frac{N_k}{\sum_{j=1}^{J_1} a_{kj}[1 + L_j(\vec{N}) - L_{kj}(\vec{N})/N_k] + \sum_{j=J_1+1}^{J_2} a_{kj}[1 + (L_j(\vec{N}) - L_{kj}(\vec{N})/N_k)/M_j] + \sum_{j=J_2+1}^{J} a_{kj}} \quad k = 1,...,K \tag{a3}$$

$$P_j(i|\vec{N}) = \begin{cases} 1 \text{ if } i = \lfloor \sum_{l \in LC(S)} L_{lj}(\vec{N}) \rfloor & j \in S, \quad j = J_1+1,...,J_2 \\ 0 \; otherwise \end{cases} \tag{a4}$$

Instead of using exact MVA to recursively solve a subnetwork, we can use any existing approximation for product form networks. For example, Schweitzer's and the Linearizer approximations can be used when there are no MS service centers in the subnetwork. It is easy to show that if Schweitzer's approximation is used in all subnetworks, the results are identical to using Schweitzer's approximation on the entire network.

The cost of the algorithm depends on the solution technique used to solve each subnetwork and the way the subnetworks are chosen. (Note that different solution techniques can be used to solve different subnetworks.) The cost per iteration is equal to the sum of the costs to solve each subnetwork. For example, if MVA is used in all subnetworks and there are no MS centers, the cost per iteration is $O(\sum_S [J_S \prod_{k \in LC(S)} (N_k+1)])$, where J_S is the total number of centers of subnetwork S. The total number of iterations is small, typically around 10 iterations for the tested cases. The memory requirements of the algorithm are on the order of the memory requirements of the subnetwork which used the largest amount of

$$\text{for } n \geq M_j - 1 \tag{3.12a}$$

$$\alpha_j(n) = \frac{1}{(1 - UM_j/M_j)^{n+1}} \qquad \text{for } n < M_j - 1 \tag{3.12b}$$

and

$$\mu_j(n) = \begin{cases} n & \text{if } n \leq M_j \\ M_j & \textit{otherwise} \end{cases}$$

4. The Algorithm.

The following algorithm can be used to iteratively solve the equations obtained for all subnetworks. It is similar to the algorithm described in [DeSO83] with additional equations for multiple server centers.

1) Obtain an initial estimate of the throughput of all chains and the mean queue sizes of all service centers. Obtain an initial estimate for the equilibrium marginal queue size probabilities of all multiple server centers.

2) Loop through the subnetworks and for each one solve the corresponding mixed queueing network recursively. New estimates for $\{D_k\}$, $\{DM_k\}$, $\{U_j\}$, $\{UM_j\}$ and $\{\alpha_j(n)\}$ are obtained just before calculating the performance measures for a subnetwork, using (3.5), (3.11), (3.3), (3.9), (3.12). (Note that for a particular subnetwork this recursion is only over the population of local chains. The populations of all foreign chains are fixed. Furthermore the only MS service centers involved are those belonging to this subnetwork).

3) Compare the current estimates of mean queue lengths with the previous estimates using (a1) below, and terminate if this result is less than a specified threshold. Otherwise go to 2).

In the tests we will report on next, we did not make use of the automated choice of subnetworks in the first series of experiments. Instead, we based our choice on the estimated utilizations. We then repeated all tests with the subnetworks chosen automatically. For the automated clustering we tried to maintain the same total cost as the manual choice. In the majority of the cases the automated choice was different than the manual choice of subnetworks. However, in some cases the automated choice produced slightly better results and in other cases slightly worse results, for approximately the same cost. Overall, the automated choice produced results with accuracy comparable to the manual choice.

It is important to note that the cost of clustering was negligible compared to the actual computation of the performance measures in all cases.

5. Empirical Results

We have tested our method on more than 180 queueing networks consisting of randomly selected networks, networks of the type proposed to model LOCUS [GOLD83] (e.g. see Figure 6) and networks of the type proposed to model packet switching networks [REIS79] (e.g. see Figure 7). We divided our tests into two sets of experiments with identical numbers of networks.

The first set consists of networks with SSFR and IS service centers only. We varied the number of chains from 3 to 9 and the total number of customers from 6 to 50. In the majority of the cases at least one service center had utilization greater than .8. For the LOCUS type of models we varied the mean think time from 1 to 20 seconds; the mean local CPU service time from 1 to 100 msec and the mean foreign CPU service time from 1.5 to 150 msec. The average number of local CPU-disk cycles was varied from 1 to 10; the average disk service time was varied from 35 msec to 150 msec; the probability of making a request to a foreign site (after a CPU service) was varied from .1 to .9. Message sizes varied from 1K bytes to 10K bytes and were transmitted over a 1 to 10M bit channel. For the packet switching network models, the packet size varied from 250 to 2K bytes, and channel speeds from 20 kbps to 100 kbps (both half and full duplex). The window size varied from 1 to 15.

memory. For example, if MVA is used in all subnetworks, the memory requirements are $O(\max [J_S \prod_{k \in LC(S)} (N_k+1)])$.

The Choice of Subnetworks.

There are many ways to choose the subnetworks and the local chains for each subnetwork. For each choice the final results and the total cost of each iteration will differ. In general, larger subnetworks with more local chains yield more accurate but more costly results. As we have mentioned our approximation should be reasonable if the main contributions to the utilization of service centers in a subnetwork are from local chains. However, it was determined after extensive tests that the approximation can be used even if the foreign chains are responsible for up to 50% of the total utilization of a service center. Furthermore, this constraint can be relaxed for service centers with low to moderate total estimated utilization, say less than .6.

We have automated the choice of the subnetworks. The choice is based on the estimated utilization of each chain at a service center, obtained from (a2) and (a3). A detailed description of the algorithm is given in the appendix. The user is prompted for the maximum total cost (defined in the appendix) allowed per subnetwork and an initial number of local chains (INC) per subnetwork. This number will be used as the starting point for the search procedure, i.e., the algorithm starts searching for subnetworks with INC local chains. This number can be decreased or increased automatically to adjust the total cost of a subnetwork up to the maximum allowed cost. The objective of each step of the algorithm is to find a subnetwork that contains at least one service center with moderate to high estimated utilization, which does not violate the cost constraint. (Note that a service center can belong to a subnetwork if its local utilization is higher than .5). The algorithm takes into account that overlap can occur among subnetworks and that there can be subnetworks with no local chains or no service centers (*).

(*) Chains that do not belong to any set of local chains are clustered in a subnetwork with no service centers. These chains will be the local chains of an empty subnetwork. Therefore, they visit only an IS center representing their complement. This case results from networks which contain chains that do not contribute much to the total utilization of any center. From the set of equations obtained above we see that the effect of these chains is approximated by a reduction in the capacity of all the centers they visit.

ing times, throughputs) were less than two percent compared with 26.8% (43.8%, 26.9%) for SCH and 99.6% (99.2%, 100%) for LIN.

From Table 1 and Figure 4 it is clear that Linearizer applied to the entire network is the most accurate approximation. However, as we will illustrate in two large LOCUS examples below, it is by far the most expensive of the approximations and it cost becomes prohibitive for networks with many chains. Schweitzer's approximation applied to the entire network is less costly, but it can yield large errors. It was not uncommon to get maximum errors around 20%. Our subnetwork approach provides favorable accuracy/cost characteristics, particularly as the number of chains increases.

Table 2 and Figure 5 present results for the second set of experiments. We note that the results are slightly worse than the results obtained for the first set of experiments. This can be explained by the fact that, for this second set, we chose several networks that produced some of the least accurate results in the first set of experiments.

We next present three networks with many chains to illustrate the potential of the approximation technique we have described. Since our initial research goal was motivated by the LOCUS system, we chose as the first two networks LOCUS types of models, as illustrated in Figure 6.

The first network is a model of LOCUS with 10 computer sites connected by a slotted ring communication channel modeled as proposed in [BUX81]. Each site has 1 CPU (modeled as processor sharing queue), 2 disks (modeled as FCFS queues) and only one type of job (thus only 1 chain per site is needed). The slotted ring is modeled by a processor sharing service center and a closed chain visiting only this center. This closed chain has only one job and models the time periods of one cycle during which the slot is empty (see [BUX81]). Therefore we have a network with 11 chains and 41 service centers. For each site we randomly chose from the following parameter values:

- mean think time: 2 to 15 sec.
- mean local CPU service time: 50 to 100 msec.

The second set of experiments consisted of networks with at least one MS service center. More than half of the networks had the visit ratio of their chains and service times of the centers randomly selected. The number of chains varied from 3 to 6, the maximum total number of customers was 50, the number of MS service centers varied from 2 to 5 and the maximum number of servers varied from 2 to 15. For the rest of the networks we chose, from the first set of tests, several networks that produced some of the least accurate results and replaced one or more of the centers by a multiple server center. For these networks, the number of chains varied from 4 to 8, the number of MS service centers varied from 1 to 3, the number of servers varied from 2 to 3 and the maximum total number of customers was 50. It is worthwhile to comment that in the second set of tests the exact solution using MVA was very costly and several of the tests took a few hours of CPU time in the VAX11/750, even for networks with a small number of chains. Below we present the results of our experiments.

In the first set of experiments (i.e., for networks with SSFR and IS service centers only) each queueing network model was solved exactly using MVA and approximately using exact MVA to solve each subnetwork (MVASUB) and approximately using Linearizer to solve each subnetwork (LINSUB). For comparison with existing approximations we also solved the entire network using Schweitzer's approximation (SCH) and Linearizer approximation (LIN). In the second set of experiments (i.e., for networks with at least one MS service center) each queueing network model was solved exactly using MVA and approximately using exact MVA to solve each subnetwork.

Table 1 and Figure 4 summarize the errors obtained for the first set of experiments. Table 1 gives the average absolute value of the percent errors and the average of the maximum for each network. Separate results are given for mean queue sizes, mean waiting times and throughputs in the columns headed L, W and λ respectively. Figure 4 shows densities of the absolute value of percent errors for L, W, and λ. We found virtually the same errors were obtained when we used exact MVA or Linearizer to solve each subnetwork. Since LINSUB is less costly than MVASUB we only plot the results for LINSUB in Figure 4. We found that for LINSUB 73.7% (74.5%, 88.7%) of the errors for mean queue sizes (mean wait-

for network 2: 171 sec, 123 sec and 1 hour and 15 min respectively (*). In Table 3 we give the absolute value of the percent differences between the LIN solutions and the LINSUB and SCH solutions for the two networks. Our LINSUB method provides close to the accuracy of Linearizer applied to the entire network at close to the cost of Schweitzer's approximation applied to the entire network. It appears to be the method of choice when Linearizer is too expensive to apply to the entire network.

The third network is a model of small packet switching network with window flow control as illustrated in Figure 7. There are five virtual circuits, each modeled as a closed chain as proposed in [REIS79]. All the links are half-duplex. Centers 1 to 5 represent groups of links 1 to 5 respectively and centers 6 to 10 represent the sources of virtual routes 1 to 5 respectively. In this example there are two identical links connecting node 2 to node 3 and three identical links connecting node 4 to node 5. We assume that the switching node 2 (4) has only one internal queue to route packets to node 3 (5) and chooses the first available free link for that. Therefore, we model the multiple links 2 and 4 as MS service centers with 2 and 3 servers respectively.

Table 4 presents the parameter values for this network. The automated clustering algorithm divided the network into four non-overlapping subnetworks as shown in Table 5. Note that subnetwork 4 has no local chains which means that the complement for this subnetwork is represented only by Poisson arrivals. Table 6 gives the absolute percent errors for the solution obtained when MVASUB was used in comparison to MVA used for the whole network. The CPU times to solve this network in a VAX11/750 using MVASUB and MVA where 12.7 sec and 2.86 hours, respectively.

(*) In [DeSO83] we reported higher CPU times to solve these examples with LINSUB and LIN. The reason was that we implemented the Linearizer algorithm as suggested in the original paper [CHAN82], i.e., with the core algorithm costing $O(JK^2)$. However, it can be shown that the core algorithm necessary for Linearizer can be implemented with a cost of only $O(JK)$. These new CPU times reflect the new implementation.

- mean foreign CPU service time: 7% to 20% greater than local service time.
- average disk service times: 50 to 80 msec.
- average number of local CPU-disk cycles: 8 - 10.
- probability of making a request to a foreign site after a CPU-disk cycle: 0.1 to 0.4.
- message size: 1000 bytes.
- channel speed: 10 Mbps.
- jobs from one site may be restricted to run at only some of the sites.
- number of jobs per chain: 6 to 12 (except for the chain of the slotted ring model which has 1 job).

The second network is a much larger one and reflects the large number of sites and types of jobs that would be found in a LOCUS network. The model is for a 16 site computer network, again connected by a slotted ring. Each site has 2 or 3 different types of jobs (thus 2 or 3 chains per site). Each site has 1 CPU and 2 disks. The total number of chains and service centers is 41 and 65 respectively.

We randomly chose from the following parameter values for each site:

- mean think time: 1 to 15 sec.
- mean local CPU service time: 10 to 120 msec.
- mean foreign CPU service time: 10 to 75% greater than local CPU service time.
- average disk service time: 40 to 90 msec.
- average number of local CPU-disk cycles: 2 to 10.
- probability of making a request to a foreign site after a CPU-disk cycle: .1 to .8.
- message size: 1000 bytes.
- channel speed: 10 Mbps.
- jobs from one site may visit a subset of sites or all the other sites.
- number of jobs per chain: 1 to 10.

The networks are too large to solve exactly using MVA and we only solved them approximately using LIN, SCH and LINSUB. When LINSUB was used, the first network was manually divided into 11 subnetworks, 10 of them representing the sites and 1 representing the communication channel. Similarly, the second network was manually divided into 17 subnetworks, 16 of them representing the sites and 1 representing the communication channel. The CPU times to approximately solve the networks on a VAX11/750 using LINSUB, SCH and LIN were for network 1: 32 sec, 23 sec and 164 sec respectively and

Figure 8 shows the average message delay in the Ethernet channel for a network with five sites when the population of each site increases from 1 to 50, the message size is 2000 bits and the maximum propagation delay (MPD) is .1 msec. Also shown are 90% confidence intervals obtained from a simulation of this model. Figures 9 and 10 show the average response time when the number of sites increases (*), when Ethernet and FCFS models are used for the channel. In Figure 9 the message size is 4000 bits and the maximum propagation delay is .1 msec. In Figure 10 the message size is 500 bits and the maximum propagation delay is .05 msec. In both cases the population of each site is 10. Observe that a FCFS service center for modeling the Ethernet channel provides a good approximation only when the channel is not near saturation. The sharp knee in the response time curves indicates that a threshold model [KLEI76] would be a good approximation for these systems.

6. Conclusions.

We presented an iterative approximation technique applicable to queueing network models with a large number of closed chains. The method applies to product form networks with single server fixed rate service centers, infinite server service centers and multiple server service centers. The approach can be easily extended to support the class of queue dependent servers described in [HEFF82]. Extensive empirical results indicate that this method has good accuracy/cost characteristics when compared to existing methods, particularly for networks with many chains. The approximation involves partitioning a queueing network into subnetworks. A critical part of applying the algorithm is the "clustering" of chains and service centers to form the subnetworks. An efficient and effective heuristic was described to automatically perform the clustering based on the computational cost that is specified. The approximation also provides the flexibility to trade off increased cost for increased accuracy by choosing larger subnetworks with more local chains.

(*) We didn't simulate the networks for these two Figures due to the cost of simulating such large networks.

An Application Involving a Non-product Form Network.

Many local computer networks uses a CSMA-CD protocol to access the bus connecting the different sites in the network. In the examples above, we assumed that the communication channel was a slotted ring and used Bux's model for the ring. Therefore the whole network model was product form. In [GOLD83] a FCFS single server center was used to model the effect of the Ethernet communication channel in the LOCUS network. We now address the problem of introducing a more detailed representation of the Ethernet channel in a LOCUS type model.

In the first two examples presented above, we applied the approximation developed in this paper to solve a model of LOCUS with several sites. The network was divided into several subnetworks, one for each site and one representing the communication channel. For the communication channel, the effect of customers in the rest of the network was represented as Poisson arrivals. Furthermore, the effect of the communication channel on the different sites was represented as an infinite server delay. Instead of using a FCFS center for the channel or Bux's model, we chose to use the analytical results obtained by Lam [LAM80] for a CSMA-CD protocol, with the minor heuristic modification introduced in [BUX81] for a nonslotted channel. Lam obtained a formula for the average delay for a packet in a CSMA-CD channel, assuming Poisson arrivals. We used the arrival rates implied by equation (3.3) as input to Lam's formula. The mean delay obtained was used in (3.4).

Figures 8, 9 and 10 show the results obtained for a model of LOCUS with identical sites, when a FCFS center was used for modeling the channel and when Lam's average delay formula was used. The parameters for each site are:

- mean think time: 4 sec.
- mean CPU service time: 8 msec.
- average disk service time: 10 msec.
- average number of local CPU-disk cycles: 8.
- probability of making a request to a foreign site after a CPU-disk cycle: .3.
- channel speed: 1 Mbps.
- jobs from one site may choose any foreign site to run, with equal probability.

Appendix.

We developed a heuristic search algorithm to divide a network into subnetworks and choose the local chains for each subnetwork. The goal of the algorithm can be described as follows:

Given:

1. The estimated utilizations of each chain at each service center.

2. The maximum cost per iteration of a subnetwork.

Find:

A minimum covering set of subnetworks (the subnetworks need not be disjoint), and for each subnetwork a subset of the chains (the local chains).

So that:

1. For each subnetwork, the sum of utilizations of local chains at each service center of that subnetwork is greater than 50% of the total utilization of that center.

2. The cost of each subnetwork is less than the given cost.

3. For a subset of chains, a subnetwork is formed by adding all possible service centers so that the constraints 1 and 2 above are not violated.

4. Constraint 1 may be relaxed if the total utilization of a service center is less than .6.

Based on the requirements above we developed a heuristic search algorithm to divide a general network into subnetworks (*). In addition to giving the maximum cost per iteration of a subnetwork the user is prompted to provide a "starting point" for the search, in terms of an initial number of local chains per subnetwork. The cost function of a subnetwork used by the algorithm, when MVA is used to solve a subnetwork, is $\prod_{l \in LC(S)} (N_l + 1) 2^{\#\text{ MS service centers}}$, since this function provides an estimate of the computational costs to solve each subnetwork exactly using MVA. Other functions may be used as appropriate.

We classify the centers in the network into critical and non-critical. Critical service centers are SSFR or MS service centers that have total estimated utilization greater than or equal to .6. We also define two functions:
1) "Usage value" (UV) of a chain:

(*) However, any other algorithm satisfying the above requirements should work.

The physical interpretation obtained with this technique provides a way of using it in a broader class of problems. In particular, we described a simple heuristic application for a non-product form network. The method is extendible to other classes of non-product form networks and the results will be presented in a future report.

```
FIND_A_SUBNETWORK_S()
{
 - Take PRCS chains from NAG which greater UV_k(NAG)
   and store in LC(S).
 - FIND_CENTERS();

 while (# of critical centers in S not yet present in another subnetwork == 0) {
   if (# of chains in LC(S) < PRCS) {
        - Take a new set of (PRCS − # chains in LC(S)) chains
          from the original network with grater UV_k(NAG)
          value and store in LC(S);
        - FIND_CENTERS();
   }
   else {
        if (more than 1 chain in LC(S) have not yet been replaced && REPLACE_CHAIN() == YES) {
          - FIND_CENTERS();
        }
        else {
          if (PRCS has not yet been decremented) {
            - increment PRCS;
            - return;
          }
          else {
            - "Cost is too low. Can't cluster";
            - return;
          }
        }
   }
 }

 if (cost(S) ≤ given cost) {
   - S is a new subnetwork;
   - return;
 }
 else {
   - REMOVE_UN_CHAINS();
   if (cost(S) > given cost) {
        - Remove MS service centers in S (one by one) until
          cost is satisfied or at least 1 critical MS service
          center remains (if there is any).
   }
   else S is a new subnetwork; return;
   if (cost(S) > given cost) {
        if (more than 1 chain in LC(S) has not yet been replaced) {
          - Replace a chain in LC(S) with smallest UV_k(S)
            by a chain in the original network (which does
            not belong to LC(S)) with the highest UV_k(S)
            AND less number of customers than the chain to
            be replaced. The replacement takes place only if
            there is at least 1 non-critical center (in the
            new formed subnetwork S) not yet assigned
            to any other subnetwork.
        }
   }
   else S is a new subnetwork; return;
   if (cost(S) > given cost) {
        if ( PRCS has not yet been incremented) {
          - Decrement PRCS;
          - return;
        }
        else {
          - "Cost is too low. Can't cluster";
          - return;
        }
   }
   else S is a new subnetwork; return;
 }

}
```

$$UV_k(S) = \sum_{j \in C(S)} U_j \frac{U_{kj}}{(1 - U_{kj})}$$

2) "Local usage" (LU) of a center:

$$LU_j(S) = U_j \frac{\sum_{k \in LC(S)} U_{kj}}{1 - \sum_{k \in LC(S)} U_{kj}}$$

where:

U_j = total estimated utilization of center j.
U_{kj} = estimated utilization of chain k at center j.
C(S) = critical centers in subnetwork S.

Note that $UV_k(S)$ is a measure of the contribution of chain k to the utilization of critical centers in subnetwork S, and $LU_j(S)$ is a measure of the contribution of the local chains of subnetwork S to the utilization of a center $j \in S$.

We make use of two main subsets during the algorithm:

NAG = subset of all centers not yet assigned to any subnetwork and all chains not yet assigned to be local to any subnetwork.
AG = subset of all chains and centers already assigned.

The the algorithm searches for a subnetwork so that:

1. At least one new local chain is present.

2. At least one critical center not yet used is present.

The Algorithm.

Initially NAG is set to contain all centers and chains and PRCS (present number of local chains per subnetwork) is set to the initial number of local chains per subnetwork.

```
MAIN()
{
 while ( # critical centers in NAG is > 0 ) {
    - FIND_A_SUBNETWORK_S();
    if (S is found) {
         - REMOVE_UN_CHAINS();
         - update NAG and AG;
    }
 }
 - Remove any subnetwork covered by other.
 - Form a subnetwork containing no centers and
   whose local chains are the ones in NAG.
 - Form a subnetwork containing only centers in NAG.
}
```

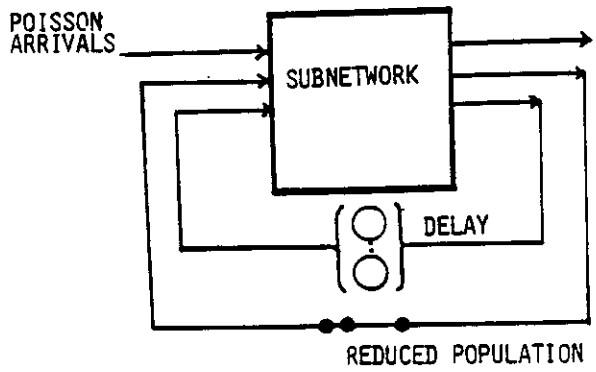

Figure 1.

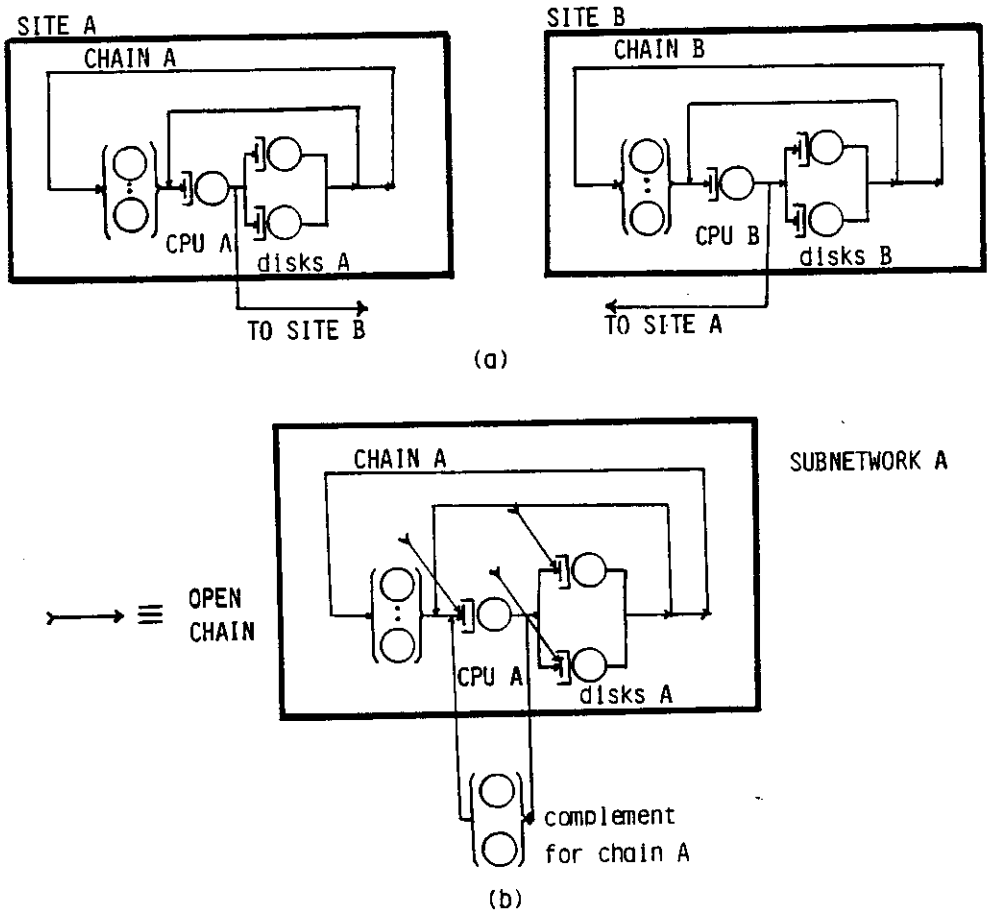

Figure 2. A distributed system with 2 sites.

FIND_CENTERS()
{
- Find all centers from the original network so that constraint 1 above is satisfied for the set of local chains of S.
  Store these centers in S.
}

REMOVE_UN_CHAINS()
{
- Remove any chain in LC(S) which is "unnecessary". A unnecessary chain l is the one that, if removed, do not alter the number of critical centers in S, AND

$$\sum_{j \in C(S)} \frac{LU_j(S) - LU_j(S \text{ without local chain } l)}{LU_j(S)} < \epsilon$$

  This last condition avoids the removal of local chains that contribute significantly to the total utilization of centers in S.
}

REPLACE_CHAIN()
{
- Replace a local chain of S with smallest $UV_k(NAG)$ value by a chain in the original network with greater $UV_k(NAG)$ value (if possible);
 if (replacement is done)
   return(YES);
 else return(NO);
}

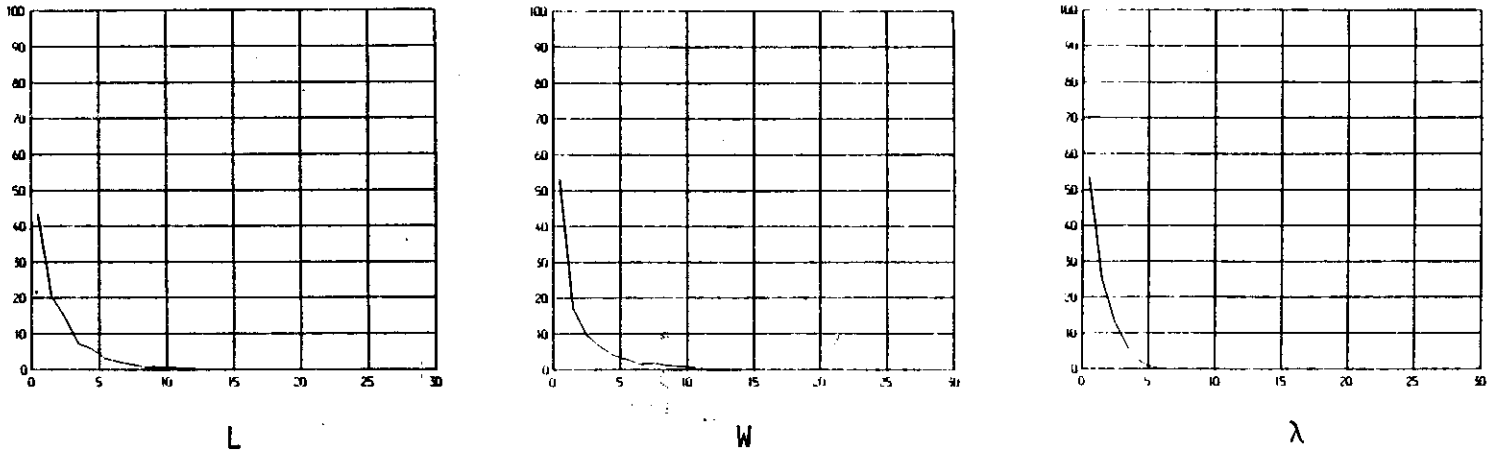

Figure 5. Densities of absolute value of percent errors for MVASUB, for networks with MS centers.

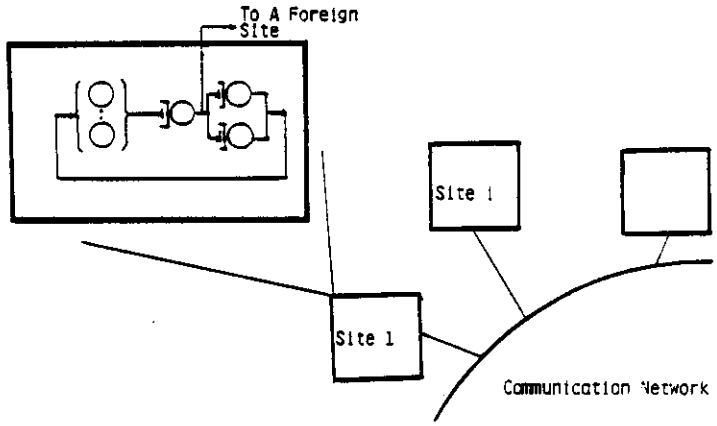

Figure 6.

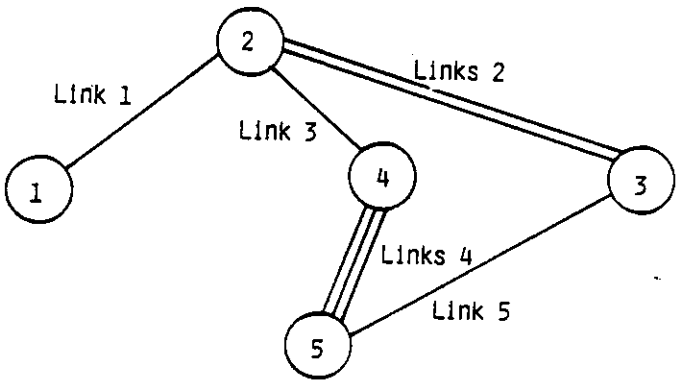

Figure 7.

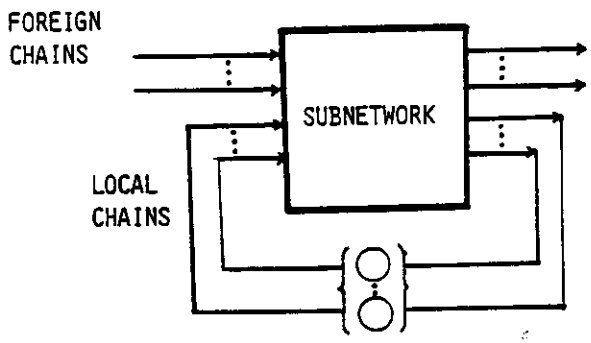

Figure 3. Representation of the effect of the complement for local and foreign chains.

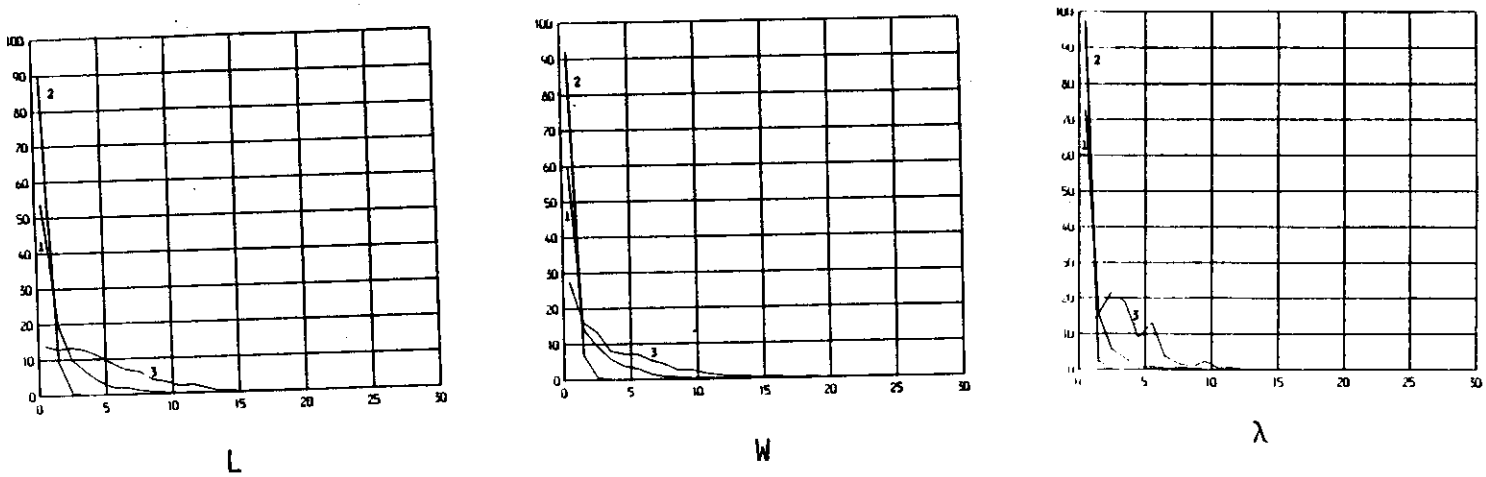

Figure 4. Densities of absolute value of percent errors for LINSUB (1), LIN (2) and SCH (3). All axes in percent.

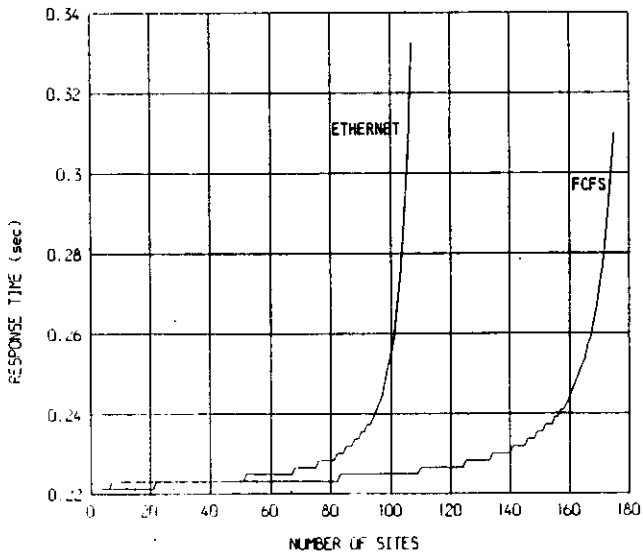

Figure 10. Mean response time versus number of sites. Message = 500 bits, MPD = .05 msec.

	Average			Average of Maxima		
	L	W	λ	L	W	λ
MVASUB	1.44	1.38	.81	5.37	5.82	2.10
LINSUB	1.54	1.44	.85	5.67	6.19	2.11
SCH	4.41	3.56	3.48	10.68	13.29	6.56
LIN	.41	.33	.26	1.12	1.34	.49

Table 1. Absolute Value of Percent Errors. First Set of Experiments.

	Average			Average of Maxima		
	L	W	λ	L	W	λ
MVASUB	1.96	1.74	1.17	6.76	7.43	2.4

Table 2. Absolute Value of Percent Errors. Second Set of Experiments.

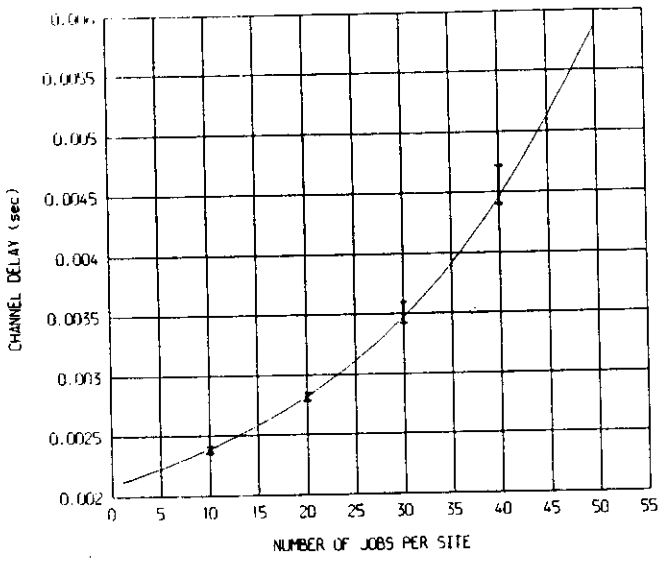

Figure 8. Average delay in the channel. Message = 1000 bits, MPD = .1 msec.

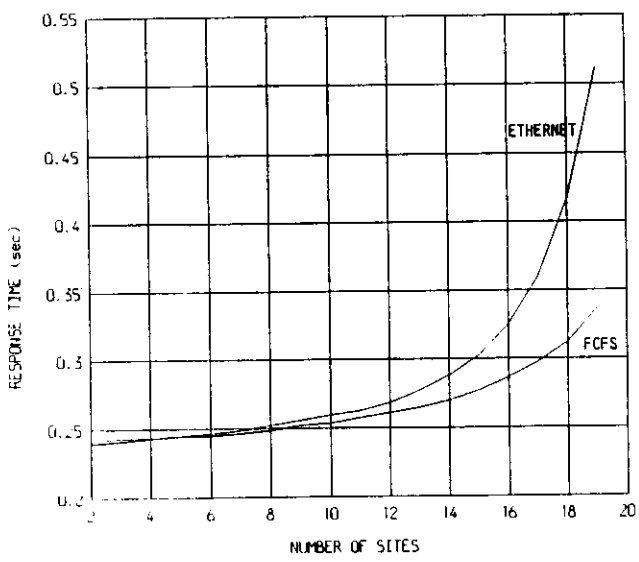

Figure 9. Mean response time versus number of sites. Message = 4000 bits, MPD = .1 msec.

LINK	CAPACITY (Kbps)	Mean Service Time (msec)	
1	70	29	
2	50	40	
3	30	66	
4	40	50	
5	40	50	
VIRTUAL CIRCUIT	WINDOW SIZE	ROUTE (Switching Nodes)	MEAN SERVICE TIME AT THE SOURCES (msec)
1	10	1 → 2 → 3	50
2	6	1 → 2 → 4 → 5	200
3	8	1 → 2 → 3	100
4	5	2 → 4 → 5	100
5	5	4 → 5 → 3	50

Table 4. Parameters of Example 3.

	Average			Maximum		
	L	W	λ	L	W	λ
11 Chain network						
LINSUB	.19	.18	.04	1.29	1.30	.8
SCH	3.54	2.44	2.09	15.3	20.9	5.75
41 Chain network						
LINSUB	.38	.33	.11	4.61	4.95	.39
SCH	4.90	3.21	3.36	15.8	24.8	10.3

Table 3. Comparison with LIN Solutions - Absolute Value of Percent Differences.

References.

[BARD81] Y. Bard, "A Simple Approach to System Modeling", Performance Evaluation 1, 225-248, 1981.

[BUX81] W. Bux, "Local-Area Subnetworks: A Performance Comparison", IEEE Transactions on Communications COM 29, no. 10, 1465-1473, 1981.

[CHAN75] K.M. Chandy, U. Herzog, and L.S. Woo, "Parametric Analysis of Queueing Networks", IBM Journal of Research and Development 19, 43-49, 1975.

[CHAN82] K.M. Chandy and D. Neuse, "Linearizer: A Heuristic Algorithm for Queueing Network Models of Computer Systems", Communications of the ACM 25, 126-134, 1982.

[DeSO83] E. de Souza e Silva, S.S. Lavenberg and R.R. Muntz, "A Perspective on Iterative Methods for the Approximate Analysis of Closed Queueing Networks", Proceedings of the International Workshop on Applied Mathematics and Performance/Reliability Models of Computer/Communication Systems, G. Iazeola and S. Tucci (Editors), 191-210, 1983.

[GOLD83] A. Goldberg, G. Popek and S.S. Lavenberg, "A Validated Distributed System Performance Model", Performance 83, A.K. Agrawala and S.K. Tripathi (Editors), 251-268, 1983.

[HEFF82] H. Heffes, "Moment Formulae for a Class of Mixed Multi-Job-Type Queueing Networks", The Bell System Technical Journal, vol. 61, no. 5, 709-745, May-June 1982.

[JACO80] P.A. Jacobson and E.D. Lazowska, "The Method of Surrogate Delays: Simultaneous Resource Possession in Analytic Models of Computer Systems", Department of Computer Science, University of Washington, Technical Report no. 80-04-03, Dec. 1980.

[JACO82] P.A. Jacobson and E.D. Lazowska, "Analyzing Queueing Networks with Simultaneous Resource Possession", Communications of the ACM 25, 142-151, 1982.

[JACO83] P.A. Jacobson and E.D. Lazowska, "A Reduction Technique for Evaluating Queueing Networks with Serialization Delays", Performance 83, A.K. Agrawala and S.K. Tripathi (Editors), North-Holland, 45-59, 1983.

[KLEI76] L. Kleinrock, "Queueing Systems, Volume II: Computer Applications", Wiley-Interscience, New York, 1976.

[LAM80] S.S. Lam, "Carrier Sense Multiple Access Protocol for Local Networks", Computer Networks 4, 21-32, 1980.

[LAM83] S.S. Lam and Y.L. Lien, "A Tree Convolution Algorithm for the Solution of Queueing Networks", Communications of the ACM 26, 203-215, 1983.

[LAVE83] S.S. Lavenberg (Editor), "Computer Performance Modeling Handbook", Academic Press, New York, 1983.

SUBNETWORK	LOCAL CHAINS	CENTERS
1	1, 3	1, 6, 8
2	2, 4	3, 7, 9
3	5	5, 10
4		2, 4

Table 5. Grouping of Example 3.

	Average			Maximum		
	L	W	λ	L	W	λ
MVASUB	0.75	0.66	.29	2.13	2.11	.84

Table 6. Comparison with MVA. Third Example.