THE TREE MVA ALGORITHM

Salvatore Tucci
Department of Computer Science
University of Pisa
56100 Pisa, Italy

Charles H. Sauer
IBM Entry Systems Division
Austin, Texas 78758

Abstract: A new algorithm to solve product form queueing networks, especially those with large numbers of centers and chains, is presented. This algorithm is a Tree version of Mean Value Analysis (MVA). Tree MVA is analogous to the Tree version of Convolution developed by Lam and Lien. Like Tree Convolution, Tree MVA allows exact solution of large networks which are intractable with previous sequential algorithms. As with the sequential versions of Convolution and MVA, Tree MVA has better numerical properties than Tree Convolution. Further, Tree MVA avoids the computational complexity of sequential MVA in networks with several queue dependent centers. Thus, we consider Tree MVA to be the best algorithm for general product form networks.

* Work performed while visiting the IBM Thomas J. Watson Research Center.
1. INTRODUCTION

Queueing networks models are widely used to evaluate the performances of computer systems and communication systems. For an introduction, see Sauer and Chandy [SAUE81b] and Lavenberg and Sauer [LAVE83].

Exact numerical solution of queueing networks is usually practical only if the joint queue length distribution in a network with $J$ centers, is given by the product form

$$P(n_1,\ldots,n_j) = \frac{X_1(n_1) \cdots X_J(n_j)}{G(N)}$$

where $X(n_j)$, $j = 1,\ldots,J$, is a factor obtained from the marginal queue length distribution of center $j$ in isolation, $N$ gives the numbers of jobs in the routing chains of the network and $G(N)$ a normalizing constant.

Although the product form solution is appealing there are strong constraints in its application. To compute $G(N)$ the numerical evaluation requires a summation of the product terms over the entire state space. The first satisfactory algorithm for this purpose was Buzen's Convolution Algorithm [BUZE73]. In this algorithm $G(N)$ is obtained from the factors $X_j(n_j)$, $j = 1,\ldots,J$, in a manner similar to the Convolution of discrete probability distributions. Later Chandy, Herzog and Woo [CHAN75], and Reiser and Kobayashi [REIS75] extended this algorithm to multichain networks. One of problems of this algorithm is potential for numerical instability. The values of $G(N)$ can exceed the range of floating-point representation in computer systems.

Reiser and Lavenberg [REIS80] proposed an alternate way to solve product form queueing networks, called Mean Value Analysis (MVA). This method avoids the computation of $G(N)$, but for queue dependent centers shows some instability [CHAN80]. A modified version of MVA proposed by Reiser [REIS81] avoids this problem. Sauer [SAUE81a] showed that MVA is as general as the Convolution Algorithm. The modified version of MVA has been implemented by MacNair and Tucci [TUCC82] and included in the Research Queueing Package [SAUE81c].

There are many applications, especially in computer-communication networks where it is important to obtain results for large queueing networks. Recently Lam and Lien [LAM81] developed a "Tree" version of the Convolution Algorithm. The "tree" designation indicates that the algorithm considers the centers of the network by traversing a tree. The leaves of the tree represent the centers and the subtrees represent subnetworks. An appropriate tree traversal Convolution may be much more efficient than the usual sequential Convolution algorithm. With the Tree Convolution algorithm solutions are feasible for certain networks which previously would have been intractable. These networks are those with many closed chains and many centers where jobs in a given chain visit only a small subset of the centers.
With the Tree Convolution Algorithm there is still the problem of numerical instability associated with the sequential Convolution algorithm. We can avoid this numerical instability with a Tree version of MVA, based on the hierarchical MVA developed by Sauer [SAUE81a]. The Tree version of MVA also avoids the excessive computational complexity of the modified MVA for networks with many queue dependent centers. Since Tree MVA avoids the potential numerical problems of Tree Convolution and the potential computational complexity of sequential MVA, we consider it to be the best algorithm available for solution of general product form networks.

In this paper we will not provide a formal justification of our claims of numerical stability and computational efficiency. We concentrate on the algorithm itself and assume that the general accuracy of the claims is evident. It should be noted that, as in the sequential versions, Tree Convolution is somewhat more computationally efficient than Tree MVA and that there are many networks that Tree Convolution can handle without numerical difficulty. A formal numerical analysis of the sequential Convolution algorithm was given by Reiser [REIS77]. Reiser’s analysis can be directly applied to the Tree Convolution algorithm. Sauer and Chandy [SAUE81b] discuss example problems where the Convolution algorithm fails numerically and example problems where the original MVA of Reiser and Lavenberg [REIS80] fails numerically. Discussion of the computational complexity of product form algorithms is found in Reiser [REIS77], Reiser and Lavenberg [REIS80], Zahorjan [ZAH080], Sauer [SAUE81a], Lam and Lien [LAM81] and Lavenberg and Sauer [LAVE83].

The rest of the paper assumes that the reader is familiar with Lam and Lien [LAM81]. Section 2 of the paper discusses the Tree MVA algorithm, Section 3 presents some complexity aspects and Section 4 describes application of the algorithm to an example network.

2. TREE MEAN VALUE ANALYSIS

In the following we consider multichain closed product form networks with queue-dependent service rates but without service rates or routing dependent on subnetwork population [BASK75,TOWS80,SAUE81a]. Restricting our attention to closed networks is not a true limitation because the solution of mixed networks is based on solution of closed networks [REIS75,SAUE81a,LAVE83].

For notational convenience, we assume the number of centers in the network, \( J \), is a power of two, i.e., \( J = 2^p \), \( p \) a positive integer. Without loss of generality we assume \( p \geq 2 \), since Tree MVA is the same as sequential MVA for \( J = 2 \). We assume we have a binary tree representation of the network obtained from one of the tree planting procedures of Lam and Lien [LAM81]. A Tree MVA implementation would likely use a general tree representation rather than a binary tree, and would likely use a slightly different tree planting procedure, as we discuss in Section 4.
Following Reiser [REIS77] we define the routing in terms of "local classes" and "routing chains" (see also Sauer and Chandy [SAUE81b]). A network is defined by $J$, by $C$, the number of classes in the network, by $K$, the number of chains in the network, $N_k$, $k = 1,...,K$, the population of chain $k$, the routing between centers of the network, the mean service demands at the centers and the service capacities at the centers.

Let $N = \sum_{k=1}^{K} N_k$. Let $r_{dc}$ be the probability a job leaving class $d$ goes to class $c$, let $\mathcal{C}_j$ be the set of classes belonging to center $j$ and let $\mathcal{K}_k$ be the set of classes of chain $k$. The relative throughputs for the classes $y_c$, $c = 1,...,C$ are positive values which satisfy

$$y_c = \sum_{d \in \mathcal{K}_k} y_d r_{dc}, \quad c \in \mathcal{K}_k, \quad k = 1,...,K$$

The relative throughput at center $j$ for chain $k$ is given by

$$y(k,j) = \sum_{c \in \mathcal{C}_j \cap \mathcal{K}_k} y_c, \quad k = 1,...,K, \quad j = 1,...,J. \quad (1)$$

Let the mean service demand at class $c$ be $S_c$. The mean service demand at center $j$ and chain $k$ is given by

$$S(k,j) = \frac{\sum_{c \in \mathcal{C}_j \cap \mathcal{K}_k} y_c S_c}{y(k,j)}, \quad k = 1,...,K, \quad j = 1,...,J.$$

The service capacity of center $j$ given $n$ jobs at the center is specified by the positive valued function $\mu_{(j)}(n)$, $n = 1,...,N$.

The basic idea of the Tree MVA algorithm is to consider the $2^J$ centers of the network in pairs and to replace each pair by a single aggregate center. The resulting network will consist of $2^{J-1}$ aggregate centers. This process is repeatedly applied, thus giving rise to new networks with $2^{J-2}$, $2^{J-3}$, ..., $2^0$ aggregate centers. The results of those networks are then interpreted to obtain results for the original network with $2^J$ centers.

The centers, when the network is represented as a binary tree, are ordered in such a way as to minimize time or memory requirements for the computation.

We indicate the $J = 2^P$ centers by the corresponding serial number and the aggregates, i.e., pairs of centers $\{2j-1, 2j\}$ for $j = 1,...,2^{P-1}$, by $P_j^{-1}$, $j = 1,...,2^{P-1}$ and we will call them aggregates of $(p-1)$-level of aggregation. The aggregation is justified by "Norton's theorem" for queueing networks [CHAN75] and related results by Sauer and Chandy [SAUE81b] and Sauer [SAUE81a]. For each pair of centers $\{2j-1, 2j\}$, $j = 1,...,2^{P-1}$ we can have two sets of chains that we call $\mathcal{A}_j^{P-1}$ and $\mathcal{B}_j^{P-1}$, $j = 1,...,2^{P-1}$. The set $\mathcal{A}_j^{P-1}$ contains the chains visiting only the aggregate $P_j^{P-1}$, $j = 1,...,2^{P-1}$. Following Lam and Lien [LAM81] we call these "covered chains".
contains the chains that also visit other aggregates. These are called "partially covered" chains.

In order to build the $2^{p-1}$ aggregates at $(p-1)$-level of aggregation, we apply the sequential MVA Algorithm separately for the pair of centers $\{2j-1, 2j\}, j = 1, \ldots, 2^{p-1}$. We use the following notation for performance measures determined by MVA:

$$Q'_{(k,j)}(N) - \text{ the queueing time at center or aggregate } j \text{ for chain } k \text{ when } \vec{N} \text{ jobs are in the level } l \text{ subnetwork.}$$

$$L'_{(k,j)}(N) - \text{ the mean queue length for center or aggregate } j \text{ for chain } k \text{ when } \vec{N} \text{ jobs are in the level } l \text{ subnetwork.}$$

$$R'_{(k,j)}(N) - \text{ the throughput for center or aggregate } j \text{ for chain } k \text{ when } \vec{N} \text{ jobs are in the level } l \text{ subnetwork.}$$

For each centers or aggregate $a$ at level $l$ we will use $\vec{N}$ to denote the vector whose components are the populations of the chains at level $l-1$. (To be more precise, instead of $\vec{N}$ we would use the notation $\vec{N}_{k\in\mathcal{A}_{j}^{p-1}\cup\mathcal{B}_{j}^{p-1}}$ we avoid this notation as too cumbersome.)

Let $\vec{e}_k$ be a vector with a one in the $k$th position and zero elsewhere. For the pairs of centers at level $p$ we have the usual MVA queueing time equations. For IS (Infinite Server) centers

$$Q_{(k,q)}(\vec{N}) = \frac{S_{(k,q)}}{\mu_{(q)}(1)} \frac{\vec{N} \neq 0}{q = 2j - 1, 2j} \quad j = 1, \ldots, 2^{p-1} \quad k \in \mathcal{A}_{q}^{p-1} \cup \mathcal{B}_{q}^{p-1}$$

(2)

For single server fixed rate servers

$$Q_{(k,q)}(\vec{N}) = \frac{S_{(k,q)}}{\mu_{(q)}(1)} (1 + L_{(q)}(\vec{N} - \vec{e}_k)) \quad N \geq \vec{e}_k$$

$$q = 2j - 1, 2j$$

$$j = 1, \ldots, 2^{p-1}$$

$$k \in \mathcal{A}_{q}^{p-1} \cup \mathcal{B}_{q}^{p-1}$$

(3)

For queue-dependent centers

* The set $\mathcal{A}_{j}^{p-1}$ can be empty.
\[ Q^p_{(k,q)} (N) = S_{(k,q)} \sum_{i=1}^{N} \frac{i}{\mu_{(q)}(i)} P^p_{(q)} (i-1 | N \rightarrow e_k) \]

where

\[ P^p_{(q)} (i | N) = \frac{1}{\mu_{(q)}(i)} \sum_{k \in \mathcal{A}_q^{p-1} \cup \mathcal{B}_q^{p-1}} S_{(k,q)} R^p_{(k,q)} (N) P^p_{(q)} (i-1 | N \rightarrow e_k) \]

and

\[ P^p_{(q)} (0 | N) = \frac{R^p_{(k,q)}(N)}{N_k \mu_{(q+1)}(N) y_{(k,q)}} P^p_{(q)} (0 | N \rightarrow e_k) \]

and \( k \) is a chain for which \( N_k > 0 \). For the center \( q + 1 \)

\[ P^p_{(q+1)} (i | N) = P^p_{(q)} (N - i | N), i = 0, ..., N \]

Throughput is determined by Little's Rule [LITT61]

\[ R^p_{(k,q)} (N) = y_{(k,q)} \frac{N_k}{\sum_{i=1}^{2} y_{(k,2j+i)} Q^p_{(k,i)} (N)} \]

and then mean queue length is determined by Little's Rule:

\[ L^p_{(k,q)} (N) = R^p_{(k,q)} (N) Q^p_{(k,q)} (N) \]

Starting from the initial conditions \( L^p_{(k,q)} (0) = 0 \) for all chains and all centers, and \( P_{(q)} (0 | 0) = 1 \) for all queue dependent centers and using the equations (2) - (9) we solve each
pair of centers for the total population of chains $\mathcal{A}_j^{p-1}$ for each combination of populations in chain $\mathcal{B}_j^{p-1}$ (if the set $\mathcal{A}_j^{p-1}$ is empty the iteration will be done only on $\mathcal{B}_j^{p-1}$).

The results we eventually obtain are mean queueing time, mean queue length, throughput and utilization. Mean queueing length, throughput and utilization are all obtained in a similar manner, so we will only consider mean queue length. Mean queue length is obtained from Little's Rule. Mean queue length is obtained for each combination of populations in chains $\mathcal{B}_j^{p-1}$. We indicate this queue length as in the above equations

$$L^p_{(k,\mathcal{A})} \left( n_{\mathcal{B}_j^{p-1}} \right) \quad q = 2j - 1, 2j \quad j = 1, \ldots, 2^{p-1} \quad k \in \mathcal{A}_j^{p-1} \cup \mathcal{B}_j^{p-1}$$

where by $n_{\mathcal{B}_j^{p-1}}$ we indicate the queue length is obtained for each combination of populations in the chain set $\mathcal{B}_j^{p-1}$. In effect the term in parenthesis should be $n_{\mathcal{A}_j^{p-1}}, n_{\mathcal{B}_j^{p-1}}$ where the comma indicates concatenation of the vectors.

Now we have $2^{p-1}$ aggregates and we can proceed in a second step of aggregation. The subsequent aggregation steps (if any) are performed in a manner analogous to the second step. We represent those aggregates as a composite center [CHAN75, SAUE81a, SAUE81b] having service rate

$$\Lambda^p_{(k,\mathcal{F})} \left( n_{\mathcal{B}_j^{p-1}} \right) = \frac{1}{y_{(k,\mathcal{F})}} R^p_{(k,\mathcal{F})} \left( n_{\mathcal{B}_j^{p-1}} \right) \quad q = 2j - 1, 2j \quad j = 1, \ldots, 2^{p-1} \quad \forall k \in \mathcal{B}_j^{p-1}$$

Note that the composite center gives service simultaneously to jobs of all chains, at rates dependent on numbers of jobs of specific chains.

At this step we solve the aggregates

$$\mathcal{P}_j^{p-2} = \{\mathcal{P}_j^{p-2}, \mathcal{P}_j^{p-2} \} \quad j = 1, \ldots, 2^{p-2} - 1$$

applying the sequential MVA algorithm for composite centers [SAUE81b].

For each aggregate $\mathcal{P}_j^{p-2}$, $j = 1, \ldots, 2^{p-2}$ we have two sets of partially covered chains, the sets $\mathcal{B}_j^{p-2}$ belonging to the aggregates $\mathcal{P}_j^{p-1}$ and the sets $\mathcal{B}_j^{p-2}$ belonging to the aggregates $\mathcal{P}_j^{p-1}$, $j = 1, \ldots, 2^{p-2}$. The sets $\mathcal{B}_j^{p-2}$, consisting of the partially covered chains belonging to the aggregates $\mathcal{P}_j^{p-2}$ are determined as $\mathcal{B}_j^{p-1} \cup \mathcal{B}_j^{p-2} \mathcal{A}_j^{p-2}$, $j = 1, \ldots, 2^{p-2}$ where, for each $j$, $\mathcal{A}_j^{p-2}$ is the set of partially covered chains at level $p - 1$ that become fully covered at level $p - 2$.

The aggregates are solved with the populations of the sets $\mathcal{A}_j^{p-2}$ for all combinations of populations in the sets $\mathcal{B}_j^{p-2}$ for $j = 1, \ldots, 2^{p-2}$. Note that we don't take into account the
populations in the sets \( \mathcal{A}_j^{p-1} \) (the fully covered chains at the previous level) since their influences are completely summarized in the performance measures of the \( p \)-level given by equation (10). Thus for each combination of populations in the set \( \mathcal{A}_j^{p-2} \), and for each of the \( 2^{p-2} \) subnetworks, the equation corresponding to equation (4) is

\[
Q^{p-1}_{(k, \mathcal{G}_q^{p-1})} (N) = \sum_{n=\mathbf{e}_k}^{N} \Lambda_{(k, \mathcal{G}_q^{p-1})}(n) \frac{n_k}{n} \sum_{k \in \mathcal{A}_j^{p-2} \cup \mathcal{A}_j^{p-2}} \left( \mathbf{P}^{p-1}_{(\mathcal{G}_q^{p-1})}(n - \mathbf{e}_k | N - \mathbf{e}_k) \right) \quad (12)
\]

The equation corresponding to equations (5) and (6) are:

\[
P^{p-1}_{(\mathcal{G}_q^{p-1})} (n | N) = \frac{1}{\Lambda_{(k, \mathcal{G}_q^{p-1})}(n)} R^{p-1}_{(k, \mathcal{G}_q^{p-1})} (N) P^{p-1}_{(\mathcal{G}_q^{p-1})} (n - \mathbf{e}_k | N - \mathbf{e}_k) \quad (13)
\]

and

\[
P^{p-1}_{(\mathcal{G}_q^{p-1})} (0 | N) = \frac{R^{p-1}_{(k, \mathcal{G}_q^{p-1})}(N)}{\Lambda_{(k, \mathcal{G}_q^{p-1})}(N)} P^{p-1}_{(\mathcal{G}_q^{p-1})} (0 | N - \mathbf{e}_k) \quad (14)
\]

for one chain \( k \) where \( N_k > 0 \). Then

\[
P^{p-1}_{(\mathcal{G}_q^{p-1})} (n | N) = P^{p-1}_{(\mathcal{G}_q^{p-1})} (N - n | N) \quad (15)
\]

The equations corresponding to equations (8) and (9) are

\[
R^{p-1}_{(k, \mathcal{G}_q^{p-1})} (N) = \frac{N_k}{\sum_{q=2j-1,2j}^{q=2j-1,2j} Q^{p-1}_{(k, \mathcal{G}_q^{p-1})}(N)} \quad (16)
\]
\[ i = U - 2 \]
\[ \text{and} \]
\[ L_{(k, \mathcal{G}_q)}^{p-1} (N) = R_{(k, \mathcal{G}_q)}^{p-1} (N) Q_{(k, \mathcal{G}_q)}^{p-1} (N) \]  
\[ k \in \mathcal{A}_j^{p-2} \cup \mathcal{B}_j^{p-2} \]

Before passing to the next level of aggregation (the aggregation at the \( p-2 \) level to obtain the \( p-3 \) level) we must remove in the performance measures for the centers in \( \mathcal{G}_q^{p-2}, q = 2j - 1, 2j, j = 1, \ldots, 2^{p-3} \) the conditioning on combination of population in the aggregates \( \mathcal{G}_q^{p-1}, q = 2j - 1, 2j, j = 1, \ldots, 2^{p-2} \). This will be performed using

\[ L_{(k, m)}^{p-1} \left( \frac{n_{\mathcal{A}_j^{p-2}}}{n_{\mathcal{B}_j^{p-2}}} \right) = \sum_{n = \epsilon_k} L_{(k, m)}^{p-1} (n) M_{(k, 91)}^{p-1} (n \mid N) \]

\[ m = 1, \ldots, J \]
\[ j = 1, \ldots, 2^{p-2} \]
\[ q = 1, \ldots, 2^{p-1} \]
\[ k \in \mathcal{A}_j^{p-2} \cup \mathcal{B}_j^{p-2} \]

This is simply the extension to the multichain case of Theorem 6.2 of Sauer and Chandy [SAU81b].

We will continue this process of aggregation of pair of composite centers representing each pair, again by a composite queue and so on until the last step when we have two composite centers \( \mathcal{G}_q^1 q = 1, 2 \) with service rates

\[ \Lambda_{(k, \mathcal{G}_q^1)} \left( \frac{n_{\mathcal{B}_q^1}}{n_{\mathcal{B}_q^1}} \right) = R_{(k, \mathcal{G}_q^1)} \left( \frac{n_{\mathcal{B}_q^1}}{n_{\mathcal{B}_q^1}} \right) \]

\[ V k \in \mathcal{G}_q^1 \]
\[ q = 1, 2 \]

At this level we have also all performance measures, e.g.,

\[ L_{(k, m)}^1 \left( \frac{n_{\mathcal{B}_q^1}}{n_{\mathcal{B}_q^1}} \right) \]

\[ m = 1, \ldots, J \]
\[ q = 1, 2 \]

We apply MVA for the last time. Now, \( \mathcal{B}_1^1 = \mathcal{B}_2^1 \), so after this last aggregation the chain set \( \mathcal{B} \) is empty. With the (18) we will obtain the actual performance measures for the centers:
The remaining performance measures for chains are similarly obtained.

3. COMPLEXITY ANALYSIS

Let us consider a network with \( J \) service centers, with \( K \) closed chains and with \( N_k \) jobs in chain \( k \). Assuming all fixed rate servers, to compute the normalizing constant using the Convolution algorithm requires space on the order of \( \prod_{k=1}^{K} (N_k + 1) \) and time on the order of
\[
\sum_{k=1}^{K} \prod_{k=1}^{K} (N_k + 1).
\]
Without the assumption of fixed rate servers, i.e., with queue dependent service at all centers, the space requirement is on the order of \( 2^{K} \prod_{k=1}^{K} (N_k + 1) \) and the time requirement is on the order of
\[
(J - 1) \prod_{k=1}^{K} \frac{(N_k + 2)(N_k + 1)}{2}.
\]
Additional time and space are required to determine performance measures.

With Mean Value Analysis, with fixed rate service centers, the time and space requirements are both on the order of \( \prod_{k=1}^{K} (N_k + 1) \). In the case of a network with \( J'' \) queue-dependent service centers and \( J' \) fixed rate service centers, it is easy to show, using the approach suggested by Reiser [REIS81], that the time requirement is on the order of
\[
\sum_{j=0}^{J''} (j + J') (j'') \prod_{k=1}^{K} (N_k + 1).
\]
Indeed, note that to avoid the numerical instability of the original MVA algorithm it is necessary to solve \( 2^{J''} \) additional subnetworks [LAVE82]. Similarly, the space requirement is on the order
With Tree MVA we have to solve \( J-1 \) different two center networks. Each of these networks must be solved a number of times equal to the number of different combinations of populations in partially covered chains visiting the (two center) network. For each pair of centers and each combination of populations in the \( \mathcal{B} \) set there is a time requirement on the order of

\[
2( | \mathcal{A} | ) \prod (N_k + 1),
\]

where \( | \mathcal{A} | \) is the number of fully covered chains at the considered level of aggregation and where the product is over the chains in \( \mathcal{A} \). Of course, we will assume that this quantity is one in the case where \( | \mathcal{A} | = 0 \) (i.e., a pair without fully covered chains at the level considered).

If this pair is visited by partially covered chains, one has to solve this pair \( \prod (N_k + 1) \) times, where the product is over the chains in \( \mathcal{B} \).

Having \( 2^{l-1} \) pairs at level \( l \) and \( p-l \) levels, the total time requirement is on the order

\[
\sum_{i=1}^{p} \sum_{j=1}^{2^{l-1}} \left[ \prod_{k \in \mathcal{A}_{j}^{l-i}} (N_k + 1) \right] \left[ 2 | \mathcal{A}_{j}^{p-l} | \prod_{k \in \mathcal{A}_{j}^{p-l}} (N_k + 1) \right]
\]

A similar expression holds for the space requirement.

4. EXAMPLE

Suppose we want to solve the network in Figure 1.
Figure 1 - An Example of Computer Network

We have $p=3$ and seven chains. Chain 1 has classes in the set of centers $\{1,2\}$, chain 2 has classes in the set of centers $\{3,4\}$, chain 3 has classes in the set of centers $\{5,6\}$, chain 4 has classes in the set of centers $\{7,8\}$, chain 5 has classes in the set of centers $\{2,4\}$, chain 6 has classes in the set of centers $\{6,8\}$ and chain 7 has classes in the set of centers $\{2,4,6,8\}$. Each chain has two jobs. The tree representation of the network is given in Figure 2.

Figure 2 - Tree Representation of Networks in Figure 1

At level 3 we will make four aggregations consisting in the solution of the four subnet-
works depicted in Figure 3.

![Diagram of networks](image)

**Figure 3 - The Aggregation of Centers**

For each subnetwork \( \mathcal{S}_q^l \), \( q = 1, \ldots, 2^l \) at level \( l = p - 1 = 2 \), one chain belongs to the set \( \mathcal{A}_q^l \) and two chains belong to the set \( \mathcal{B}_q^l \).

We apply sequential MVA to the subnetworks separately and solve each subnetwork \( \mathcal{S}_q^l \), \( q = 1, \ldots, 2^l \) in chain of \( \mathcal{A}_q^l \) for each combination of populations in \( \mathcal{B}_q^l \). In the case \( q = 1 \), the set \( \mathcal{A}_1^l = \{1\} \) and \( \mathcal{B}_1^l = \{5,7\} \) and we will obtain the utilization \( U_{(k,j)}(\mathcal{N}) \), the throughput \( R_{(k,j)}(\mathcal{N}) \), and the mean queue length \( L_{(k,j)}(\mathcal{N}) \), for centers \( j = 1,2 \) and chains \( k \in \{1\} \cup \{5,7\} \), where \( \mathcal{N} \) is the vector indicating the populations in chain \( \{1\} \) and in chains \( \{5,7\} \), i.e.,

\[
\begin{align*}
U_{(k,j)}(2,0,0) & & R_{(k,j)}(2,0,0) & & L_{(k,j)}(2,0,0) \\
U_{(k,j)}(2,0,1) & & R_{(k,j)}(2,0,1) & & L_{(k,j)}(2,0,1) \\
& & & & \\
& & & & \\
& & & & \\
U_{(k,j)}(2,2,2) & & R_{(k,j)}(2,2,2) & & L_{(k,j)}(2,2,2)
\end{align*}
\]

for centers \( j = 1,2 \) and chains \( k \in \{1\} \cup \{5,7\} \) analogously for the centers \( j = 3, \ldots, 8 \) for which the sets of chains, of course, are not the same. Now we proceed to the next level of aggrega-
tion \( l = p - 2 \), solving the two subnetworks depicted in Figure 4.

Each aggregate \( \mathcal{S}_q^{p-1} \) is represented by a composite center with service rate given by (11). The relative throughput in this example is 1 so the service rate will be

\[
\Lambda_{(k, \mathcal{S}_q^{p-1})}(0,0) = R_{(k,q)}(0,0)
\]

\[
\vdots
\]

\[
\Lambda_{(k, \mathcal{S}_q^{p-1})}(2,2) = R_{(k,q)}(2,2)
\]

where \( q \) is any center belonging to the aggregate and \( k \in \{5\} \cup \{7\} \) for \( \mathcal{S}_1^{p-1} \) and \( \mathcal{S}_2^{p-1} \) and \( k \in \{6\} \cup \{7\} \) for \( \mathcal{S}_3^{p-1} \) and \( \mathcal{S}_4^{p-1} \).

At this level chains 5 and 6 become fully covered chains.

Using equations (2) - (9), we solve the two subnetworks, \( \mathcal{S}_q^{1} \), \( q = 1,2 \), for each combination of populations in the sets of chains \( \{5\} \cup \{7\} \) and \( \{6\} \cup \{7\} \), respectively obtaining \( P_q^2(\hat{n} | \hat{N}) \), \( q = 1,2 \) for \( \hat{n} = (0,0), \ldots, (2,2) \).

At this point we apply equation (18). For \( \mathcal{S}_1^l \)

\[
L_{(k,q)}((2,2),0) = L_{(k,q)}((2,0,0) \ P_1^2(0,0 | 0) + \ldots + L_{(k,q)}((2,2,2) \ P_1^2(2,2 | 0)
\]

\[
L_{(k,q)}((2,2),1) = L_{(k,q)}((2,2,0) \ P_1^2(0,0 | 1) + \ldots + L_{(k,q)}((2,2,2) \ P_1^2(2,2 | 1)
\]

\[
L_{(k,q)}((2,2),2) = L_{(k,q)}((2,2,0) \ P_1^2(0,0 | 2) + \ldots + L_{(k,q)}((2,2,2) \ P_1^2(2,2 | 2)
\]
where \( q = 1,2 \) \( k \in \{1,5\} \cup \{7\} \). We proceed analogously for the other measures.

![Figure 5 - The Last Aggregation](image)

The last step \( l=1 \) is to aggregate \( \mathcal{S}_1 \) and \( \mathcal{S}_2 \) as depicted in Figure 5, applying MVA to two composite centers having service rates given by equation (11):

\[
\Lambda_{(k,\mathcal{S}_q)}(n) = R_{(k,q)}(n), \quad n = 0,1,2
\]

with \( k = 7 \) and any \( q \in \mathcal{S}_1 \) for the \( \mathcal{S}_1 \) aggregate and any \( q \in \mathcal{S}_2 \) for the \( \mathcal{S}_2 \) aggregate. We obtain now for the two aggregates

\[
P_{1}(n | 2) \text{ and } P_{2}(n | 2) \quad n = 0,1,2
\]

respectively, and using equation (18) we finally will obtain:

\[
U_{(k,q)}(2,2,2) = \sum_{q \in \mathcal{S}_1}^{2} U_{(k,q)}((2,2),n) P_{i}^{1}(n | 2) \quad i = 1,2
\]

\[
R_{(k,q)}(2,2,2) = \sum_{q \in \mathcal{S}_1}^{2} R_{(k,q)}((2,2),n) P_{i}^{1}(n | 2) \quad i = 1,2
\]

\[
L_{(k,q)}(2,2,2) = \sum_{q \in \mathcal{S}_1}^{2} L_{(k,q)}((2,2),n) P_{i}^{1}(n | 2) \quad i = 1,2
\]

The queueing time is given by Little's rule:

\[
Q_{(k,q)}(N) = \frac{L_{(k,q)}(N)}{R_{(k,q)}(N)} , q = 1,...,J, \quad k = 1,...,K.
\]

5. CONCLUSION

We have presented Tree MVA, a new algorithm to solve product form queueing networks. The two most important characteristics of Tree MVA with respect to sequential MVA are
1. This algorithm allows exact solution of large networks which are intractable, because of time and space constraints, with previous versions of MVA.

2. This algorithm avoids the computational complexity of sequential MVA in networks with several queue dependent centers.

We developed the algorithm assuming a binary tree representation of the network obtained from one of the tree planting procedure that Lam and Lien introduced for their Tree Convolution Algorithm. In a Tree MVA implementation, other factors must be considered in the tree planting. From a complexity point of view, the tree must be binary from the \( p-1 \) level to the root. Otherwise, the complexity problems of sequential MVA for queue-dependent service rates will arise.

At the bottom level, \( p \)-level, the most convenient solution seems to aggregate the queue dependent centers in couples. For subnetworks of fixed rate centers, a trade-off between computational complexity and space requirement is possible. If the memory space allows the solution of the subnetwork with the sequential MVA algorithm all centers of the subnetworks can be replaced by one equivalent center with queue dependent rates. Otherwise, the subnetwork may be further subdivided into subnetworks, each to be solved with the sequential MVA algorithm and each subnetwork to be replaced by one equivalent center.

ACKNOWLEDGEMENT

We are grateful to Y.L. Lien for his many helpful comments on this work and on a draft of this paper. We would like to thank the anonymous referees for their many constructive suggestions.

REFERENCES


S. S. Lam, Y. L. Lien, "A Tree Convolution Algorithm for the Solution of Queueing Networks" Technical Report TR-165, Department of Computer Science, University of Texas at Austin. *CACM* 26, 3 (March 1983).


J. Zahorjan, "The Approximate Solution of Large Queueing Network Models," CSRG-12,2, University of Toronto.