Mean Value Analysis Approximation for Multiple Server Queueing Networks

Ian F. Akyildiz
School of Information and Computer Science, Georgia Institute of Technology, Atlanta, Georgia 30332-0280, U.S.A.

Gunter Bolch
IMMDIV, Universität Erlangen-Nürnberg, Martensstrasse 1, 8520 Erlangen, Fed. Rep. Germany

Received May 1986
Revised 3 June 1987

An approximation of mean value analysis is presented for queueing networks containing multiple server stations. The approximation is based on the estimation of the conditional marginal probabilities used by the mean residence time formula in mean value analysis. A comparison against classical mean value analysis allows us to determine the accuracy of our algorithm. In all investigated network models, the approximate results vary from the exact results by less than four percent on the average. This approximation method has all the advantages of classical mean value analysis; specifically, it is easy to implement and has a very short run time.

Keywords: Queueing Network, Analytical Method, Mean Value Analysis, Conditional Marginal Probability.

1. Introduction

Mean value analysis has enjoyed widespread popularity during recent years as an exact technique for providing solutions to product-form queueing networks. The basic concept of mean value analysis is the application of an iterative procedure to calculate mean residence time, system throughput, and the mean number of jobs. A number of studies about mean value analysis has been published in the last few years

I.F. Akyildiz was born in 1954 in Istanbul, Turkey. He received his B.S. (1978), M.S. (1981), and Ph.D. (1984) degrees in Computer Science from the University of Erlangen-Nürnberg, Fed. Rep. Germany. From 1981 through 1985 he served as a Scientific Employee in the Informatik IV (Operating Systems, chair of Prof. F. Hoffman) Department at the University of Erlangen-Nürnberg. During that time he coauthored a text book titled *Analysis of Computer Systems* (in German) published by Teubner Verlag in 1982. In January of 1985 he joined the faculty in the Computer Science Department at Louisiana State University as an Assistant Professor. He was also a Visiting Professor in the Computer Science Department at the University of Florida in the summer of 1985 and in the Computer Science Department of the Universidad Tecnica de Federico Santa Maria in Valparaiso, Chile in the summer of 1986. In Fall 1987 he joined the faculty in the School of Information and Computer Science at Georgia Institute of Technology as an Assistant Professor. His research interests are performance evaluation, operating systems, and communication networks.

Dr. Akyildiz is a member of IEEE, ACM (SIGOPS and SIGMETRICS), GI (Gesellschaft für Informatik), MMB (German Interest Group in Measurement, Modeling and Evaluation of Computer Systems).

Gunter Bolch received his Ph.D. in Electrical Engineering from the University of Karlsruhe in 1973, where he also worked as an Assistant Professor. In 1974 he joined the Computer Science Department of the University of Erlangen-Nürnberg. From 1977 to 1979 he was a Visiting Professor in the Computer Science Department of The Catholic University of Rio de Janeiro (PUC). Since 1982 he has been the "Akademischer Direktor" in the operating systems division of the Computer Science Department at the University of Erlangen-Nürnberg. He spent two months (April–May 1986) at the Mathematical Institute of the Academy Sciences in Minsk, U.S.S.R. and another two months (August–September 1987) at the Computer Science Department of PUC in Rio de Janeiro. Dr. Bolch is a coauthor of a text book on performance analysis and has published numerous papers. His research interest are operating systems, process control, and analytic modeling of computer systems.

Dr. Bolch is a member of GI (Gesellschaft für Informatik) and MMB (German Interest Group in Measurement, Modeling and Evaluation of Computer Systems).

North-Holland
Performance Evaluation 8 (1988) 77–91

[1,3,6,7,10–15,17–21]. The principle advantage to mean value analysis lies in its ability to effectively compute the performance measures without computing the normalization constants. However, when considering systems with multiple job classes, the storage requirement of mean value analysis increases very rapidly. This problem is further magnified when one incorporated multiple server stations into mean value analysis.

Therefore, Reiser and Lavenberg [13], Bard [2], and Schweitzer [15] have introduced an approximation for mean value analysis which eliminates the storage complexity problems associated with classical mean value analysis, in short form MVA. They approximate the mean number of jobs at each station for the network, and then iterate on this approximation, halting when the number of jobs at each of the stations stabilizes. However, this approach is limited to single-server models.

Chandy and Neuse [4] improved the work of Bard and Schweitzer with the *Core* and *Linearizer* algorithms, but this work was still limited to single-server networks. The method involved estimating the fraction of class-r jobs in each station and the fractional change in this value when one job from class s is removed from the station. The Chandy and Neuse *Core* algorithm took as input these estimates for fractional changes in the number of jobs at each station with the removal of any one job. This algorithm can be shown equivalent to the method proposed by Bard and Schweitzer if one assumes this fractional change is zero. The *Linearizer* algorithm was an improvement over the *Core* algorithm as it applied to *Core* algorithm to each of the $(\boldsymbol{K} - 1_r)$ populations in order to improve their performance measures estimates. These improved performance measures were then used in calculating the necessary values for the final population $\boldsymbol{K}$.

The first significant technique introduced for the analysis of multiple server stations was done by Neuse and Chandy [10] in 1981. SCAT (for Self Correcting Approximation Technique), allowed the analysis of queueing network models with multiple job classes and multiple servers. This method involved application of the *Core* algorithm, as introduced in their earlier work with the Linearizer algorithm. The approximation was taken back two steps from the final network, in an effort to improve the values used by *Core* in the final approximation. It also provided a simplistic method for approximating the conditional marginal probabilities for the multiple-server stations. However, experience has shown that large errors can occur using the SCAT method. While SCAT represented the first approximation method for multiple-server networks, it was still prone to error. The lack of accuracy of the SCAT method is due primarily to the inadequate approximations for the conditional marginal probabilities.

Heinselman [5], Krzesinski and Greyling [7] as well as Zahorjan and Lazowska [19] have independently introduced improved algorithms for computing approximate solutions of queueing networks with multiple classes and multiple-server stations. All of these investigations provide tolerance errors on performance measures that are under five percent.

In this work we present a new mean value analysis approximation method for BCMP networks with multiple job classes and multiple-server stations. The method contains all the advantages of classical mean value analysis while requiring far less storage than the classical approach.

2. Mean value analysis

Classical mean value analysis is based on two theorems: The *Arrival Instant Distribution* theorem of Lavenberg and Reiser [8] and Sevcik and Mitrani [16] and Little's Law [9]. The Arrival Instant Distribution Theorem states that "*a class-r* job, arriving at station i in the system with population $\boldsymbol{K}$, sees the system with population $(\boldsymbol{K} - 1_r)$ in equilibrium". From the Arrival Instant Distribution Theorem the mean residence time of a class-r job in station i, $\bar{t}_{ir}(k)$, is computed. Using Little's Law, the system throughput for each class, $\lambda_r(\boldsymbol{k})$, and the mean number of jobs, $\bar{k}_{ir}(\boldsymbol{k})$, are obtained. The general algorithm for classical mean value analysis with multiple server stations is given below.

Notation

$\boldsymbol{K}$	: the total network population;
K_r	: the total number of jobs in class r;

N : the total number of stations in the network;
R : the total number of classes in the network;
μ_{ir} : the mean service rate of a class-r job at station i;
m_i : the number of servers at station i,
$\boldsymbol{k}$: a job vector denoted by $(k_1, k_2, \ldots, k_R)$;
$\boldsymbol{k} - 1_r$: the job vector $\boldsymbol{k}$ with one job removed from class r;
$p_{ir;js}$: the transition probability of a class-r job at station i to class s at station j;
$e_{ir} = \sum_{j=1}^{N} \sum_{s=1}^{R} e_{js} p_{js;ir}$: the mean number of visits a class-r job makes to station i;
$x_{ir} = e_{ir}/\mu_{ir}$: the relative utilization of station i for class-r jobs;
$\bar{t}_{ir}(\boldsymbol{k})$: the mean residence time for a class-r job at station i, given population $\boldsymbol{k}$;
$\bar{k}_{ir}(\boldsymbol{k})$: the mean number of jobs at station i in class r, given population $\boldsymbol{k}$;
$\lambda_r(\boldsymbol{k})$: the total throughput of class r, given population $\boldsymbol{k}$;
$p_i(j \mid \boldsymbol{k})$: the marginal probability of j jobs at station i, given population $\boldsymbol{k}$.

Initialization

- $\bar{k}_{ir}(\mathbf{0}) = 0$ for all $i = 1, \ldots, N$, $r = 1, \ldots, R$.
- $p_i(j \mid \mathbf{0}) = 0$ for all $j = 1, \ldots, (m_i - 1)$.
- $p_i(0 \mid \mathbf{0}) = 1$.

Processing

for $\boldsymbol{k} = \mathbf{0}$ **to** $\boldsymbol{K}$ **do**
  for all stations $i = 1$ **to** N **and all** classes $r = 1$ **to** R **do**

$$\cdot \bar{t}_{ir}(\boldsymbol{k}) = \frac{1}{\mu_{ir} m_i}\left[1 + \sum_{s=1}^{R} \bar{k}_{is}(\boldsymbol{k} - 1_r) + \sum_{j=1}^{m_i - 1} (m_i - j) p_i(j - 1 \mid \boldsymbol{k} - 1_r)\right]$$

$$\cdot \lambda_r(\boldsymbol{k}) = \frac{k_r}{\sum_{i=1}^{N} e_{ir} \bar{t}_{ir}(\boldsymbol{k})}$$

$$\cdot \bar{k}_{ir}(\boldsymbol{k}) = e_{ir} \lambda_r \bar{t}_{ir}(\boldsymbol{k})$$

  for $j = 1$ **to** $m_i - 1$ **do**

$$\cdot p_i(j \mid \boldsymbol{k}) = \frac{1}{j}\left[\sum_{r=1}^{R} x_{ir} \lambda_r(\boldsymbol{k}) p_i(j - 1 \mid \boldsymbol{k} - 1_r)\right]$$

  endfor

$$\cdot p_i(0 \mid \boldsymbol{k}) = 1 - \frac{1}{m_i}\left[\sum_{r=1}^{R} x_{ir} \lambda_r(\boldsymbol{k}) + \sum_{j=1}^{m_i - 1} (m_i - j) p_i(j \mid \boldsymbol{k})\right]$$

  endfor
endfor.

We obtain the final performance measures for the population $\boldsymbol{K} = (K_1, K_2, \ldots, K_R)$, the total number of jobs in the network.

Each iteration of mean value analysis requires that the performance measures for the $(k-1)$st iteration are known. The iterations are carried out from $\mathbf{0}$ to the total jobs in the network, $\boldsymbol{K}$. This makes classical mean value analysis very time consuming.

3. Previous approximations

The *Core* and *Linearizer* methods, as proposed by Chandy and Neuse [4], attempted to approximate the fractional percentage of jobs in each station and the change in this percentage due to the deletion of a particular job from one class. The *Core* algorithm computes the mean number of jobs as follows:

$$\bar{k}_{ir}(\boldsymbol{K}-1_r) = (\boldsymbol{K}-1_r)_r \times (F_{ir}(\boldsymbol{K}) + D_{irs}(\boldsymbol{K})),$$

where

$$F_{ir}(\boldsymbol{K}) = \frac{\bar{k}_{ir}(\boldsymbol{K})}{K_r}$$

represents the fraction of class-r jobs at station i, and

$$D_{irs}(\boldsymbol{K}) = F_{ir}(\boldsymbol{K}-1_s) - F_{ir}(\boldsymbol{K})$$

represents the change in class-r jobs at station i by removing one class s job.

The *Linearizer* algorithm can be viewed as a major advance in approximate mean value analysis. The concept of retreating two steps from the final $\boldsymbol{K}$ population in an effort to improve the $(\boldsymbol{K}-1_r)$ performance measures greatly increases the accuracy of approximate mean value analysis.

The SCAT method proposed by Neuse and Chandy [10] takes this idea of regressing two steps and applies it to multiple server stations. SCAT uses a modified version of the *Core* algorithm that computes simplistic conditional marginal probabilities. The general algorithm is outlined below.

procedure SCAT
- Apply the modified *Core* algorithm at population $\boldsymbol{K}$, with D values of zero
- Apply the modified *Core* algorithm at each of the $(\boldsymbol{K}-1_r)$ populations
- Calculate the F and D values from the above results
- Apply the modified *Core* algorithm using the computed F and D values.

end. {SCAT}

In general, the SCAT algorithm provides good results. The accuracy of these results seems to be due primarily to their improved estimates of the performance measures at the $(\boldsymbol{K}-1_r)$ populations. However, accurate results also depend on the ability to realistically approximate the conditional marginal probabilities $p_i(j \mid \boldsymbol{K}-1_r)$. The conditional marginal probabilities that SCAT computes in its calls to the *Core* algorithm can not be considered realistic.

The SCAT method suggests setting the probability mass very close to the mean number of jobs in the station at that iteration. They assign the complete probability mass to the two neighboring values of the mean number of jobs $\bar{k}_{ir}$. Computation of the conditional marginal probabilities is outlined below:

SCAT Marginal Probability Computation
- compute $floor_{ir}$ and $ceiling_{ir}$, the two integers surrounding $\bar{k}_{ir}(\boldsymbol{K}-1_r)$
- compute the marginal probabilities as follows:

$$p_i(\text{floor}_{ir} \mid \boldsymbol{K}-1_r) = \text{ceiling}_{ir} - \bar{k}_{ir}$$

$$p_i(\text{ceiling}_{ir} \mid \boldsymbol{K}-1_r) = 1 - p_i(\text{floor}_{ir} \mid \boldsymbol{K}-1_r)$$

$$p_i(j \mid \boldsymbol{K}-1_r) = 0 \quad \text{for all other } j$$

end. {Marginal Probability Computation}

If, for example, a value for $\bar{k}_{ir}(\boldsymbol{K})$ of 2.4 is obtained, then the marginal probabilities would be assigned as follows:

$$p_i(2 \mid \boldsymbol{K}-1_r) = 0.6,$$

$$p_i(3 \mid \boldsymbol{K}-1_r) = 0.4,$$

$$p_i(j \mid \boldsymbol{K}-1_r) = 0 \quad \text{for all other } j.$$

This assignment of conditional marginal probabilities is not realistic. An investigation of several networks with multiple servers reveals that in many cases the error margin is quite large (exceeding 30%) It should also be noted here that as Neuse and Chandy used the following formula for computing deviations, their results were not accurately reflected:

$$\text{dev} = \frac{|\text{exact value} - \text{computed value}|}{\text{number of jobs}} \times 100.$$

If the standard formula for relative error is used, then the accidental masking of large errors, as occurred in the original SCAT algorithm, is avoided:

$$\text{dev} = \frac{|\text{exact value} - \text{computed value}|}{\text{exact value}} \times 100.$$

The primary reason for this large deviation is the poor approximation of the conditional marginal probabilities $p_i(j \mid \boldsymbol{k})$. Therefore we introduce an improved approximation for the conditional marginal probabilities.

4. Approximation of mean value analysis

The setting of the probability mass close to the mean number of jobs in the ith station, $\bar{k}_{ir}(\boldsymbol{K} - 1_r)$, fails to fully reflect the fact that these are only the mean values. We cannot ignore the probability of other numbers of jobs existing at that station. Specifically, if $\bar{k}_{ir}(\boldsymbol{K} - 1_r) = 2.6$, then you cannot ignore the probability of 0, 1, 4, 5,..., K_r jobs in that station.

We find this especially true if the value for $\bar{k}_{ir}(\boldsymbol{K} - 1_r)$ is approximately above m_i, the number of servers at that station. In this case, all the $p_i(j \mid K - 1_r)$ values for $j = 1, \ldots, (m_i - 2)$ are zero under the SCAT algorithm. The computation of $\bar{t}_{ir}(\boldsymbol{K})$ is seriously degraded, as all of the conditional marginal probabilities are essentially ignored.

Therefore, our algorithm seeks a method of scattering the probability mass over a wider range of numbers, thus increasing its standard deviation. Specifically, we note that this range should encompass the entire range of values $0, \ldots, \bar{k}_i(\boldsymbol{K} - 1_r), \ldots, (2\lfloor \bar{k}_{ir}(\boldsymbol{K}) \rfloor + 1)$. This is denoted in Fig. 1.

In order to accomplish this, we must first identify some weight function which will provide a normal distribution of the probability values. This is done using the two formulas shown below. The PR function simply scales down the distribution probabilities, while the $W[i, j]$ array represents the calculated weighting function for a distribution over pairs of numbers, with j indicating the relative distance of a specific pair to the mean. This function will attempt to mimic a normal probability distribution.

Calculation of Conditional Marginal Probabilities

- calculate a scaling function PR for all $n = 1, \ldots, \max(K_r)$

$$\text{PR}[1] = \alpha \tag{1a}$$

$$\text{PR}[n] = \beta\ \text{PR}[n-1] \tag{1b}$$

- calculate a weight function W for all values $i = 1, \ldots, \max(K_r)$

$$W[0, 0] = 1 \tag{2a}$$

$$W[i, j] = W[i-1, j] - \frac{W[i-1, j]\ \text{PR}[i]}{100} \quad \text{for } j = 1, \ldots, (i-1) \tag{2b}$$

$$W[i, i] = 1 - \sum_{j=0}^{i-1} W[i, j] \tag{2c}$$

end.

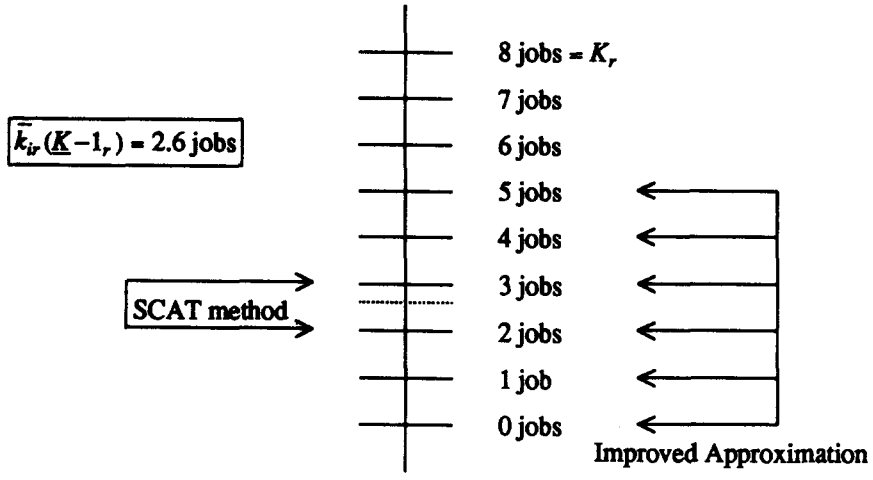

Fig. 1.

Table 1 lists the values for the above weight function for a distribution over five pairs of numbers. Our study has shown that the algorithm operates optimally using an α value of 45 and a β value of 0.7.

When assigning conditional marginal probabilities, we first define three values, floor, ceiling, and maxval. Floor and ceiling are simply the two integers surrounding $\bar{k}_{ir}(\boldsymbol{K}-1_r)$, while maxval is the width of the probability mass distribution function:

$$\text{floor}_{ir} = \lfloor \bar{k}_{ir}(\boldsymbol{K}-1_r) \rfloor, \tag{3a}$$

$$\text{ceiling}_{ir} = \text{floor}_{ir} + 1, \tag{3b}$$

$$\text{maxval}_{ir} = \min\{2\ \text{floor}_{ir} + 1;\ m_i\}. \tag{3c}$$

We then partition the probability mass into $(\text{floor}_{ir}+1)$ pairs of numbers. The sum of each of these pairs will have the combined probability mass of $W[i, j]$, as shown in Fig. 2. The conditional marginal probability of j jobs in station i is computed as:

$$p_i(j \text{ jobs} \mid \boldsymbol{K}-1_r) = (\text{weight function}) \times (\text{relative proximity of } \bar{k}_{ir}(\boldsymbol{K}-1_r)).$$

Mathematically, this is represented by:

$$p_i(\text{floor}_{ir} - j \mid \boldsymbol{K}-1_r) = W[\text{floor}_{ir},\ l\_\text{dist}]\,\frac{(\text{upperval} - \bar{k}_{ir}(\boldsymbol{K}-1_r))}{\text{upperval} - \text{lowerval}} \tag{4a}$$

for all numbers of jobs $j = 0, \ldots, \text{floor}_{ir}$, where

l_dist represents the distance from j to floor_{ir},

upperval represents the value defined by $(\text{ceiling}_{ir} + l\_\text{dist})$,

lowerval represents the value defined by $(\text{floor}_{ir} - l\_\text{dist})$.

Table 1

n	$W[n,0]$	$W[n,1]$	$W[n,2]$	$W[n,3]$	$W[n,4]$	$W[n,5]$
0	1.0					
1	0.55	0.45				
2	0.377	0.308	0.315			
3	0.294	0.240	0.245	0.221		
4	0.248	0.203	0.208	0.187	0.154	
5	0.222	0.181	0.185	0.166	0.138	0.108

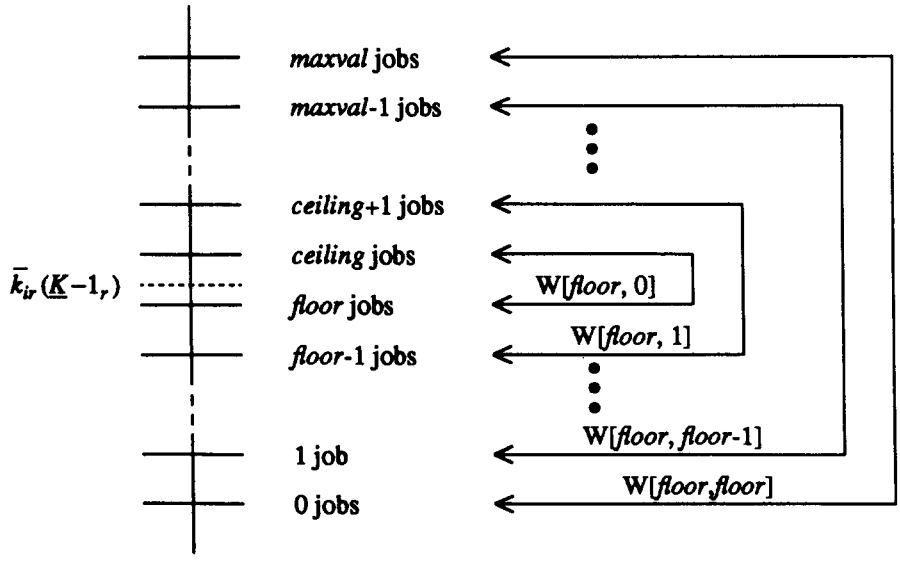

Fig. 2.

The conditional marginal probability for all jobs $j = \text{ceiling}_{ir}, \ldots, \text{maxval}_{ir}$ is computed by

$$p_i(j \mid \boldsymbol{K} - 1_r) = W[\text{floor}_{ir},\ u\_\text{dist}] - p_i(\text{floor}_{ir} - u\_\text{dist} \mid \boldsymbol{K} - 1_r), \tag{4b}$$

where u_dist represents the value of $(j - \text{ceiling}_{ir})$. Finally, set

$$p_i(j \mid \boldsymbol{K} - 1_r) = 0 \quad \text{for } j > \text{maxval}_{ir}. \tag{4c}$$

This approximation makes the assumption that $\bar{k}_{ir}(\boldsymbol{K} - 1_r)$ is always less than half of K_r, as shown in Fig. 3(a). If the value for $\bar{k}_{ir}(\boldsymbol{K} - 1_r)$ exceeds $\{(K_r - 1)/2\}$, as shown in Fig. 3(b), then the value for upperval in the above equations can exceed the allowable range of $0, \ldots, (K_r - 1)$.

In order to prevent assigning values outside the legal range, the value computed for upperval is checked to ensure that its range never exceeds the maximum allowed by $(K_r - 1)$. Furthermore, the algorithm will assign all probabilities for values exceeding $(K_r - 1)$ to the point $(K_r - 1)$. This maintains the integrity of the probability mass function, and does not harm the performance measures. This noninterference is due to the fact that the number of servers in station i will realistically never approach K_r in situations where approximate mean value analysis is being used.

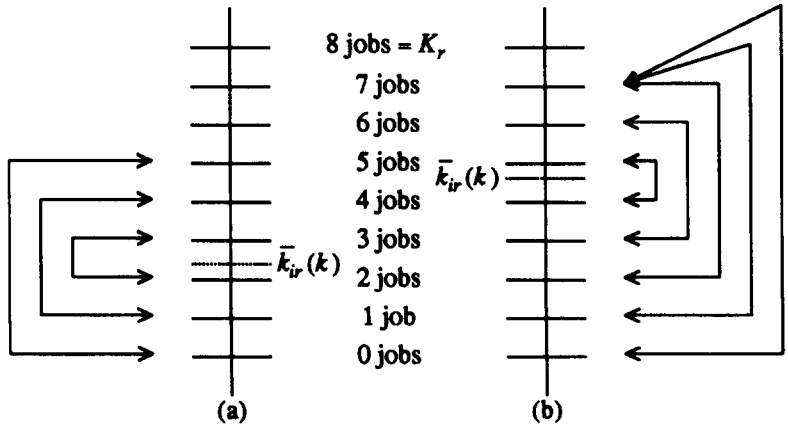

Fig. 3.

Finally, when a distance of 25 or greater is reached form $\bar{k}_{ir}(\boldsymbol{K}-1_r)$ to the current j value, the weight function $W[25, 25]$ has a value of less than 0.0001. Thus computing the weight function or conditional marginal probabilities for values whose distance from the mean number of jobs $\bar{k}_{ir}(\boldsymbol{K}-1_r)$ exceeds 25 is unnecessary.

5. Algorithm summary

The following is a summary of the algorithm used to compute the marginal probabilities.

procedure FindMarginalProbs
begin
- compute the weight function W using (1) and (2)
- compute floor_{ir}, ceiling_{ir}, and maxval_{ir} using (3a), (3b), and (3c)

for $j = 0$ **to** maxval_{ir} **do**
  if $j \leqslant \text{floor}_{ir}$ **then**
    - compute l_dist, upperval, and lowerval as per (4a)

    if upperval $> (K_r - 1)$ **then** upperval $= (\boldsymbol{K}_r - 1)$
    if $l\_\text{dist} > 25$ **then** {too far from k_{ir}}
      - $p(j \mid \boldsymbol{K} - 1_r) = 0.0$

    else
      - compute $p(j \mid \boldsymbol{K} - 1_r)$ using (4a)

    endif
  else
    - compute u_dist as per (4b)

    if $u\_\text{dist} > 25$ **then** {too far from $\bar{k}_{ir}$}
      - $p(j \mid \boldsymbol{K} - 1_r) = 0.0$

    elseif $j > (K_r - 1)$ **and** $u\_\text{dist} < 25$ **then** {exceed range, add to $(K_r - 1)$}
      - $p(K_r - 1 \mid \boldsymbol{K} - 1_r) = p(K_r - 1 \mid \boldsymbol{K} - 1_r)$
        $+ (W[\text{floor}, u\_\text{dist}] - p(\text{floor} - u\_\text{dist} \mid \boldsymbol{K} - 1_r))$

    else
      - compute $p(j \mid \boldsymbol{K} - 1_r)$ using (4b)

    endif
  endif
endfor
end.

6. Sample algorithm execution

Consider the following network. There are $N = 4$ stations and $R = 2$ classes in the network. Class one contains $K_1 = 20$ jobs and class two contains $K_2 = 20$ jobs. The number of servers and service disciplines for each station are outlined in Table 2.

Table 2

Station	1	2	3	4
Number of servers	5	4	5	6

Table 3

Station	Class	Mean service rate	Transition probabilities
1	1	$\mu_{11} = 1.0\ s^{-1}$	$p_{12} = 0.5;\ p_{13} = 0.25;\ p_{14} = 0.25$
1	2	$\mu_{12} = 2.0\ s^{-1}$	$p_{1j} = 0.25$ for all j
2	1	$\mu_{21} = 0.66667\ s^{-1}$	$p_{21} = 1.0$
2	2	$\mu_{22} = 1.0\ s^{-1}$	$p_{2j} = 0.25$ for all j
3	1	$\mu_{31} = 0.8\ s^{-1}$	$p_{31} = 1.0$
3	2	$\mu_{32} = 0.8\ s^{-1}$	$p_{3j} = 0.25$ for all j
4	1	$\mu_{41} = 0.66667\ s^{-1}$	$p_{41} = 1.0$
4	2	$\mu_{42} = 1.25\ s^{-1}$	$p_{4j} = 0.25$ for all j

The mean service times and transition probabilities are listed in Table 3.

In Tables 4 and 5 the exact results obtained by classical mean value analysis are given, as well as the relative errors for both our approximation and the SCAT method.

As can easily be seen from Tables 4 and 5, our approximation provides superior results to that of the SCAT algorithm. The average deviation for our approximation was 1.28%, while SCAT showed an average deviation for 11.52%. Table 6 compares the actual conditional marginal probabilities for the two $(\boldsymbol{K} - 1_r)$ populations with those obtained by both SCAT and our approach. The results are shown for $(0, \ldots, (m_i - 2))$ jobs in each station, as those are the values used in future iterations.

Table 4
Mean number of jobs $\bar{k}_{ir}(\boldsymbol{K})$

Station	Class	Classical MVA	Our method	Deviation	SCAT	Deviation
1	1	3.4683	3.5247	1.63%	2.7069	21.95%
1	2	1.3109	1.3300	1.46%	0.9844	24.91%
2	1	14.5049	14.4940	0.08%	15.4321	6.39%
2	2	14.1935	14.2255	0.23%	14.9798	5.54%
3	1	1.0000	0.9740	2.60%	0.8459	15.41%
3	2	2.8871	2.8589	0.98%	2.4609	14.76%
4	1	1.0267	1.0073	1.89%	1.0151	1.13%
4	2	1.6084	1.5856	1.42%	1.5750	2.08

Table 5
Mean residence time $\bar{t}_{ir}(\boldsymbol{K})$

Station	Class	Classical MVA	Our method	Deviation	SCAT	Deviation
1	1	1.2876	1.3122	1.91%	1.0000	22.34%
1	2	0.6622	0.6711	1.34%	0.5000	24.49%
2	1	10.7700	10.7915	0.20%	11.4020	5.87%
2	2	7.1697	7.1774	0.11%	7.6090	6.13%
3	1	1.4850	1.4503	2.34%	1.2500	15.82%
3	2	1.4584	1.4424	1.10%	1.2500	14.29%
4	1	1.5247	1.5000	1.62%	1.5000	1.62%
4	2	0.8125	0.8000	1.54%	0.8000	1.54%

Table 6
Conditional marginal probabilities

		$K = (19, 20)$			$K = (20, 19)$		
Station	Number of jobs	Classical MVA	SCAT	Our method	Classical MVA	SCAT	Our method
1	0	0.0206	0.0000	0.0732	0.0185	0.0000	0.0695
1	1	0.0771	0.0000	0.0870	0.0707	0.0000	0.0813
1	2	0.1433	0.0000	0.0941	0.1343	0.0000	0.0853
1	3	0.1765	0.2687	0.0858	0.1693	0.2687	0.0714
2	0	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2	1	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2	2	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
3	0	0.0297	0.0000	0.0981	0.0327	0.0000	0.1018
3	1	0.1021	0.0000	0.1039	0.1099	0.0000	0.1096
3	2	0.1746	0.0000	0.0893	0.1833	0.0000	0.0986
3	3	0.1979	0.6540	0.0339	0.2025	0.7394	0.0679
4	0	0.0728	0.0000	0.1509	0.0750	0.0000	0.1528
4	1	0.1905	0.0000	0.1433	0.1941	0.0000	0.1464
4	2	0.2490	0.3970	0.1487	0.2508	0.4278	0.1600
4	3	0.2167	0.6030	0.2281	0.2158	0.5722	0.2167
4	4	0.1422	0.0000	0.1649	0.1391	0.0000	0.1619

7. Evaluation

We have derived an improved algorithm for approximate mean value analysis of multiple-class multiple-server queueing networks. It provides a reliable method for determining performance measures in systems were classical mean value analysis cannot be utilized. Our method incorporates the improved accuracy of the SCAT algorithm by incorporating SCAT's approximation for the performance measures at the $(K - 1_r)$ populations. It then refines this accuracy by providing a more realistic computation of the conditional marginal probabilities than SCAT provides.

The algorithm has been implemented and analysis was conducted for over 75 different sample networks. The number of classes for these networks ranged from one to five. The total number of jobs in the system varied from five to 200. The number of stations ranged between two and ten. Multiple-server stations were tested with up to eight servers at one specific station. Numerous stress tests were applied, and in no instance did our enhancement demonstrate any major drop in accuracy. In all cases, our enhancement showed deviations that did not exceed twelve percent. Table 7 outlines the minimum, maximum, and average deviations for our enhancement as well as the SCAT results for all test cases.

The run time and storage utilization for our algorithm have the same bounds as that of SCAT. The run time for one iteration of our algorithm depends only on the number of classes R, the number of stations N, the number of multiple server stations Z, and the maximum number of servers in any station $(\max(m_i))$. The storage requirements depend only on the number of classes R, the number of stations N, and the extent of the weight function W, which is bounded by $(\max(m_i))$. The run time and storage requirements for our approximation, SCAT, and classical mean value analysis are detailed in Tables 8 and 9.

Table 7

Method	Minimum deviation	Maximum deviation	Average deviation
Our method	0.01%	11.73%	1.887%
SCAT	0.21%	26.90%	7.703%

Table 8
Algorithmic time complexity

Our approximation	SCAT	MVA
$O(NR^2 + ZR \max(m_i))$	$O(NR^2)$	$O\left(m_i R \prod_{r=1}^{R} (K_r+1)\right)$

Table 9
Algorithmic space complexity

Our approximation	SCAT	MVA
$O(NR^2 + \max(m_i)^2)$	$O(NR^2)$	$O\left(N \prod_{r=1}^{R} (K_r+1)\right)$

While our method provides more accurate results when dealing with multiple-server stations, it should be pointed out that in cases where only single-server stations are used, the result do not differ from those obtained by SCAT. SCAT and the Linearizer algorithm are both known to provide excellent results in the case of single-server stations. This is simply due to the fact that the calculations do not incorporate marginal probabilities for any values other than $(0,\ldots,(m_i-2))$. In other words, the marginal probabilities are ignored in the case of single-server stations.

Appendix A. Sample test cases

A.1. Example. The following system, consisting of $R = 5$ job classes and 100 jobs, is examined. Each class contains twenty jobs. The number of servers in each stations is outlined in Table A.1.

The average deviation from the exact results is 1.44% for our approximation, 9.48% for SCAT. For results, see Tables A.2–A.4.

A.2. Example. A system with $R = 2$ classes and $N = 3$ stations is examined. There are 75 jobs in class one and 50 in class two.

The average deviation is 1.58% for our method and 8.95% for SCAT. For results see Tables A.6 and A.7.

A.3. Example. A system with $R = 2$ classes and $N = 2$ stations is analyzed. There are 100 jobs in class one and 100 in class two.

The average deviation is 0.43% for our method and 13.85% for SCAT. For results see Tables A.9 and A.10.

Table A.1

Station	1	2	3	4
Number of servers	5	4	5	3

Table A.2

Station	Class	Mean service rate	Probabilities
1	1	$\mu_{11} = 2.0\ s^{-1}$	$p_{1j} = 0.25$ for all j
1	2	$\mu_{12} = 1.0\ s^{-1}$	$p_{12} = 0.5;\ p_{13} = p_{14} = 0.25$
1	3	$\mu_{13} = 1.0\ s^{-1}$	$p_{12} = p_{13} = p_{14} = 0.33333$
1	4	$\mu_{14} = 1.3333\ s^{-1}$	$p_{12} = p_{13} = p_{14} = 0.33333$
1	5	$\mu_{15} = 1.25\ s^{-1}$	$p_{12} = p_{13} = p_{14} = 0.33333$
2	1	$\mu_{21} = 1.0\ s^{-1}$	$p_{2j} = 0.25$ for all j
2	2	$\mu_{22} = 0.8\ s^{-2}$	$p_{21} = 1.0$
2	3	$\mu_{23} = 0.66667\ s^{-1}$	$p_{21} = p_{23} = p_{24} = 0.33333$
2	4	$\mu_{24} = 1.0\ s^{-1}$	$p_{21} = 1.0$
2	5	$\mu_{25} = 0.8\ s^{-1}$	$p_{21} = p_{24} = 0.5$
3	1	$\mu_{31} = 1.0\ s^{-1}$	$p_{3j} = 0.25$ for all j
3	2	$\mu_{32} = 1.0\ s^{-1}$	$p_{31} = 1.0$
3	3	$\mu_{33} = 1.0\ s^{-1}$	$p_{31} = p_{32} = p_{34} = 0.33333$
3	4	$\mu_{34} = 1.0\ s^{-1}$	$p_{31} = 1.0$
3	5	$\mu_{35} = 1.0\ s^{-1}$	$p_{31} = p_{32} = p_{34} = 0.33333$
4	1	$\mu_{41} = 0.5\ s^{-1}$	$p_{4j} = 0.25$ for all j
4	2	$\mu_{42} = 0.66667\ s^{-1}$	$p_{41} = 1.0$
4	3	$\mu_{43} = 0.8\ s^{-1}$	$p_{41} = p_{42} = p_{43} = 0.33333$
4	4	$\mu_{44} = 0.66667\ s^{-1}$	$p_{41} = 1.0$
4	5	$\mu_{45} = 0.5\ s^{-1}$	$p_{41} = 1.0$

Table A.3
Mean residence time $\tilde{t}_{ir}(\boldsymbol{K})$

Station	Class	Classical MVA	Our method	Deviation	SCAT	Deviation
1	1	0.6071	0.6066	0.08%	0.5000	17.64%
1	2	1.1179	1.1586	1.64%	1.0000	15.10%
1	3	1.1901	1.1658	2.04%	1.0000	15.97%
1	4	0.8970	0.8705	2.95%	0.7500	16.39%
1	5	0.9599	0.9677	0.81%	0.8000	16.66%
2	1	1.2417	1.2641	1.80%	1.0000	19.47%
2	2	1.5563	1.5731	1.08%	1.2500	19.68%
2	3	1.8675	1.8824	0.80%	1.5000	19.68%
2	4	1.2456	1.2643	1.50%	1.0000	19.72%
2	5	1.5502	1.5816	2.03:	1.2500	19.37%
3	1	1.0098	0.9800	2.95%	1.0000	0.97%
3	2	1.0081	0.9800	2.79%	1.0000	0.80%
3	3	1.0102	0.9800	2.99%	1.0000	1.01%
3	4	1.0158	0.9800	3.52%	1.0000	1.56%
3	5	0.9950	0.9800	1.51%	1.0000	0.50%
4	1	60.3182	60.6568	0.56%	61.4954	1.95%
4	2	45.4091	45.5952	0.41%	46.1764	1.69%
4	3	37.9099	37.9521	0.11%	38.4549	1.44%
4	4	45.4564	45.5611	0.23%	46.1461	1.52%
4	5	60.7932	60.6643	0.21%	61.4957	1.16%

Table A.4
Mean number of jobs $\bar{k}_{ir}(\boldsymbol{K})$

Station	Class	Classical MVA	Our method	Deviation	SCAT	Deviation
1	1	0.1964	0.1910	2.75%	0.1563	20.42%
1	2	1.7213	1.7053	0.93%	1.4904	13.41%
1	3	0.5643	0.5554	1.58%	0.4767	15.52%
1	4	1.0661	1.0360	2.82%	0.8928	16.25%
1	5	0.4509	0.4560	1.13%	0.3748	16.88%
2	1	0.3979	0.3981	0.05%	0.3125	21.46%
2	2	1.1337	1.1576	2.11%	0.9315	17.84%
2	3	0.8923	0.8968	0.50%	0.7150	19.87%
2	4	0.4947	0.5016	1.39%	0.3968	19.79%
2	5	0.3258	0.3313	1.69%	0.2603	20.10%
3	1	0.3221	0.3086	4.16%	0.3125	2.98%
3	2	0.3662	0.3606	1.53%	0.3726	1.75%
3	3	0.4819	0.4669	3.11%	0.4767	1.08%
3	4	0.4007	0.3888	2.97%	0.3968	0.97%
3	5	0.1533	0.1539	0.39%	0.1562	1.89%
4	1	19.0865	19.1022	0.08%	19.2187	0.69%
4	2	16.7788	16.7765	0.01%	17.2055	2.54%
4	3	18.0615	18.0809	0.11%	18.3316	1.50%
4	4	18.0385	18.0737	0.20%	18.3134	1.52%
4	5	19.0700	19.0588	0.06%	19.2087	0.73%

Table A.5

Station	Servers	Class	Mean service rate	Probabilities
1	1	1	$\mu_{11} = 2.0\ s^{-1}$	$p_{12} = p_{13} = 0.5$
1	1	2	$\mu_{12} = 0.66667\ s^{-1}$	$p_{12} = 0.7;\ p_{13} = 0.3$
2	2	1	$\mu_{21} = 0.25\ s^{-1}$	$p_{21} = 1.0$
2	2	2	$\mu_{22} = 1.25\ s^{-1}$	$p_{21} = 0.8;\ p_{23} = 0.2$
3	3	1	$\mu_{31} = 0.2\ s^{-1}$	$p_{31} = 1.0$
3	3	2	$\mu_{32} = 1.3333\ s^{-1}$	$p_{31} = 1.0$

Acknowledgment

We are grateful to the reviewers for their suggestions and constructive criticisms. Special thanks are due to Dr. Martin Reiser for his invaluable advice on the manuscript.

Table A.6
Mean residence time $\bar{t}_{ir}(\boldsymbol{K})$

Station	Class	Classical MVA	Our method	Deviation	SCAT	Deviation
1	1	42.5967	42.3822	0.50%	42.3483	0.58%
1	2	124.5160	124.1247	0.31%	124.0078	0.41%
2	1	72.3141	73.5022	1.64%	75.4397	4.32%
2	2	15.3304	15.4953	1.08%	15.8877	3.64%
3	1	9.8579	9.5552	3.07%	7.6393	22.51%
3	2	1.4820	1.4397	2.85%	1.1522	22.25%

Table A.7
Mean number of jobs $\bar{k}_{ir}(\boldsymbol{K})$

Station	Class	Classical MVA	Our method	Deviation	SCAT	Deviation
1	1	38.1769	37.8814	0.77%	37.8615	0.83%
1	2	45.8118	45.7671	0.10%	45.7134	0.21%
2	1	32.4055	32.8483	1.37%	33.7235	4.07%
2	2	3.9483	3.9994	1.29%	4.0997	3.83%
3	1	4.4176	4.2702	3.34%	3.4150	22.70%
3	2	0.2399	0.2336	2.63%	0.1869	22.09

Table A.8

Station	Servers	Class	Mean service rate	Probilities
1	3	1	$\mu_{11} = 0.1\ s^{-1}$	$p_{11} = p_{12} = 0.5$
1	3	2	$\mu_{12} = 0.08333\ s^{-1}$	$p_{12} = 1.0$
2	4	1	$\mu_{21} = 0.06667\ s^{-1}$	$p_{21} = p_{22} = 0.5$
2	4	2	$\mu_{22} = 0.125\ s^{-1}$	$p_{21} = 1.0$

Table A.9
Mean residence time $\bar{t}_{ir}(\boldsymbol{K})$

Station	Class	Classical MVA	Our method	Deviation	SCAT	Deviation
1	1	647.5947	647.7606	0.03%	652.7501	0.80%
1	2	776.7732	776.9627	0.02%	782.9391	0.79%
2	1	26.6294	26.4012	0.86%	19.4061	27.13%
2	2	14.3592	14.2411	0.82%	10.5306	26.66%

Table A.10
Mean number of jobs $\bar{k}_{ir}(\boldsymbol{K})$

Station	Class	Classical MVA	Our method	Deviation	SCAT	Deviation
1	1	96.0504	96.0838	0.03%	97.1129	1.11%
1	2	98.1849	98.2001	0.02%	98.6728	0.50%
2	1	3.9496	3.9162	0.85%	2.8871	26.90%
2	2	1.8150	1.7999	0.83%	1.3272	26.88%

References

[1] I.F. Akyildiz, Mean value analysis for closed queueing networks with Erlang service time distributions, *The Comput. J.* (1988) to be published.

[2] Y. Bard, Some extensions to multiclass queueing network analysis, in: *4th Internat. Symp. on Modeling and Performance Evaluation of Computer Systems, Vol. 1,* Vienna, 1979.

[3] S.C. Bruell, G. Balbo and P.V. Afshari, Mean value analysis of mixed, multiple class BCMP networks with load dependent-service stations, *Performance Evaluation* **4** (1984) 241–260.

[4] K.M. Chandy and D. Neuse, Linearizer: A heuristic algorithm for queueing network models of computing systems, *Comm. ACM* **25** (2) (1982) 126–134.

[5] R. Heinselman, *Heuristic Iterative Mean Value Analysis: Evaluation of Some Issues,* Rept. TM-49-9, Sperry Research Center, Sudbury, MA, 1983.

[6] K.P. Hoyme, S.C. Bruell, P.V. Afshari and R.Y. Kain, A tree-structured mean value analysis algorithm, *ACM Trans. Comput. Systems* **4** (2) (1986) 1178–185.

[7] A.E. Krzesinski and A. Greyling, Improved linearizer methods for queueing networks with queue dependent centers, in: *ACM SIGMETRICS Conf. Proc.*, Cambridge, MA (1984) 41–51.

[8] S.S. Lavenberg and M. Reiser, Stationary state probabilities at arrival instants for closed queueing networks with multiple types of customers, *J. Appl. Probab.* **17** (1980) 1048–1061.

[9] J.D.C. Little, A proof for the queueing formula $L = \lambda W$, *Oper. Res.* **9** (3) (1961) 383–387.

[10] D. Neuse and K.M. Chandy, SCAT: A heuristic algorithm for queueing network models of computing systems, in: *ACM SIGMETRICS Conf. Proc.* **10** (3) (1981) 59–79.

[11] M. Reiser, Mean value analysis of queueing networks: A new look at an old problem, in: *4th Internat. Symp. on Modeling and Performance Evaluation of Computer Systems, Vol. I,* 1979.

[12] M. Reiser, A queueing network analysis of computer communication networks with window flow control, *IEEE Trans. Comm.* **COM-27** (1979) 1199–1209.

[13] M. Reiser and S.S. Lavenberg, Mean value analysis of closed multichain queueing networks, *J. ACM* **27** (2) (1980) 313–322.

[14] M. Reiser, Mean value analysis and convolution method for queueing dependent servers in closed queueing networks, *Performance Evaluation* **1** (1) (1981) 7–18.

[15] P. Schweitzer, Approximate analysis of multiclass closed networks of queues, in: *Internat. Conf. on Stochastic Control and Optimization*, Amsterdam, 1979.

[16] K.C. Sevcik and I. Mitrani, The distribution of queueing network states at input and output instances, *J. ACM* **28** (2) (1981) 358–371.

[17] J.C. Strelen, A generalization of mean value analysis to higher moments: Moment analysis, in: *Proc. Performance and ACM SIGMETRICS 86 Joint Conf.* (1986) 129–140.

[18] J. Zahorjan and E. Wong, A solution of separable queueing network models using mean value analysis, in: *ACM SIGMETRICS Conf.* **10** (3) (1981) 80–85.

[19] J. Zahorjan and E.D. Lazowska, Incorporating load dependent servers in approximate mean value analysis, in: *ACM SIGMETRICS Conf.* **12** (3) (1984) 52–62.

References added in proof

[20] C.H. Sauer, Computational algorithms for state-dependent queueing networks, *ACM Trans. Comput.* **1** (1) (1983) 67–92.

[21] S. Tucci and C.H. Sauer, The tree MVA algorithm, *Performance Evaluation* **5** (3) (1985) 187–196.