

Research Article

Reduction of computational complexity of mean value analysis in modeling of multiple class-server stations for optimal performance

A. D. Gbadebo^{a*}, S. A. Akinleye^b, A. A. Adedokun^a & Y. A. Odunayo^a

^aDepartment of Computer Science Education, Lagos State University of Education, Oto/Ijanikin, Nigeria

^bDepartment of Mathematics, Federal University of Agriculture, Abeokuta, Nigeria

Abstract

The evaluation of stochastic systems results in product-form queue networks with multiple customers' class-server stations. As a result of the exponential computational complexity involved in the number of classes, it is impossible to evaluate this model efficiently using the mean value analysis (MVA) algorithm. A number of approximation methods had been applied to minimize corresponding problems of computational complexity but none had sufficiently solved the problem involving multiple servers. We propose a new approach, the Multiple Server-Station Expected Performance Measure (MSEPM) to reduce the computational complexity involved in the use of the mean value analysis for the performance evaluation of multi-classes, multi-server stations using the first-come, first-served policy. The Arena training and evaluation mode (student) simulator was used to test the viability of MSEPM. Numerical results were compared with the results generated using exact MVA algorithm for the same experiments. Results indicated an improved performance of MSEPM over the exact MVA algorithm.

Keywords: Heuristic approximations, iterative methods, Markov Decision Process, product-form queue networks.

1. Introduction

The performance evaluation of any queue system is premised on the analysis of its stochastic models. Queue network models have been applied in the representation and analysis of resource sharing systems, e.g. communication and production parameters (Calvo & Artega, 2024). The arrival theorem is an important feature of queue systems with product-form. Samouylov, Dudina and Dudin (2023) describes this as distribution of arrivals' patterns at service station which is identical to its distribution at arbitrary times of the same system for open networks and a system with one less arrival for closed systems. As long as a closed queue system meets the product-form requirement and structure, the exact analysis method could be adopted to measure it (Chaves & Gosavi, 2022).

Performance measures' computation requires the derivation of normalization constants for the specific network, which requires computational efforts as it involves enormous number of states and parameters (Baron, et al (2022). Consequently, more robust computational approaches, including the convolution algorithm, the mean value analysis, among others had been proposed to overcome

this bottleneck. There are two main bottlenecks connected with the use of the convolution algorithm, these are: firstly, numerical problems occur when the network involves the applications of large stations with classes of customers; and secondly, the method does not accommodate heuristic extensions when more general models of systems are involved (Amjath, Kerbach & Elomri, et al. 2024). However, the mean value analysis is more often applied to derive systems' performance features because these problems are not known in this method.

It is possible to derive the exact solutions of a BCMP (Baskett, Chandy, Muntz and Palacois) system using the exact MVA algorithm, being a class of system network involving a product-form equilibrium distribution. However, computational costs might be prohibitive for system models not restricted to a few customers in a class (Legato and Mazza, 2017). This is so as the recursive MVA method is characteristically dependent on variables being updated at the current system iteration as well as others that had been derived at previous iterations. In this case, if N_b denotes the amount of class b customers while

*Corresponding author: A. D. Gbadebo

Email address: gbadeboad@lasued.edu.ng (A. D. Gbadebo)

ORCID: <https://orcid.org/0009000782831762>

Received: 14/06/2026; Accepted: 30/07/2026; Published: 31/07/2026



$N = (N_1, \dots, N_b, \dots, N_B)$ denotes the population of the network, then it is necessary to repeat the iteration from $(0_1, \dots, 0_B)_{t_0}$ to $(N_1, \dots, N_B)$.

The MVA is often described as a vibrant iterative model used to derive average performance metrics, including systems' response time, queue length, throughput, etc of product-form queue systems (Efrosinin, et al, 2025). Most often, it is applicable in modelling systems and servers' stations as it is devoid of the computational costs of deriving normalization constants needed in earlier convolutional approaches (Baron, et al 2024). In most multiple class queue systems, the complexity involved in the computation of the MVA is often exponential regarding the number of variables as well as number of iterations (Suri, Sahu & Vernon, 2007). For systems involving limited service stations and multiple customers' classes, the Mean Value Analysis by Class (MVAc), a variant of the MVA can be applied. This variant exhibits a much lower computational costs compared to MVA. However, as the number of classes rises, it becomes more costly, thereby making the method unsuitable for applications in systems involving multiple classes (Sherzer, 2024).

The management of queue systems with multiple class and servers' stations requires computational complexities which are often expensive (Sherzer, et al 2024). The introduction of modern techniques to reduce this complexity had become necessary in order to ensure optimality in throughput and cost reduction of network resources, including queue lengths (Ni, Okudel & Wakita (2020). This is why some authors had proposed the MVA approach and its variants but this had become inadequate with regards to high consumption of network resources including servers' and customers' times, resulting in increase in network costs, among others (Whitt & You 2022, Tan & Khayyati 2022).

The need for this study was borne out of the necessity to manage this complexity to a level such that servers' operational costs, customers' wait times and similar network parameters are minimized while optimizing systems' throughput. This is necessary in order to minimize operational costs while optimizing system potentials and throughputs. In the 21st century manufacturing systems for instance, minimization of operational costs is inevitable as a result of rising inflations, economic uncertainties and productions management.

2. Related studies

Agboola (2021) in a study involving the use of mean value analysis devoid of normalization constant in closed queuing systems considered the use of multiple node system in which a customer demands service at multiple nodes. The author perceived this as network system in which each node serves as a service centre for queues to be formed, with multiple servers to provide needed services. This method provides insight into performance measure analysis of the

system such that the normalization constant is exempted in the computation of system's performance measure. Dudin, et al (2024) in a study on the analysis of tandem queue with multi-server stages considered a tandem two-queue system made of multiple server stages. While stage 1 has infinite buffer size with exponential service distribution, stage 2 has finite buffer size with a system of phase-type service distribution. When the buffer size is full, arrivals at this time are dropped or transmitted to stage 1 for service. With this approach, arrivals are defined using a Markov process, thereby giving rise to a multi-dimensional continuous-time Markov chain.

In Sherzer (2025), the author used a machine learning method to forecast the stationary distribution functions of arrivals in a mono-station, multi-server system. Two systems were considered including homogeneous service system i.e. the $G1/G1/c$ queue network as well as dual-heterogeneous service system, i.e. the $G1/G1/2$ queue network. A neural network was trained for these queue models using inter-arrival and service times. Previously, Wang and Sevcik (1998) modeled a package named Improved Approximate Mean Value Analysis Library (IAMVAL), which could be integrated into existing commercial and research queue system analysis packages. The IAMVAL has two new approximate Mean Value Analysis (MVA) algorithms, i.e. the Queue Line (QL) and the Fraction Line (FL) algorithms for the purpose of analyzing multiple-class separable-queue networks. Although the QL algorithm is more exact but has approximately the same level of computational success as, the Bard-Schweitzer Proportional Estimation algorithm, which is the most widely used approximate MVA algorithm. However, FL algorithm possesses the same level of computational requirements. In non-congested separable-queue networks with minimal queue lengths, this algorithm produces more reliable solutions compared to QL algorithms.

Petriu (1994) in a study on approximate mean value analysis of client-server systems with multi-class requests, applied the Task-Directed Aggregation (TDA) approach to model an algorithm which allows a client to request unlimited services from a service station by randomly changing the class of request. This algorithm was included into a decomposition approach for models with infinite servers. The author applied the Stochastic Rendezvous Networks (SRVN) modeling system to a large distributed database which gave insight to system peculiarities while assisting to manage network performance issues. Related to this, Samouylov, et al (2023) in a study on the analysis of multi-server queue system with flexible priority system, considered a multi-server queue system providing services to a dual correlated arrival flow. In this study, a priority of the non-preemptive form was given to an arrival by means of preliminary delay in intermediate buffers having different extraction rates.

Wang, et al (2023) in a study involving estimation of

multi-class service demand distributions using Markovian arrival processes proposed estimating higher moment of a service demand distribution for use in a micro-service system. The authors generated a closed queue model to abstract software efficiency and apply it to design departure process of arrivals as a Markovian arrival process (MAP). With this, it was possible to formulate the approximation of service requirement as an optimization problem targeted at deriving the first moments of service demand which maximizes the probabilistic form of MAP using inter-departure times.

Varki (2021) proposed a technique for deriving the mean performance of closed single-class fork-join systems with exponential distribution service system. The method is similar to the mean value analysis approach for closed product-form systems. Mean performance of the system including queue lengths, mean response time as well as throughput for closed fork-join systems could be recursively derived without the steady-state distribution. The approach is premised on the mean value derivation for fork-join networks that relates response time to the mean service times for different servers as well as the mean queue length of the network.

Conway, et al (1997) in a study on mean value analysis by chain for product-form queue networks formulated a computational algorithm applicable for closed multi-chain product-form queue networks. It applies the mean performance measures for networks consisting of single-server service rate and infinite-server systems. The algorithm, i.e. mean value analysis by chain (MVAc) was recursive and different in form compared to the recursion applicable in the mean value analysis (MVA) algorithm. However, for queue networks with small service centers and multiple chains, MVAc has lower costs compared to MVA.

3. Mathematical basis

3.1. Notations

M = number of stations

$j = 1, \dots, M$ is the index of the station

m_j is the number of service points at station j

C is the number of classes available

$c = 1, \dots, C$ defines the index of the class

N = population vector, i.e. $N = (N_1), \dots, N_c, \dots, N_C$

$n-1_c = n$ with one lesser customer of class c

N_c is the amount of class c customers

n is the current population vector such that $n = (n_1, \dots, n_c, n_C)$

n_c is the current class c population such that $0, \dots, N_c$.

R_j is the expected service length at station j

V_{jc} is the movement through station j by a class c customer

$W_{jc}(n)$ is the movement time at station j by a class c customer under n

$Z_{jc}(n)$ is the number of customers at station j under n

$Q_c(n)$ is the throughput of the network for class c customer under n

$Q_{jc}(n)$ is the expected throughput for station j for class c customer under n

$P_j(l|n)$ is the marginal probability that l customers are present at station j under n

3.2. Mean value analysis algorithm for multi-class, multi-server networks

Legato and Mazza (2017) provided the following as the mean value analysis algorithm for multi-class, multi-server networks.

1. For $j = 1$ to M Do
2. $Z_j(0_1, \dots, 0_C), P_j(0_1, \dots, 0_C) = 1$
3. For $i = 1$ to m_j-1 Do
4. $P_j(i|0_1, \dots, 0_C) = 0$
5. For $n = (0_1, \dots, 0_C)$ to $N = (N_1, \dots, N_C)$ Do
6. For $c = 1$ to C Do
7. For $j = 1$ to M Do
8. $W_{jc}(n) = \frac{R_j}{m_j} \left[1 + \sum_{d=1}^c Z_{jd}(n-1_c) + \sum_{l=1}^{m_j-1} (m_j-1) P_j(l-1|n-1_c) \right]$
9. For $c = 1$ to C Do
10. $Q_c(n) = \frac{n_c}{\sum_{j=1}^M} V_{jc} \cdot W_{jc}(n)$
11. For $j = 1$ to M Do
12. For $c = 1$ to C Do
13. $Z_{jc}(n) = V_{jc} \cdot Q_c(n) \cdot W_{jc}(n)$
14. $P_j(l|n) = \frac{1}{l} \cdot \sum_{c=1}^C R_{jc} \cdot V_{jc} \cdot Q_c(n) P_j(l-1|n-1)$
15. $P_j(0|n) = 1 - \frac{1}{m_j} \left[\sum_{c=1}^C R_{jc} \cdot V_{jc} \cdot Q_c(n) + \sum_{j=1}^{m_j-1} (m_j-1) P_j(l|n) \right]$

3.3. Exponential computational complexity of mean value analysis

According to Samouylov, Dudina and Dudin (2023), the basis of the mean value analysis is the arrival theorem which indicates that the length of queue at station m when there is an arrival is:

$$Q_m(P-1) \quad (1)$$

The characteristic feature of the exponential computational complexity is a product of the complex nature of sojourn time of arrivals in the expression indicated in (1).

As long as there is a recursive dependency on both $Z_j(\mathbf{n})$ and $P_j(l|\mathbf{n})$ regarding n population when a class C arrival is served and consequently leave the system. In this case, Legato and Mazza (2017) gave the computational complexity of the MVA algorithm for time requirement in a station as:

$$C.M.\pi_{c=1}^C(N_c + 1) \quad (2)$$

It is necessary to derive the marginal probabilities so that it could be applied in the derivation of the sojourn time of each customer at each station in a multiple server system. When only one server per station is involved, the sojourn time is given as:

$$W_{jc}(\mathbf{n}) = R_j \left[1 + \sum_{d=1}^C Z_{jd}(\mathbf{n} - 1_c) \right] \quad (3)$$

Obviously from (3), it is necessary to apply the following loop, considering the recursive nature of the MVA algorithm:

For $\mathbf{n} = (0_1, \dots, 0_C) \dots \mathbf{N} = (N_1, \dots, N_C)$ *Do*

This is necessary to derive the computation of the MVA algorithm with regards to its exponential complexity. Consequently, applying this to solve for class C arrivals at station j gives:

$$X = \frac{Z_{jc}(\mathbf{n} - 1_c)}{n_c - 1} - \frac{Z_{jc}(\mathbf{n})}{n_c} \quad (4)$$

However when ρ is considered in the arrivals at station j which is directly consequential of removing a class C arrival from $\mathbf{n}$ with the population of the network, then the following applies:

$$\rho_{jc}\mathbf{n} = \frac{Z_{jc}(\mathbf{n} - 1_c)}{n_c - 1} - \frac{Z_{jc}(\mathbf{n})}{n_c} \quad (5)$$

A meeting point could be established between the expected number of arrivals at station j as well as the value of $\rho_{jc}\mathbf{n}$ for $\mathbf{n}$ by applying the equation:

$$Z_{jc}(\mathbf{n} - 1_c) = (n_c - 1) \left[(Z_{jc}(\mathbf{n}) / n) + \rho_{jc}(\mathbf{n}) \right] \quad (6)$$

If a class C arrival exits the system after service completion, this has no effect on the expected queue length of the different classes. Consequently:

$$Z_{jd}(\mathbf{n} - 1_c) = Z_{jd}(\mathbf{n}) \quad (7)$$

while $d = 1, \dots, C; d \neq c$

Considering (7), the approximation of Bard-Schweither of exact MVA algorithm on a single-server station is given as:

$$W_{jc}(\mathbf{n}) = R_j \left[1 + \frac{n_c - 1}{n_c} Z_{jc}(\mathbf{n}) + \sum_{d=1, d \neq 1}^C Z_{jd}(\mathbf{n}) \right] \quad (8)$$

A further analysis of (7) gives the heuristic definition of the Bard-Schweitzer MVA queues which defines the single-server station because the system throughput can be embedded in the expected queue length such that:

$$Z_{jc}(\mathbf{n}) = n_c \cdot V_j$$

$$Z_{jc}(\mathbf{n}) = \frac{n_c \cdot V_{jc} \cdot W_{jc}(\mathbf{n})}{\sum_{i=1}^M V_{ic} \cdot D_{ic}(\mathbf{n})} \quad (9)$$

In order to improve this method, it is necessary to assume a fractional change of class arrival are available at station j thereby giving rise to the exit of a class C arrival which is a constant value in vector $\mathbf{n}$.

4. Proposed method

The proposed method is the Multiple Server-Station Expected Performance Measure (MSEPM). It is presented as a procedure for class – station performance measure which is highlighted as follows.

4.1. Procedure for class – station performance measure

Step 1: Repeat

1. *Initialize* $P_j(l | N - 1) N = 1, \dots, N; j = 1, \dots, M$

Initialize $W_{jc}(\mathbf{N}) = R_j c = 1, \dots, C; j = 1, \dots, M$

Do

For $c = 1 \dots C$ *Do*

$$Q_c(\mathbf{N}) = \frac{N_c}{\sum_{i=1}^M V_{ic} W_{ic}(\mathbf{N})}$$

For $j = 1 \dots M$ *Do*

$$Z_{jc}(\mathbf{N}) = V_{jc} Q_c(\mathbf{N}) W_{jc}(\mathbf{N})$$

$$W_{jc}(\mathbf{N}) = \frac{R_j}{m_j} \left[1 + \frac{N_c - 1}{N_c} Z_{jc}(\mathbf{N}) + \sum_{d=1, d \neq 1}^C Z_{jd}(\mathbf{N}) + \sum_{i=1}^{m_j-1} (m_j - 1) L_j(l - 1 | N - 1) \right]$$

Until $W_{jc}(\mathbf{N}), c = 1, \dots, C; j = 1, \dots, M$ converges

Step 2: Initialize class sample

Step 3: Derive the multiple server system of the sample class by means of polynomial analysis of mean variance

Step 4:

Calculate $P_j(l - 1 | N - 1)$, while $l = 1, \dots, m_j - 1; j = 1, \dots, M$

until convergence upon $P_j(l - 1 | N - 1)$ while $l = 1, \dots, m_j - 1; j = 1, \dots, M$

return expected performance measure for each station and class by means of

$$Q_{jc}(\mathbf{N}) = \frac{V_{jc} N_c}{\sum_{i=1}^M V_{ic} D_{ic}(\mathbf{N})}$$

while

$$Z_{jc}(\mathbf{N}) = Q_{jc}(\mathbf{N}) \cdot W_{jc}(\mathbf{N})$$

In this case, the values of $W_{jc}(\mathbf{N})$ for $j = 1, \dots, M$ as well as $c = 1, \dots, C$ are those returned at *Step 1* after convergence. This obviously warrants an estimation of the probabilities of marginality of sample class at each station, which is iteratively derived until it converges. In this case, *Step 1* is a 'repeat until' statement as it computes and updates corresponding throughput per class required to derive current parameters which defines sample classes in relations to immediate estimates of given marginal probabilities.

4.2. Functionalities of the class-station performance procedure

The marginal probability that x arrivals are waiting in station j in a single class arrivals of $N = N_1 + N_2 + N_C$ traveling in the network is given as $P_j(x|\mathbf{N})$, $x = 1, \dots, N$. The adoption of Bard-Schweitzer approximation necessitates the application of the function: $P_j(l-1|\mathbf{N}-1)$ for the length of the queue. Consequently, the concept of single class is adopted in consequent illustration. A single class queue network is associated with a multiple class queue system having a population vector of $\mathbf{N} = [N_1, \dots, N_C]$, with the matrix form: $\mathbf{V} = [V_{jc}, j = 1, \dots, M; c = 1, \dots, C]$ as well as a service request vector of the form: $\mathbf{R} = [R_j, j = 1, \dots, M]$. This class is defined as follows:

$$N = \sum_{c=1}^C N_c$$

while

$$V_j = \frac{\sum_{c=1}^C V_{jc} Q_c(\mathbf{N})}{\sum_{c=1}^C Q_c(\mathbf{N})}, j = 1, \dots, M$$

In this case, $Q_c(\mathbf{N})$ defines the values of throughput per class as needed to iteratively compute the class values. This is a fixed-point method having its root in the Bard-Schweitzer proposition for queue lengths as well as the probabilities of the marginal values of the class. The implementation of derivations in Step 1 is rooted in the fact that exact values of the throughput are computed using values given by throughput and marginal probabilities which are iteratively derived. This method requires an initial estimation of marginal probabilities of the class at every station which are iteratively updated until there is a convergence. This implies that the method requires the computation of corresponding values of throughput for each class which are needed to update the parameters defining the class.

5. Implementation framework

This section describes the methods applied in the study.

5.1. Application of proposed method to the solution of multiple-server, multiple-station queue network models

The Arena training and evaluation mode simulator (student) is a setup used to practice and test a model in a controlled environment before actual deployment to limited real – life applications being a trial version. It is applied in class environments by setting up a server queue of the $M/M/C$ form. The component includes: entities (in the form of customers' requests, jobs and packets), resources (such as CPU cores, web servers, etc), queues (the buffer where requests wait) and the process flow (queue delay, service, release and exit). It is possible to run scale-out decision of the system as well as applicable scheduling policies. The simulator was applied in the implementation of the proposed method in the solution of two-queue network models, viz: models 1 and 2.

5.1.1. Model 1

The method was applied in the solution of one multiple-server queue network model as depicted in figure 1.

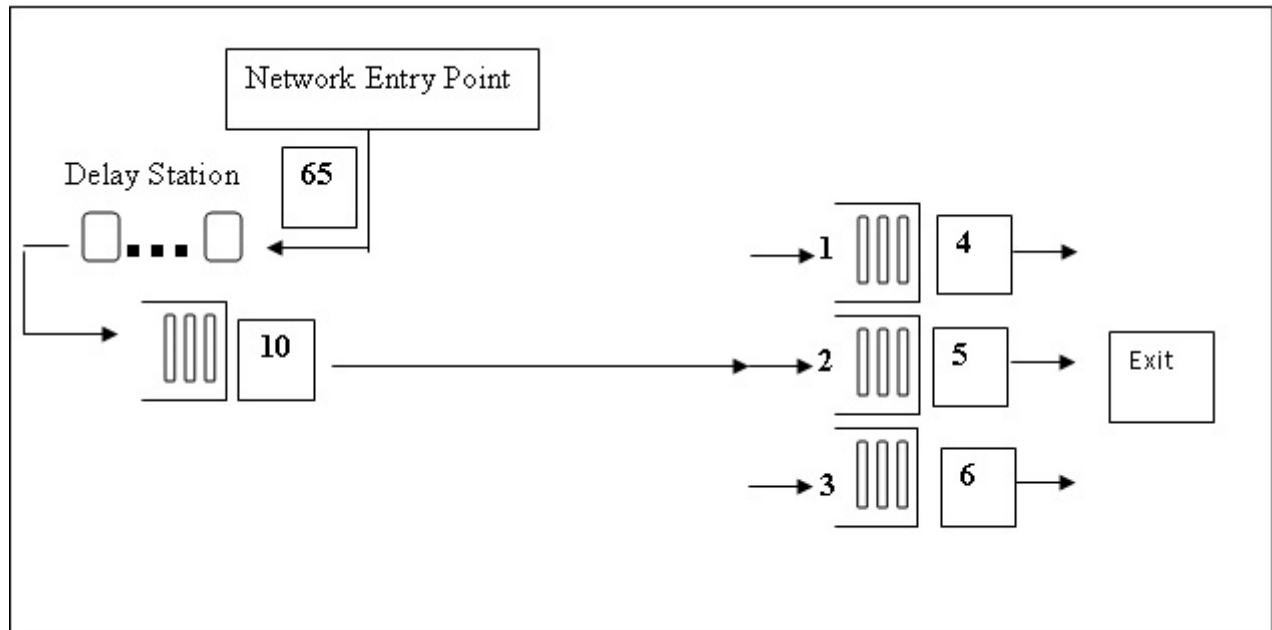


Figure 1. A five-station network with the first station providing delay switching between serviced and waiting customers

Each station is provided with processing facilities as indicated in the numbered braces with an input buffer size that is large enough to prevent an overflow. Customers are serviced using a first-come, first-served discipline and service durations are cumulated over all classes in an exponential probability distribution. Different customer types move from one station to another and are served on first-come, first served basis. Figure 1 has one customer delay station (station 1), one central station (station 2), three client stations (stations 3, 4 and 5) with four classes of customers,

i.e. classes 1, 2, 3 and 4 with corresponding customers' population of 5, 7, 9 and 11 respectively. This implies that there are 32 servers at station 1, which is visited once by a customer in the network as well as the central server i.e. station 2 at least once. The average delay time per class for classes 1, 2, 3 and 4 are 6144, 20480, 30720 and 40960 microseconds respectively. Stations 2, 3, 4 and 5 have 10, 4, 8 and 6 servers respectively as indicated in the braces. Figure 2 is the sample screen shot of movement of customers within each station of the network in model 1 as enumerated using Arena training and evaluation mode (student) simulator.

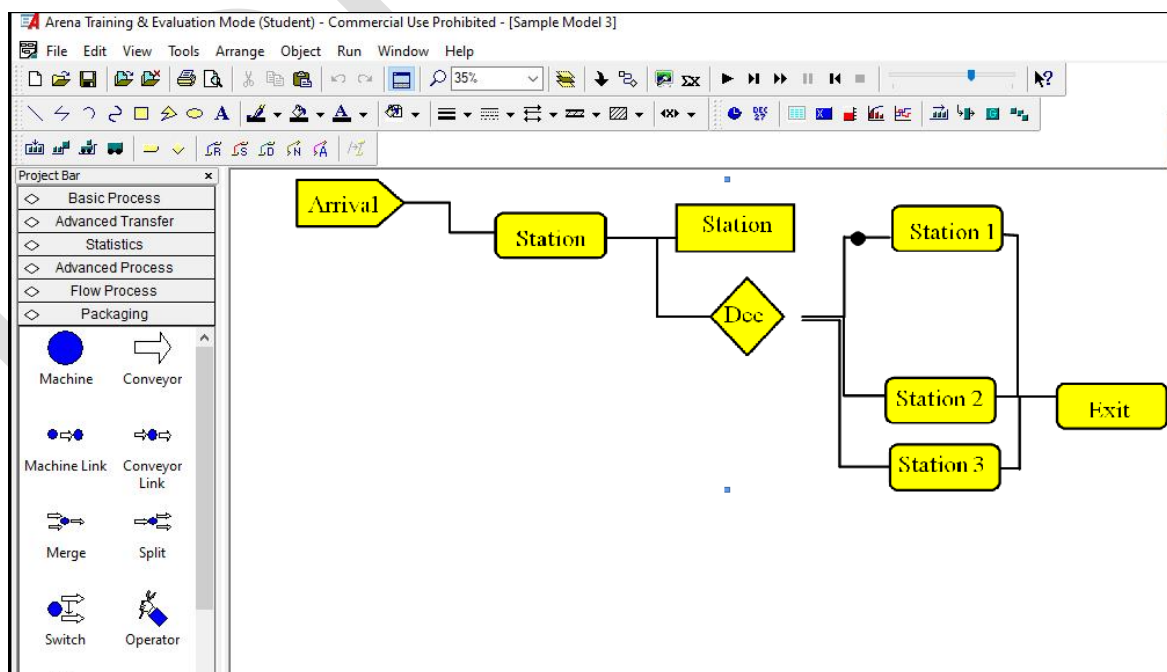


Figure 2. A screen shot of packets' movement within each station of the network in model 1

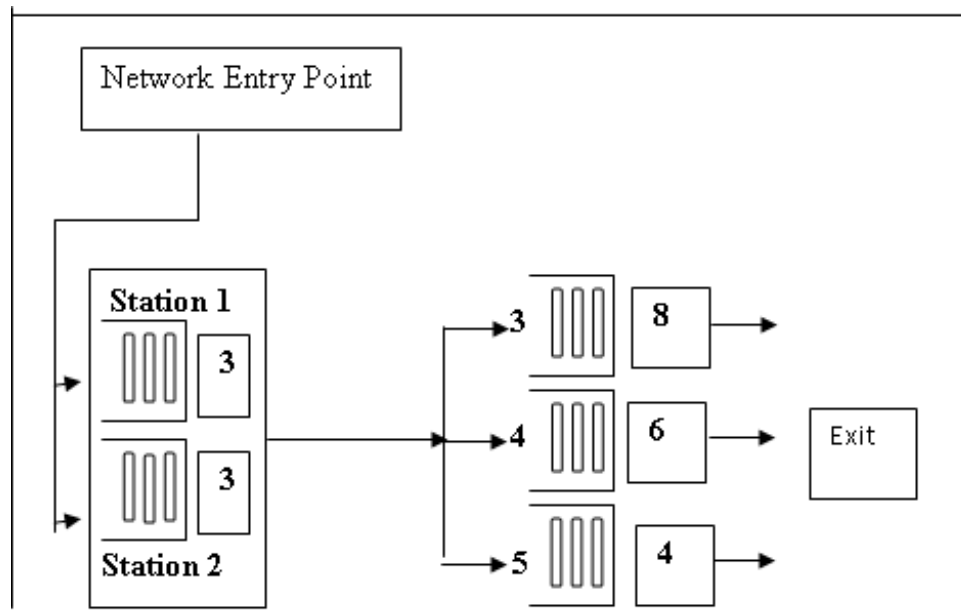


Figure 3. A five-station network with two central stations having equal preference but dedicated customer station relationship

5.1.2. Model 2

The proposed method was also applied in the solution of another multiple server queue model i.e. model 2 as depicted in Figure 3.

In Figure 3, there is no delay station within the network. There are two central stations i.e. 1 and 2 having equal preference with three client stations i.e. stations 3, 4 and 5.

The number of customer classes was increased from 4 to 5. The population of classes 1, 2, 3 and 4 customers are 12, 10, 6 and 8 respectively.

Each of the two central servers has 3 servers each while client stations 3, 4 and 5 have 8, 6 and 4 servers respectively. The model was designed such that all customer types can visit any of the client stations while only class 1 and 2 customers visit the central station 1 and only class 3 and 4 customers visit the central station 2. Figure 4 is the sample screen shot of packets' movement within each station of the network in model 2 as enumerated using Arena training and evaluation mode (student) simulator.

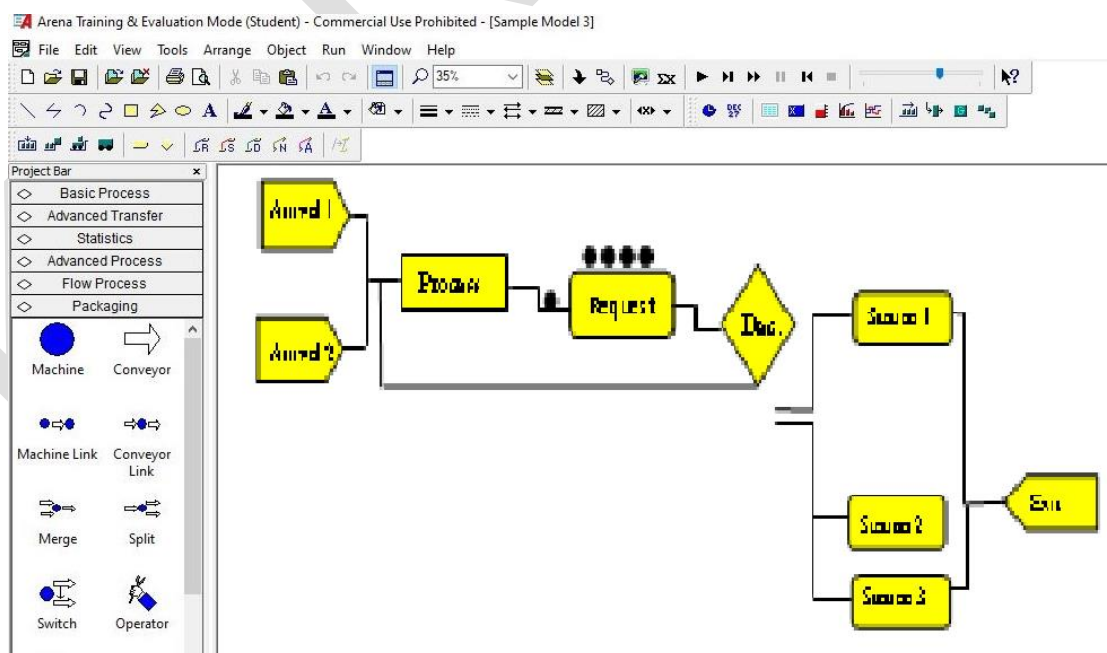


Figure 4. A screen shot of the movement of packets within each station of the network in model 2

5.1.3. Model 2, scenario 1

Table 1 gives the corresponding number of visits of customer classes at each station of the network for scenario 1.

Table 1. Corresponding number of visits of customer classes at each station of the network for scenario 1

Stations (s_i)	Visits per class(c_i)					Scenario 1
	c_1	c_2	c_3	c_4	c_5	
s_1	4	5	2	3	0	0.09
s_2	1	2	8	6	5	0.11
s_3	3	5	1	2	3	0.07
s_4	1	1	2	2	6	0.62
s_5	5	3	4	3	1	0.32

5.2. Results from model 1

The average number of visits and the average service time per visit for this network is given in Table 2.

Table 2: Corresponding service time for customer classes at each station of the network

Class (c_i)	Service time (in microseconds) at each station (s_i)				
	s_1	s_2	s_3	s_4	s_5
c_1	6144	0.75	1.0	2.25	1.75
c_2	20480	0.75	1.0	2.25	1.75
c_3	30720	0.75	1.0	2.25	1.75
c_4	40960	0.75	1.0	2.25	1.75

The values for stations 1 to 5 i.e. s_1 to s_5 with regards to C_i is given in table 3.

Table 3: Corresponding number of visits per station for customer classes at each station of the network

Class (c_i)	Visit per station for customer classes at each station (v_i)				
	s_1	s_2	s_3	s_4	s_5
c_1	1	11	7	5	6
c_2	1	9	5	4	3
c_3	1	7	3	4	3
c_4	1	5	2	3	2

In order to improve service efficiency of the system, the average service times and average number of visits given in tables 1 and 2 respectively were used to modify server utilization factor ranging from 68% at the first client station i.e. station 3 to 96% at the second client station i.e. station 4 in the simulation process. Figure 5 shows the queue lengths using the proposed method, MSEPM and exact MVA method.

The same performance indices were computed using the exact solution of the network returned by the MVA algorithm and the MSEPM. A comparison of performance indicates a significant difference exists in the performance

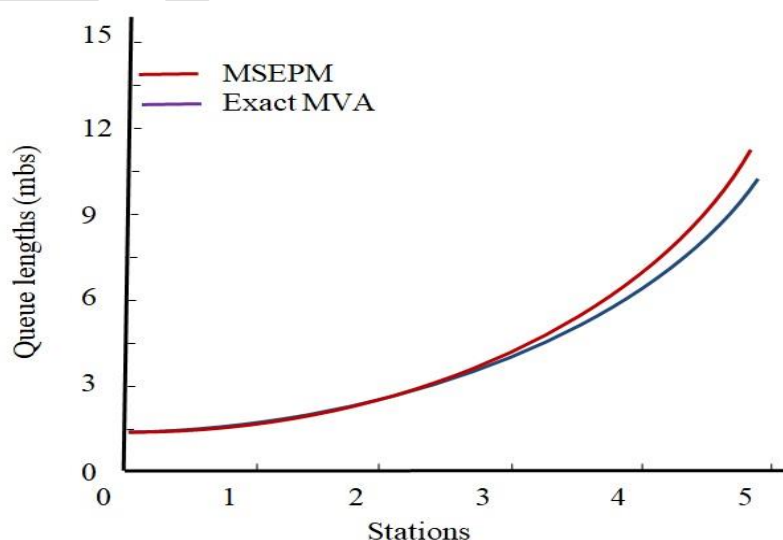


Figure 5. Queue lengths using MSEPM and exact MVA method for class 1 at station 1

of MSEPM and the exact MVA method. For instance, while exact MVA algorithm returned 6.7 m/s as queue length, the proposed method returned 6.43 m/s 12 while the others returned almost the same values.

Figure 6 shows the network throughput for class 1 at station 1 using the MSEPM and the performance indices computed using the exact solution of the network returned by the MVA algorithm.

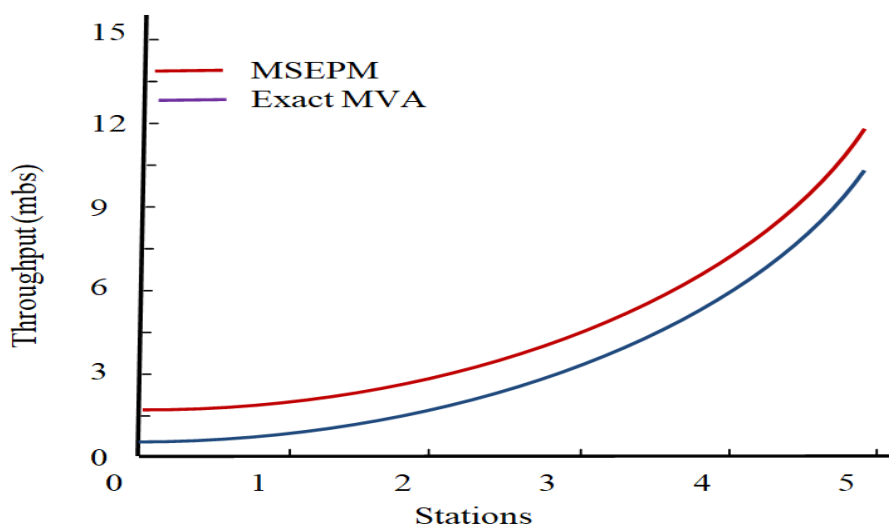


Figure 6. Network throughput using MSEPM and exact MVA method for class 1 at station 1

The comparison of performance indicates a significant difference in the performance of MSEPM and the exact MVA algorithm. For instance, while exact MVA algorithm returned 0.65mbs and 3.0mbs for the first and third stations

respectively as network throughput, the MSEPM returned 1.69mbs and 3.79mbs for the first and third stations respectively as network throughput. The trend is similar for the values returned for throughput at other stations. This indicates significant improvement in the performance of MSEPM over the exact MVA algorithm.

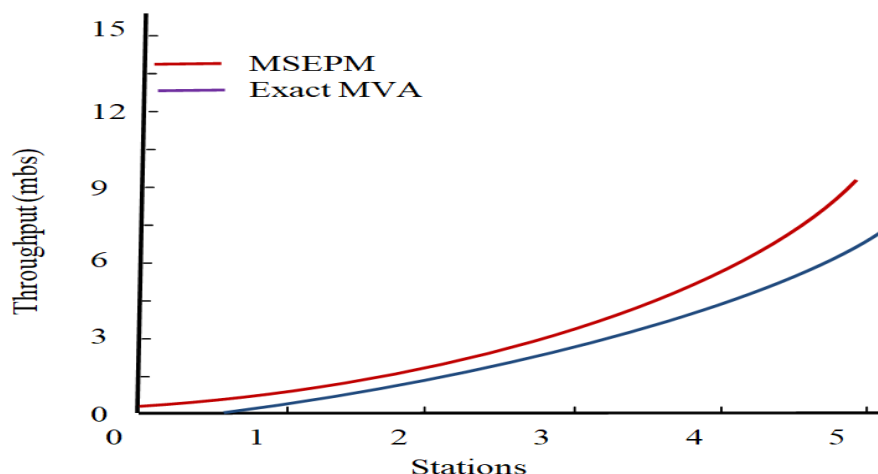


Figure 7. Network throughputs using MSEPM and the exact MVA method for class 1 at station 1

The comparison of performance indicates a difference in the performance of MSEPM and the exact MVA algorithm. For instance, while the exact MVA algorithm returned 1.5 mbs and 4.79 mbs for the second and fourth stations respectively as throughputs, the MSEPM returned 1.65 mbs and 5.67 mbs for the second and fourth stations respectively as throughputs.

Figure 8 shows the network throughput for class 2 at station 1 using MSEPM and also the performance indices computed using the exact solution of the network returned by the MVA algorithm.

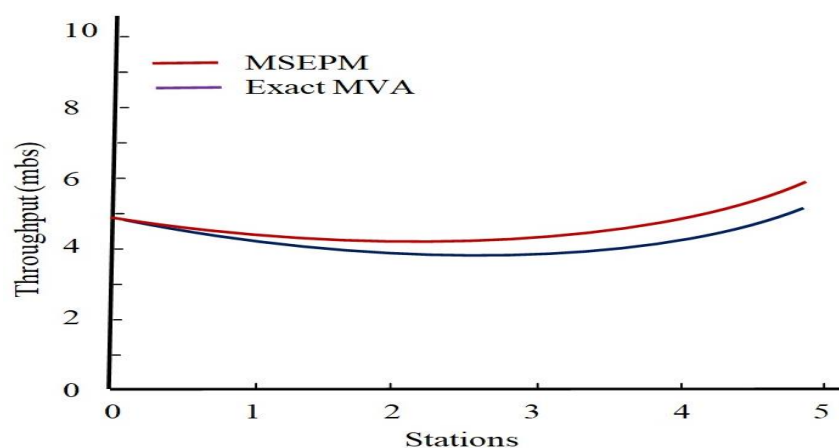


Figure 8. Network throughput using MSEPM and the exact MVA method for class 2 at station 1

There is a significant difference in the network throughput using MSEPM and the exact MVA algorithm. For instance, while exact MVA algorithm returned 4.15mbs at station 1, the MSEPM returned 4.35mbs in this case.

The throughput of the network increases linearly using the MSEPM than the exact MVA algorithm.

Figure 9 shows the queue lengths for class 3 at station 4 using the MSEPM and also the performance indices computed using the exact solution of the network returned by the MVA algorithm.

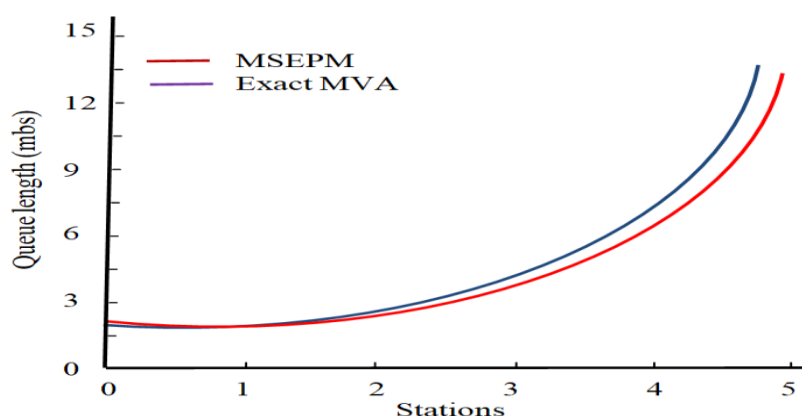


Figure 9. Queue lengths using MSEPM and the exact MVA method for class 3 at station 4

There is a performance difference in the queue lengths of the MSEPM and the MVA algorithm. For instance, while classical MVA algorithm returned 3.78mbs and 8.1mbs for stations 3 and 4, the MSEPM returned 3.65mbs and 7.8mbs for stations 3 and 4 respectively. The trend is same for station 5. However at stations 1 and 2, the difference in performance is not that significant.

The differences in the level of performance significance are at stations 2, 3, 4 and 5 where queue lengths are higher for both methods.

Figure 10 shows the network throughput for class 3 at station 4 using the MSEPM and the performance indices computed using the exact solution of the network returned by the MVA algorithm.

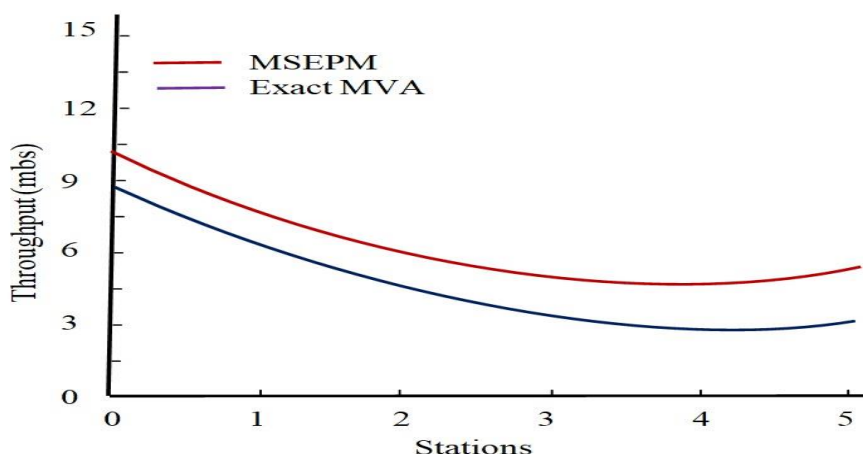


Figure 10. Network throughput of MSEPM and the exact MVA method for class 3 at station 4

In figure 10, there is a significant difference in the network throughput of MSEPM and the MVA algorithm. For instance, while the MVA algorithm returned 3.24mbs and 3.02mbs for the third and fourth stations respectively, the MSEPM returned 5.874mbs and 5.77mbs for the third and fourth stations respectively. In this case, the MSEPM

outperforms the MVA algorithm. The trend is same for the first, second and fifth stations of the queue network.

Figure 11 shows the queue lengths for class 4 at station 5 using the MSEPM and the performance indices computed using the exact solution of the network returned by the MVA algorithm.

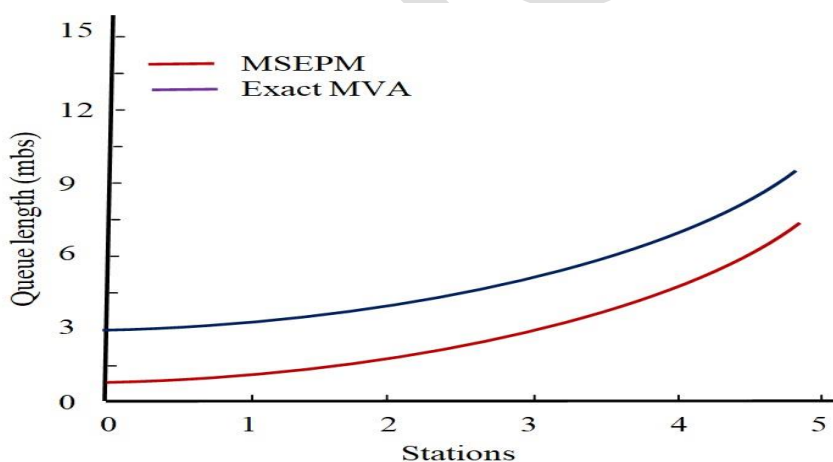


Figure 11. Queue lengths using the MSEPM and the exact MVA method for class 4 at station 5

There is a significant difference in the queue lengths of the MSEPM and the MVA algorithm. For instance, while MSEPM returned 1.35mbs, 1.67 mbs, 2.74 mbs, 4.31 mbs and 6.13 mbs for the first to the fifth stations, the exact MVA algorithm returned 3.36mbs, 4.20mbs, 5.87mbs, 7.23mbs and 10.23mbs for the first to fifth stations respectively. This

indicates that queue lengths are more managed with MSEPM compared to the exact MVA algorithm.

Figure 12 shows the network throughput for class 4 at station 5 using the MSEPM and the performance indices computed using the exact solution of the network returned by the MVA algorithm.

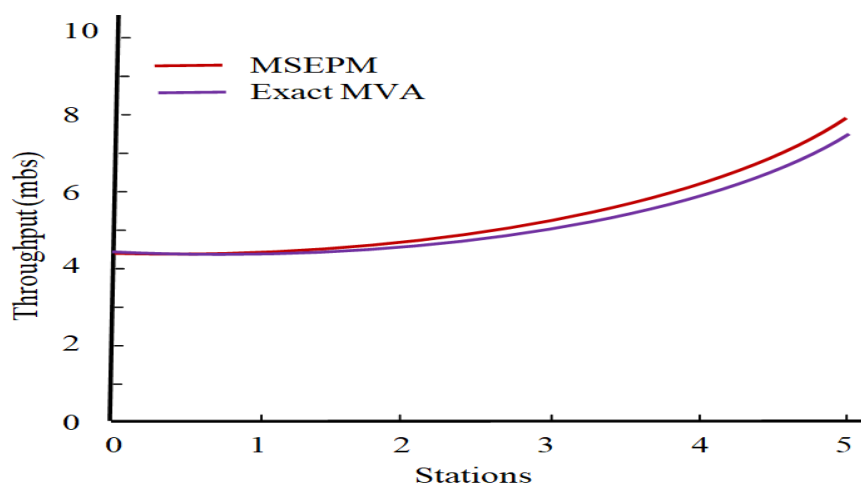


Figure 12. Network throughput for MSEPM and the exact MVA method for class 4 at station 5

There is a significant difference in the network throughputs of the MSEPM and the MVA algorithm. For instance, while MVA algorithm returned 4.35mbs, 4.48mbs, 4.95mbs, 5.11mbs and 8.23mbs, the MSEPM returned 4.35mbs, 4.47mbs, 4.91mbs, 5.09mbs and 8.05mbs respectively. In this case, there is a slight

improvement in the performance of exact MVA over MSEPM.

5.3. Results from model 2, scenario1

Figure 13 shows the queue length performance indices using the MSEPM and exact MVA algorithm for class 1, scenario 1.

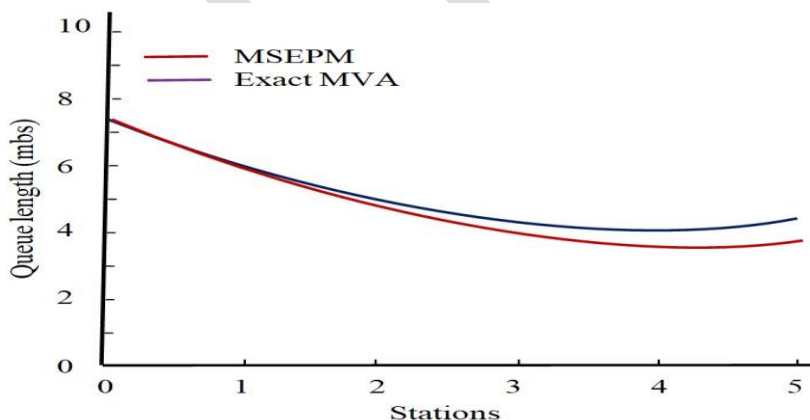


Figure 13. Queue lengths using exact MVA and the MSEPM for class 1, scenario 1

A significant difference occur in the queue lengths using both methods. While the exact MVA algorithm returned 4.25mbs, 4.02mbs and 3.97mbs, the MSEPM returned 3.96mbs, 3.87mbs and 3.84mbs for the third and fourth stations respectively.

Similarly, the first and second stations returned 6.15mbs,

4.87mbs and 6.14mbs, 6.05mbs for the exact MVA and MSEPM respectively.

Figure 14 shows the network throughput performance indices using the MSEPM and the exact MVA for class 1, scenario 1.

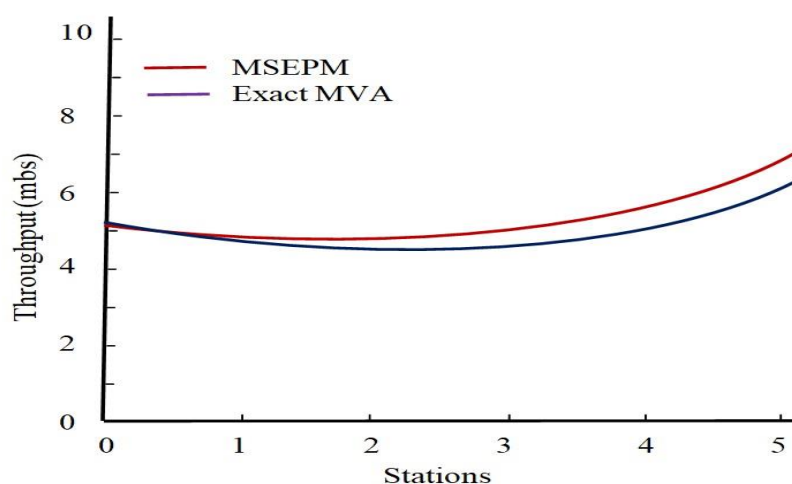


Figure 14. Network throughput using exact MVA and MSEPM for class 1, scenario 1

A significant difference occurs in the performance of both methods. While the exact MVA algorithm returned 4.89mbs, 4.98mbs, 5.05mbs, 5.23mbs and 6.32mbs, the MSEPM returned 4.91mbs, 5.02mbs, 5.12mbs, 5.32mbs and 6.85mbs for the first and fifth stations respectively.

This indicates that the MSEPM was more optimal in system throughput compared to the exact MVA algorithm.

Figure 15 shows the queue length performance indices using the MSEPM and the exact MVA algorithm for class 2, scenario 1.

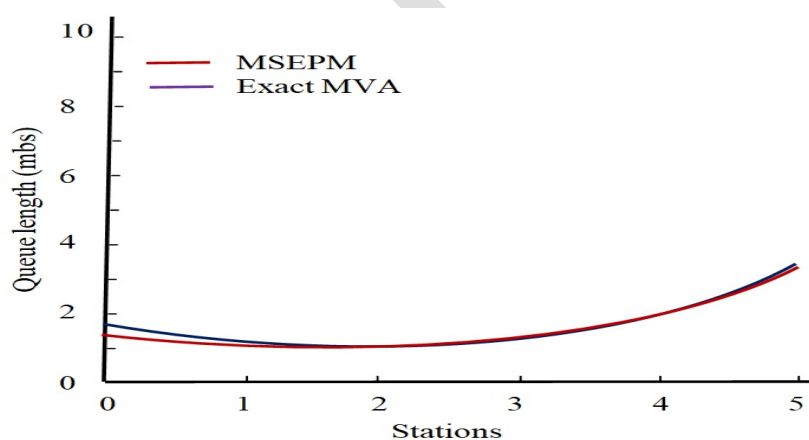


Figure 15. Network throughput using exact MVA algorithm and the MSEPM for class 1, scenario 1

The queue length performance indices using the exact MVA algorithm and the MSEPM are provided. A mild difference occur in the performance of both methods at station 1 while the performance of both at stations 2, 3, 4 and 5 are almost equivalent. While the exact MVA algorithm returned 1.88mbs, the MSPEM returned 1.65mbs at station 1. The performance is almost equivalent at stations 3, 4 and 5.

6. Possible application areas of results

The following are the application areas of results

a. Productions systems: In contemporary industrial settings, emphasis is usually laid on cost effectiveness by minimizing production costs. When the complexity of such

b. systems are at the barest minimum, this will reduce overall costs; thereby increasing organizational profits;

c. Scheduling systems: In allocation problems, jobs are often assigned optimally in order to minimize costs in terms of time, material and logistics. The proposed system is less complex and thereby more applicable in job scheduling in queue networks where this type of complexities might emerge.

d. Congestion management: With the proposed method, traffic is minimized thereby facilitating network stability and predictability. Consequently, network resources are preserved from undue wastages.

7. Conclusion

The performance of queue networks is based partly on the flexibility and the manageability of network functions and parameters. With MVA implementation, algorithmic complexity is inevitable especially when it involves multiple class-server stations. As a result of this, MSEPM was proposed to manage corresponding complexities involved in the implementation of MVA algorithm. This had been effective as evident in the series of experiments presented in this study with significant reduction in the implementation complexities of this algorithm. With the proposed method, network throughput increases while queue lengths were minimized.

Abbreviations

MSEPM	Multiple Server-Station Expected Performance Measure
MVA	Mean Value Analysis
IAMVAL	Improved Approximate Mean Value Analysis Library
TDA	Task-Directed Aggregation
MAP	Markovian arrival process
BCMP	Baskett, Chandy, Muntz and Palacios
MVA _C	Mean Value Analysis by Class
SRVN	Stochastic Rendezvous Networks

Author Contributions

Adegbuyi David Gbadebo: Conceptualization, Investigation, Validation, Writing –Original Draft, Project Administration and Methodology,

Simeon Ayoola Akinleye: Writing-Review & Editing.

Akeem Ademola Adedokun: Writing-Review & Editing

Yewande Adeyosola Akinleye: Writing-Review & Editing

Conflicts of Interest

The author(s) declare that they have no known competing financial interests, professional affiliations or personal relationships that could have appeared to influence the work reported in this paper.

Funding

None

References

- Agboola, S. O. (2021). On the application of mean value analysis without the normalization constant in closed queueing networks. *Faculty of Physical Sciences and Faculty of Life Sciences*, University of Ilorin, Nigeria. doi:10.48198/NJPAS/20.B20.3951-3961
- Amjath, M., Kerbach, L., Elomri, A. *et al.* (2024). Queueing network models for the analysis and optimization of material handling systems: A systematic literature review. *Flexible Service Manufacturing Journal* 36. 668–709. <https://doi.org/10.1007/s10696-023-09505-x>
- Baron, O., Krass, D., Senderovich, A., Sherzer, E. (2024). Supervised ml for solving the $Gi/Gi/1$ queue. *INFORMS Journal on Computing*. 36(3). 766–786. <https://doi.org/10.1287/ijoc.2022.0263>
- Baron, O., Krass, D., Sherzer, E., Senderovich, A. (2022). Can machines solve general queueing problems? In 2022 Winter Simulation Conference (WSC). 2830– 2841. <https://doi.org/10.1109/WSC57314.2022.10015451>
- Calvo, R., Arteaga, A. (2024). Production systems with parallel heterogeneous servers of limited capacity: Accurate modeling and performance analysis. *Applied Sciences* 14(1). <https://doi.org/10.3390/app14010424>
- Chaves, C., Gosavi, A. (2022). On general multi-server queues with non-Poisson arrivals and medium traffic : A new approximation and a covid-19 ventilator case study. *Operational Research* 22(5), 5205–5229. <https://doi.org/10.1007/s12351-022-00712-2>
- Conway, A. E., E. de Souza e Silva, Lavenberg, S. S. (1989). Mean value analysis by chain of product form queueing networks. *IEEE Transactions on Computers*. 38(3). 432–442. doi: 10.1109/12.21129.
- Dudin, S. A., Dudina, O. S., Dudin, N. A. (2024) Analysis of tandem queue with multi-server stages and group service at the second stage. *Axioms*. 13(4). 214. <https://doi.org/10.3390/axioms13040214>
- Efrosinin, D., Vishnevsky, V., Stepanova, N., Sztrik, J. (2025). Use cases of machine learning in queueing theory based on a $Gi/G/K$ system. *Mathematics* 13(5) (2025) <https://doi.org/10.3390/math13050776>
- Legato, P., Mazza, R. M. (2017). Multi-class multi-server queueing networks for production systems design. In *16th International Conference on Modeling and Applied Simulation*, Held at the International Multidisciplinary Modeling and Simulation Multi-Conference. 100-109.
- Nii, S., Okudal, T., Wakita, T. (2020). A performance evaluation of queueing systems by machine learning. In 2020 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-Taiwan), Taoyuan, Taiwan. 1–2 <https://doi.org/10.1109/ICCE-Taiwan49838.2020.9258268>
- Petriu, D. C. (1994). Approximate mean value analysis of client-server systems with multi-class requests. *ACM SIGMETRICS Performance Evaluation Review*. 22(1). 77 – 86. <https://doi.org/10.1145/183019.183027>
- Samouylov, K., Dudina, O. Dudin, A. (2023). Analysis of multi-server queueing system with flexible priorities.

Mathematics.

11(4). <https://doi.org/10.3390/math11041040>

- Sherzer, E. (2025). Analyzing homogenous and heterogeneous multi-server queues via neural networks. *Industrial Engineering*. Ariel University, 65 Ramat HaGolan, Ariel, 407000, Israel. <https://arxiv.org/pdf/2504.01052>
- Sherzer, E. (2024). Computing the steady-state probabilities of a tandem queueing system: A Machine Learning approach. <https://arxiv.org/abs/2411.07599>
- Sherzer, E., Baron, O., Krass, D., Resheff, Y. (2024) Approximating $G(t)/Gi/1$ queues with deep learning. *European Journal of Operational Research* <https://doi.org/10.1016/j.ejor.2024.12.030>
- Suri R., Sahu S., Vernon M. (2007). Approximate mean value analysis for closed queueing networks with multiple-server stations. *Proceedings of the Industrial Engineering Research Conference*. December 2-4, Nashville (Tennessee, USA) 1-6
- Tan, B., Khayyati, S. (2022). Supervised learning-based approximation method for single-server open queueing networks with correlated inter-arrival and service times. *International Journal of Production Research* 60(22). 6822–6847(2022). <https://doi.org/10.1080/00207543.2021.1887536>
- Varki, E. (2021). *Mean value technique for closed fork-join networks*. Department of Computer Science University of New Hampshire. 1-10
- Wang, R., Casale, G. Filieri, A. (2023) Estimating multi-class service demand distributions using Markovian arrival processes. *ACM Transactions on Modeling and Computer Simulation*. 33(1). 1 – 26 <https://doi.org/10.1145/3570924>
- Wang, H., Sevcik, K.C. (1998). Experiments with improved approximate mean value analysis algorithms. In: Puigjaner, R., Savino, N.N., Serra, B. (eds) *Computer Performance Evaluation. Tools. Lecture Notes in Computer Science*. 1469. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-68061-6_23
- Whitt, W., You, W. (2022) A robust queueing network analyzer based on indices of dispersion. *Naval Research Logistics (NRL)* 69 (1), 36–56. <https://doi.org/10.1002/nav.22010><https://onlinelibrary.wiley.com/doi/pdf/10.1002/nav.22010>