Corresponding author: Murray Woodside, cmw@sce.carleton.ca

A Robust Approximation for Multiclass Multiserver Queues with Applications to Microservices Systems

Siyu Zhou and Murray Woodside[0000-0002-2134-5994]

Carleton University, Ottawa, Canada K1S 5B6

zhousiyut@gmail.com, cmw@sce.carleton.ca

Abstract. Model-based management of software applications in the cloud is based on predicted delays at scaled out services. These services are modeled as FIFO (first-in first-out) multiservers, with many servers, users and types of operation (classes of service). Efficient approximations for these multiservers either scale badly for large systems, or have convergence and accuracy problems. This work investigates three scalable approximations in depth. The best (called AB) combines class aggregation and a binomial approximation to the queue state (which assumes that users behave independently). Over the parameters of greatest relevance, two-thirds of the errors are less than 5%. The largest errors, up to about 30%, occur near the onset of saturation.

Keywords: Queueing, Multiservers, Microservices, Cloud Computing.

1 Introduction

Model-based management of services can react quickly to changes and can coordinate the scaling of multiple services, as demonstrated in [12]. However, because a scaled-out service is a multiserver, it depends on efficient and robust solution methods for multiserver queues. Existing methods do not meet all the requirements of (i) accuracy, (ii) robust solution in terms of dependable convergence of iterative steps, and (iii) fast solution, and they have not been fully evaluated. This work fills that gap.

The problem is illustrated by the Sock Shop microservices example used in [12]. Six services have sets of load-balanced replicas. The model in Figure 1 is a layered queueing network (LQN) [8,10,11] adapted from [12] with three sets of Clients. The multiplicity of clients and services is shown in curly brackets such as {225} or (for a variable) {$m}. The small nested parallelograms indicate operations with CPU demands in ms. (e.g. [1.2]), and arrows represent calls. The LQNS solver uses iterative Approximate Mean Value Analysis (AMVA) (e.g., [2]) as an overall strategy with options for different algorithms for the multiservers, having different tradeoffs of accuracy and speed.

Table 1 shows the LQNS solution time for three multiserver options, “Conway” (preferred for accuracy), “Rolia” or “RF” (preferred for speed) and “AB”, which was developed in this research and is new. Conway takes much the longest and scales badly

in general; Rolia is fast, and AB is faster. A second difficulty with the Rolia option, which motivated this research, is weak robustness due to non- convergence.

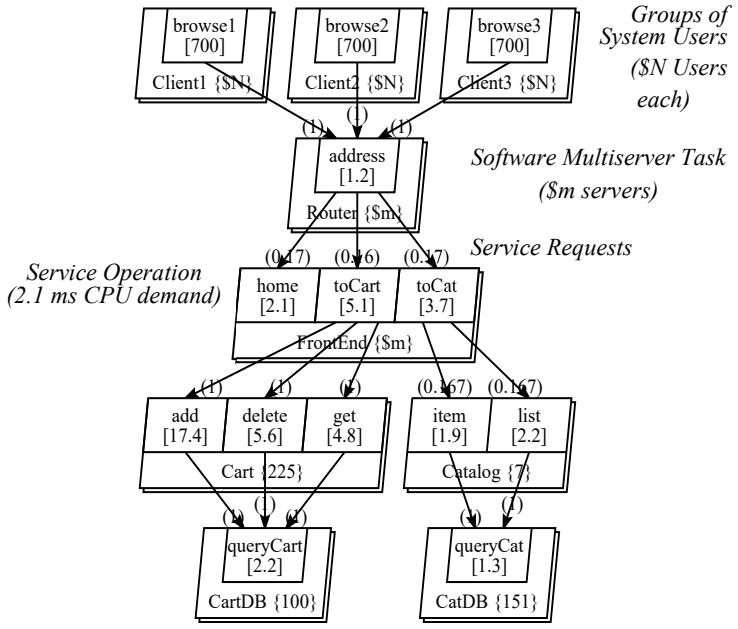


Fig. 1. A Layered Queueing Model of the SockShop Microservices Demo [12]

Table 1. Comparison of LQNS Solution Times of the SockShop Model ($\$N = 7000$, $\$m = 110$)

Approximation	Conway	AB	Rolia (RF)
Complexity per iteration (for N_c customers in class c, m servers, C classes)	$O(C^3 \prod_c N_c)$	$O(m + C)$	$O(C)$
Solution time (seconds)	928.7	0.118	1.426

The goal of this research is to evaluate approximations for multiclass FIFO queues that are heterogeneous (different service times by class), scalable (in the sense of, insensitive to the number of customers) and more robust than RF. They combine scalable single-class approximations which were previously described in [23] with class aggregation. Three approximations are considered here:

- AB, based on a binomial approximation to the queue state distribution and described for a single customer class in [22, 23]
- SS, an equivalent single server (not, at this time, implemented in LQNS and so not shown in Table 1).
- RF, the Rolia algorithm as modified by Franks [8].

The effectiveness of class aggregation in this context, and the accuracy of AB, SS and RF when used with aggregation, were evaluated by comprehensive experiments

covering the model parameter space. AB is both the fastest and the most accurate, and is robust. RF is the least accurate and not robust.

2 The Model and the Approximations

In the multiserver model each customer has two states, "thinking" and requesting service. It is a closed multiclass *M*/*M*/*m*/./*N* queue (that is, with exponential think time, exponential service, m servers, and a finite customer population). It has C classes each with N_c users, mean think time (time between services) Z_c and mean service time S_c, giving parameter vectors $\boldsymbol{N} = (N_1, N_2, \dots)$, $\boldsymbol{Z} = (Z_1, Z_2, \dots)$, and $\boldsymbol{S} = (S_1, S_2, \dots)$. Class c has throughput λ_c, mean waiting time (in the queue) W_c and mean response time R_c, with $R_c = W_c + S_c$ and (by Little's formula) $\lambda_c = N_c/(R_c + Z_c)$.

After aggregation (which is described below) the parameters of the single representative class are written without the subscripts. The probability that a representative customer after aggregation is in the queue or at the server is P:

$$P = R/(R + Z).$$

Applying approximation Appr gives results denoted as $W_{\mathrm{c,Appr}}$, $R_{\mathrm{c,Appr}}$, where Appr is one of Exact, AB, RF, SS, Sim. Approximation errors are reported as the relative error RE. For class c:

$$RE_c(\mathrm{Appr}) = (W_{c,\mathrm{Appr}} - W_{c,\mathrm{Exact}})/\, R_{c,\mathrm{Exact}}$$

Without a subscript, RE refers to the representative single class. The notation ARE is used for the absolute RE, and $MARE$ for the mean absolute RE over a set of cases.

In designing experiments the traffic intensity is set by choosing values of a load intensity value T which is defined as the ratio of the maximum arrival rate N/Z to the maximum service rate m/S:

$$T = NS/(mZ).$$

The response time function has a "knee" near to $T = 1$, where it turns up as traffic increases, and this is where the largest errors were found. Figure 2 illustrates the close relationship between T and the server saturation defined as saturation = utilization/m. The upper curves are for smaller values of m and N, and the lower curves for larger values. For large N, the two have nearly equal values from zero up to near unity, and T then increases without limit as saturation approaches 1.0.

Since service autoscaling typically keeps the saturation below about 0.9, values of T between 0 and 2 seem to be of the greatest practical interest. Higher values may occur at heavily saturated bottlenecks without autoscaling.

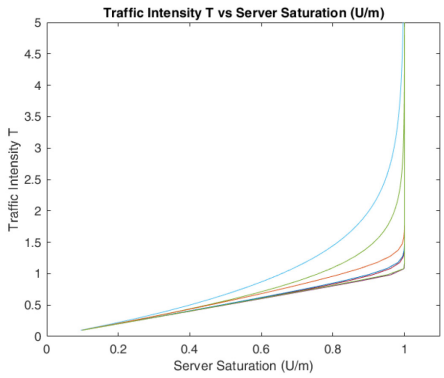


Fig. 2. The Relationship of the Traffic Intensity to Server Saturation for an M/M/m Queue, for all combinations of m = [2, 5, 20] and N = [10, 100, 1000]

For multiclass evaluation experiments the per-class intensities were set to a value $T_c' = T_{nominal}/C$, where $T_{nominal}$ is chosen as a nominal total intensity. The resulting intensity T for the representative class was then found to be close to $T_{nominal}$.

2.1 The Approximations for One Class

For the representative class with parameters N, S, and Z, the approximations are:

The Rolia-Franks Approximation (RF, giving approximation W_{RF}) ([15], [8] ch. 6) assumes independent servers to estimate the probability PB that all servers are busy. Using this PB, it estimates waiting at each iteration by:

$$W = S + PB\ (S/m)\ L^* , \tag{1}$$

where L^* is the expected customers when N is replaced by $N - 1$, as in conventional AMVA [2]. In fixed-point iteration, convergence requires under-relaxation of the form

updated W = α (new W as in Eq. (1)) + (1 - α)(previous W)

with a relaxation parameter α less than unity. The complexity of RF combined with AMVA by Proportional Estimation (PE) [18] or by Linearizer [5] is O(1) per iteration for each multiserver queue, making RF scalable in our sense.

The Equivalent Single Server (SS, giving W_{SS}): Some reports on model-based autoscaling (e.g. [20, 21] have approximated a set of m servers by a faster single server with service time S/m. SS also adds an additional delay $S(1 - 1/m)$ to provide the correct total delay at light loads, which is not found in the references. The additional delay is added to the response time of the server, but does not contribute to the server utilization. The model is solved as a single server by AMVA. Using PE the time complexity of each iteration is O(1), and with Linearizer [5] it is larger but also O(1).

The Arrival-theorem Binomial Approximation (AB, giving W_{AB}): AB assumes that the movement of customers between the thinking and server states is independent, giving a binomial distribution with probability $p^{(B)}(i)$ of i customers at the queue and server. This is the single-class version of the approach by deSouza e Silva and Muntz described in [19]. It gives:

$$p^{(B)}(i) = [N!/((N-i)!\,i!)]P^i(1-P)^{N-i}.$$

Based on the binomial distribution the probability that all servers are busy is PB_B:

$$PB_B = 1 - \sum_1^{m-1} p^{(B)}(i).$$

The mean waiting time can be found from the throughput, given by:

$$\lambda = (1/S)\sum_1^{m-1} ip^{(B)}(i) + (m/S)PB_B,$$

however this direct approach gives inferior estimates. AB instead applies the Arrival Theorem [14] by finding the mean number L* in a queue with $N-1$ customers, indicated by a superscript (N-1):

$$L^* = \sum_{i=m+1}^{N-1}(i-m)p_{AB}^{(N-1)}(i).$$

This can be re-arranged so that it uses only the first m probabilities:

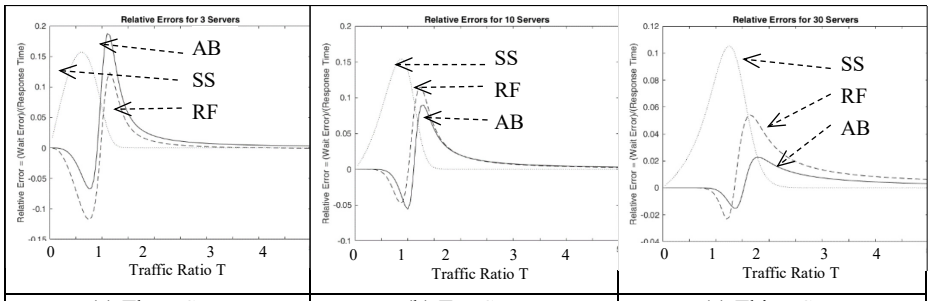
$$L^* = (N-1)P - (m-1)(1-\sum_{i=1}^{m-1} p^{(N-1)}(i)) - \sum_{i=1}^{m-1} ip^{(N-1)}(i). \quad (2)$$

Then by the Arrival Theorem, a customer waits on average for L^* departures, giving

$$W_{AB} = L^* S/m. \quad (3)$$

Using Eq. (2) the complexity of AB is O(m). Since it does not depend on N it is also scalable in our sense.

These three approximations were compared for a single class of customers in [23], from which Figure 3 illustrates the relationship of the errors to the traffic intensity T for cases with N =100, S = 1, $m = 3, 10$ and 30 and Z set to $N/(mT)$. The relative error RE is largest around $T = 1.0$ and approaches zero for light and very heavy traffic.



(a) Three Servers (b) Ten Servers (c) Thirty Servers

Fig. 3. Examples of the Relative Errors of the AB, SS and RF Approximations (from [23])

3 Class Aggregation

Class aggregation is supported by the observation that the waiting times of different classes at a FIFO multiserver are usually not very different [13]. Especially if the

populations of the classes are all quite large, it is intuitively reasonable that an arrival of any class will see a closely similar mix of classes in the queue, and the FIFO discipline treats all classes equally. This is made more precise in the following hypothesis.

Interclass Waiting Difference Hypothesis: The maximum difference between the mean class waiting times at a FIFO queue is the maximum of the class service times at the queue:

$$\max_{c,d}(|W_c - W_d|) \leq \max_c(S_c). \tag{4}$$

A heuristic argument for this hypothesis assumes (1) the Arrival Theorem and (2) that the time to the first departure is the same for all classes. Let Q represent the mean total work in the queue (excluding the server) at steady state, and Q_c, the same quantity if one customer in class c is removed. By the Arrival Theorem the difference between the waiting times of two classes c and d is the difference between Q_c and Q_d :

$$W_c - W_d = Q_c - Q_d.$$

Q_c is less than Q by an amount due to the removal of the customer, an amount that may be reduced by contributions from other classes. This suggests that

$$Q - S_c \leq Q_c \leq Q \text{ and } Q - S_d \leq Q_d \leq Q .$$

The hypothesis follows at once by considering the rectangular set of points (W_c, W_d) defined by the inequalities.

The hypothesis was also verified experimentally on 500 cases spanning the parameter space, using the Conway solution for cases where it gave a value in reasonable time and simulation otherwise, and it was satisfied for every case.

Aggregation Solution. The aggregation gave a single representative class with parameters weighted by the relative throughputs, as:

$$N = \sum_c N_c;\ \ \lambda = \sum_c \lambda_c;\ \ S = \sum_c \lambda_c S_c/\lambda;\ Z = \sum_c \lambda_c Z_c/\lambda;\ T = NS/(mZ)$$

The class throughput depends on the approximate waiting W (using Little's formula), as:

$$\lambda_c = N_c/(W + S_c + Z_c). \tag{5}$$

Because the throughput depends on W the calculation is iterative, with the following outline algorithm:

High-level Multiclass Waiting-time Algorithm

input vectors $\boldsymbol{N, S, Z}$.
initialize $N = \sum_c N_c$, $W = \sum_c N_c S_c/2$, $eps = 10^{-6}$, $relax = 0.7$
repeat until $difference < eps$
    set $\lambda_c = N_c/(W + S_c + Z_c)$ for each class c (Eq.(2))
    find the representative values N, S and Z from Eq (EQ)
    find W_{new} by the approximation AB, SS, or RF
    $difference = |W_{new} - W|$
    $W = relax * W_{new} + (1 - relax) * W$
return W

3.1 Evaluation Cases

As there is no exact solution, the relative errors were found by simulation (with 95% confidence intervals less than 1% of the estimated waiting). Experiments were performed for two, four and eight classes, with 720 cases each. To create each set, all combinations of the following values were employed:

- ten values of vectors **S** and **Z** with components uniformly distributed over [0,1],
- three values of m in the set [2, 4, 8],
- 24 values of the nominal traffic intensity $T_{nominal}$ with values concentrated in the interval (0,2) of greatest interest:
  [0.02,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.84,0.9,0.96,1,1.1,1.2,1.3,1.4,1.5,1.6,2,3,4,6]

For each combination, N_c for each class was set according to the traffic intensity, as

$$N_c = \max(1, \left\lceil \left({}^{T_{nominal}}/_{C} \right) Z_c m / S_c \right\rceil).$$

4 The Approximation Error

Table 2 summarizes the absolute relative errors, including the single-class results reported in [23]. AB has the smallest ARE, and RF the largest. RF also had many convergence failures (fewer if the relaxation factor was reduced, but this also slows down the solver). The maximum errors were close to 50% for AB and SS and nearly 100% for RF (these were due to convergence failures). It is notable that SS is nearly as good as AB on average and has smaller maximum errors.

Table 2. Approximation Errors with Different Numbers of Classes

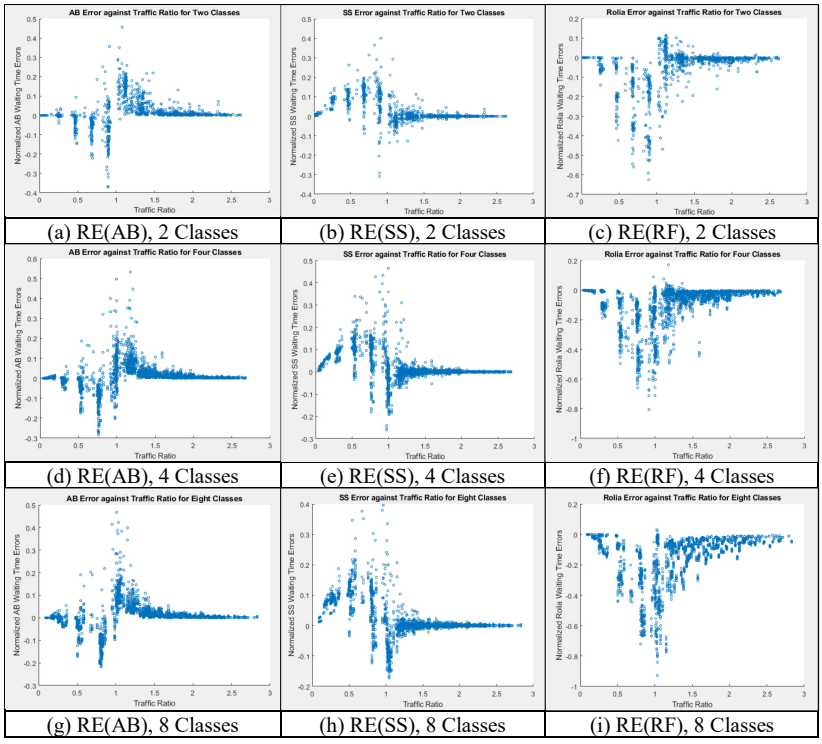
	Measure	Approximation Error			RF non-converged
		AB	SS	RF	
One class, 0<T<5 (30000 cases, [23])	Mean ARE	0.0089	0.0142	0.0191	14130/30000 (*relax* = 0.7)
	Max ARE	*0.243*	*0.190*	*0.249*	
One class, 5<T<36 (30000 cases, [23])	Mean ARE	0.00041	0.00079	0.00307	24030/30000 (*relax* = 0.7)
	Max ARE	*0.0472*	*0.1859*	*0.08792*	
Two classes (720 cases)	Mean ARE	0.04059	0.04132	0.06166	2/720 (*relax* = 0.1)
	Max ARE	*0.4563*	*0.4005*	*0.6249*	
Four classes (720 cases)	Mean ARE	0.03397	0.03895	0.07989	5/720 (*relax* = 0.1)
	Max ARE	*0.5321*	*0.4649*	*0.8061*	
Eight classes (720 cases)	Mean ARE	0.03236	0.03752	0.1191	25/720 (*relax* = 0.1)
	Max ARE	*0.4669*	*0.3962*	*0.9288*	

As classes increase, the mean errors decrease for AB and SS, but increase for RF. The relatively small mean errors reported for a single class reflect the distribution of single-class cases which emphasized large values of T (which gave small errors).

The relationship of the relative error to the traffic intensity is displayed in the scatter plots of Figure 4. The plots are restricted to $T < 3$; all errors were small for larger T. The error patterns are all similar, and similar to the examples in Fig. 3. Away from T=1 the errors approach zero fastest for AB and slowest, for RF. Above 30% error there are

only nine points for AB, and six points for SS, but many more (with much larger values) for RF. The largest errors for RF were due to non-convergent solutions, which were included in the plots.

The scatter plots tend to conceal the preponderance of points with small errors, which is better seen in the percentile values in Table 3. For AB almost two-thirds of the cases had less than 5% error.



(a) RE(AB), 2 Classes (b) RE(SS), 2 Classes (c) RE(RF), 2 Classes

(d) RE(AB), 4 Classes (e) RE(SS), 4 Classes (f) RE(RF), 4 Classes

(g) RE(AB), 8 Classes (h) RE(SS), 8 Classes (i) RE(RF), 8 Classes

Fig. 4. Scatter plots for the Approximation Errors

Table 3. Percentiles of Absolute Relative Errors

Range of ARE	AB	SS	RF
0 – 0.01	0.42	0.38	0.40
0 – 0.05	0.65	0.57	0.58
0 – 0.20	0.96	0.98	0.82

4.1 Errors with $T < 2$ and Large N

It was argued in Section 2 that practical cases will tend to have $T < 2$ and large N, and anther set of cases was examined in this range.

Simulations were run with all combinations of the parameters N = [100, 200, 300, 400, 500], m = [2, 5, 10, 20, 50]. T took 10 values between 0.2 and 2.0, and S was

random with a mean of 1.0. Fig. 5 shows error histograms and Table 4 shows the mean errors for 2, 4 and 8 classes.

Figure 5 shows that when we focus on T less than 2.0, the errors for AB increase with more classes, unlike the results for the wider range of traffic, and the other approximations do not show a clear trend.

AB has errors less than 5% in more than two-thirds of the cases, and less than 20% in 94% or more of the cases. This is somewhat better than SS and very much better than RF. If we consider only 10 or more servers RE is much smaller, less than 5% in over 85% of cases, and less than 20% in over 98% of cases.

For less than 10 servers the error increases with the number of classes, possibly due to class aggregation, but with 10 or more servers this was not observed.

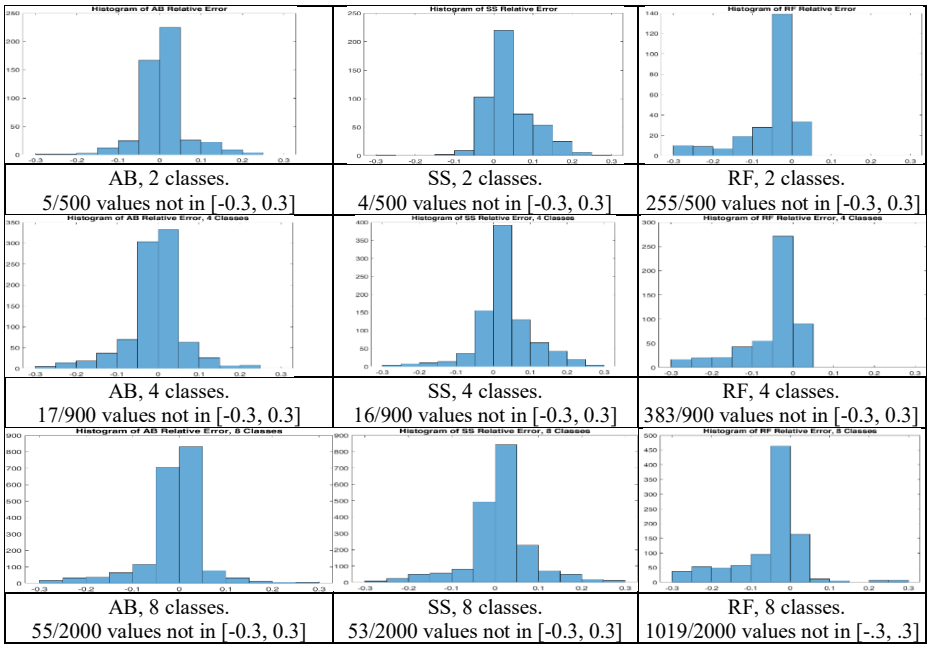


Fig. 5. Histograms of RE for the Waiting Times of All Classes

Table 4. Mean Absolute Relative Errors with Populations in [100, 500] and T in [0.2, 2]

	$MARE$ for AB	$MARE$ for SS	$MARE$ for RF
2 Classes (250 cases)	0.0364	0.0483	0.4246
4 Classes (225 cases)	0.0453	0.0584	0.3569
8 Classes (250 cases)	0.0625	0.0568	0.6329

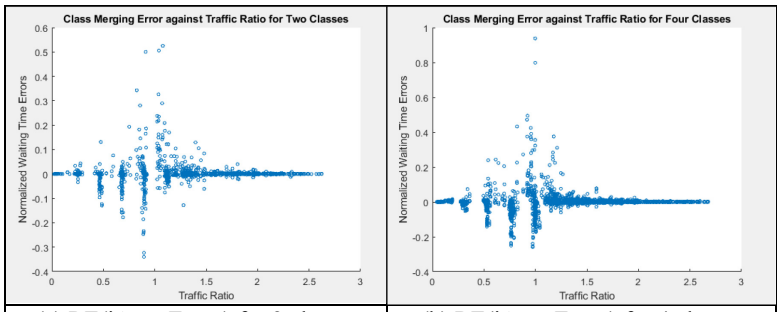
4.2 Error Component Due to Class Aggregation

To isolate the error due to class aggregation, an "ideally aggregated" model "iAgg" was created using simulation throughputs as weights. RE(iAgg+Exact) is the error of

aggregation only, using the exact single class solution, while RE(iAgg+Appr) is the error of approximation Appr applied to the iAgg model. Table 5 shows that the errors for the two-class and four-class cases are roughly evenly divided between aggregation error and the approximation errors (columns 4, 5, 6 are about double column 3). Scatter plots in Fig. 6 show that the aggregation error alone is largest around $T = 1$, and takes both positive and negative values. The results in column 3 suggest that the aggregation error may increase with C, but the approximations appear to counter than increase, with smaller errors for four classes.

Table 5. Errors when Classes Ideally Aggregated Based on the Simulation Throughputs

	Measure	Algorithm used for the Ideal Representative Class			
		iAgg+Exact	iAgg+AB	iAgg+RF	iAgg+SS
Two Classes	$MARE$	0.01599	0.03887	0.04031	0.03922
Four Classes	$MARE$	0.01966	0.03250	0.03245	0.03431



(a) RE(iAgg+Exact) for 2 classes (b) RE(iAgg+Exact) for 4 classes

Fig. 6. The Approximation Error due only to Class Aggregation

5 Application to the SockShop Microservices Model

SockShop is a small but realistic microservices demonstration system. Figure 1 shows a LQN performance model adapted from [12], with three groups of users instead of one, and variable numbers of service replicas. The three approximations implemented in the LQNS solver (AB, RF, Conway) were compared in predicting the performance effect of different scalings of the Router and FrontEnd services. given by the parameter $\$m$ in Figure 1. The value shown of 10 replicas was increased in 15 steps up to 240, reducing the response time as shown in Figure 7. The performance improves by several orders of magnitude. The AB results were identical to Conway within 3 figures, while Rolia diverged by up to 10% at the lowest value of $\$m$..

The mean CPU time for the solution across the cases was 0.37 sec. for AB, and 1.05 sec for Rolia. The time for Conway increased steeply with $\$m$ up to 28859 sec (more than 8 hours) at $\$m$ = 190, where the experiment stopped. Thus in this case Conway is impractical, while Rolia is less accurate than AB and takes nearly three times as long (although both are quite fast).

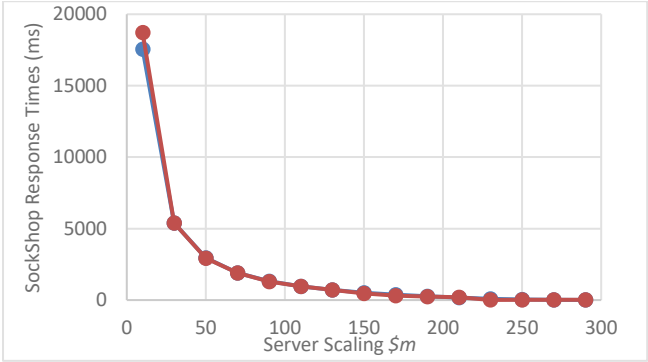


Fig. 7. Response Time of the SockShop Model with Scaled Front-end Services

6 Scalability to of the Approximations to Very Large Models

To consider large models it is not convenient or necessary to use models of real systems. Every possible pattern of components and interactions might be of interest, so models with random structure and parameters were used, with 20, 100 and 200 tasks. Each task offers an average of 10 operations and is scaled to an average of 20 replicas; the model represents a highly complex system. The 100-task case is illustrated in Figure 8. All cases had 5 groups of users.

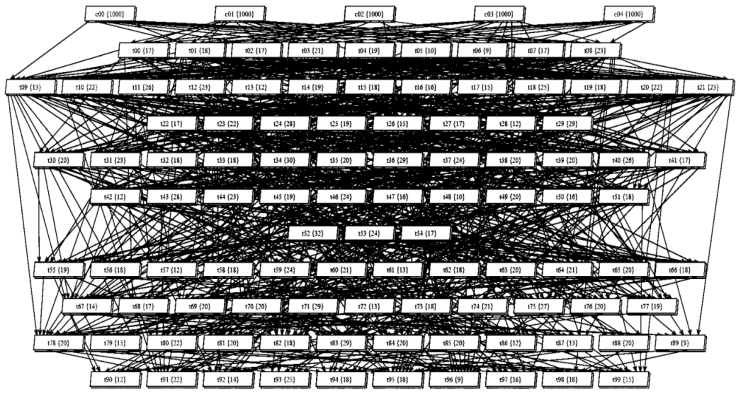


Fig. 8. A Random Model with 100 Tasks.

The models were solved by AB and RF, with Conway for further comparison. SS was not applied because it is not implemented in the solver. The solution times for these models are shown in Table 6.

Table 6. Average Solution Times for Large Random Models

Cases	AB	RF	Conway
15 cases with 20 Tasks and 20–300 users per group.	0.062 sec	0.189 sec	14.138 sec
20 cases with 100 Tasks and 50–1000 users per group.	10.3 sec	358.2 sec	no result
One case with 200 Tasks and a mean of 10 replicas each	80.5 sec	no result	no result

The AB algorithm is clearly more scalable and more robust than RF. The 200-task model is approaching the storage limits of the solver, and exceeds those of the simulator. The increase in the times for AB is greater than is accounted for by the complexity of AB per server and per iteration, presumably dominated by other operations in LQNS.

7 Related Work on FIFO Multiclass Multiservers

The exact solution for product-form multiclass multiservers (for FIFO service with equal class service times or for processor-sharing queueing with equal or unequal times) is described in [14]. For FIFO service and unequal times, AMVA approximations are described in [1, 5] and (by recasting the solution as an optimization problem) in [3]. Recently Legato and Mazza [13] have solved these networks using class aggregation, giving a single class queueing network which they solve by exact MVA. They go further in aggregation than our SS approximation, which keeps the classes separate except for the calculation of waiting times at multiservers. However, all of these methods apply to FIFO servers only if the mean class service times are equal.

For unequal class service times, Ruth [16] adapted exact MVA; Schmidt [17] approximated the multiserver by a (product-form) state-dependent-rate server; de Souza e Silva and Muntz approximated the queue state by a multinomial distribution [19]; and Conway adapted their approach to use Linearizer [6]. Rolia in [15] introduced a much simpler AMVA as described above and Franks [8] made a modest improvement to it. Casale [4] has recently proposed replacing any server by an approximate processor-sharing server updated iteratively based on properties of a diffusion model; the work does not mention multiservers but appears to be capable of generalization in that direction.

Regarding class aggregation, Dowdy et al [7] considered aggregated classes in queueing networks in which class service times at FIFO servers are the same, and showed that the aggregate analysis understates the total throughput. This result suggests that waiting time errors due to aggregation should be positive, however with different class service times the results reported in Section 4 show both positive and negative errors.

Single-server approximations similar to SS have been used (without the added delay term) in applied studies such as [20, 21], but only for a single class. Their approximation accuracy does not appear to have been studied previously, for one or many classes.

8 Conclusions

All the approximations which are scalable in the sense that their time complexity does not increase with the customer population have similar patterns of error. The results in Section 4.1 indicate that errors are substantial only in an interval of the traffic intensity T between 0.5 and 1.5, and they do not exceed a relative value of about 30%. All the approximations had asymptotic errors under light and heavy traffic which rapidly approached zero. For AB and RF the approach as T diverged from unity was more rapid than for SS.

The practical usability of model-based autoscaling using these approximations may be limited by these errors, since the stated traffic levels include the intended operating point of most autoscaling systems. Model-based autoscaling would have to tolerate this range of prediction errors in the response time of individual services. Fortunately in a large system with many scaled services the errors will tend to average out, so the overall QoS control may be considerably better than this. Also predictions regarding more heavily loaded services, which are the most important, are more accurate.

Practical usability also implies a preference for AB or SS over RF, due to non-convergence of RF in some situations, particularly for large T.

The errors for heterogeneous multiclass servers are contributed partly by class aggregation and partly by the use of the approximation for the single representative (i.e. aggregated) class. The results in Section 4.2 indicate that the two contributions are roughly equal.

The impact of the number of classes C on the error depends on the traffic levels. For T in a broad range (in Table 2) the average error decreases with C for AB and SS, and increases for RF. For T near to unity (in Table 4) it increases for AB, and shows no clear trend for SS and RF. The impact of the number of servers, m, is also complex. However if m is larger than about 10, then the error decreases with further increases in m, for all approximations.

An unexpected result is that SS is quite close to AB in accuracy. Clearly it is much the simplest to calculate, so it might be the algorithm of choice. The additional delay term in SS has to be implemented with care since it is part of the response time of the server.

References

1. Akyildiz, I.F., Bolch, G.: Mean Value Analysis Approximation for Multiple Server Queueing Networks. Performance Evaluation 8, 77-91 (1988).
2. Bolch, G., Greiner, S., Meer, H., Trivedi, K.: Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications, 2nd Edition (2006).
3. Casale, G. Perez, J., Wang, W.: QD-AMVA: evaluating systems with queue-dependent service requirements, Performance Evaluation 91, 80-98 (2015).
4. Casale, G.: Integrated Performance Evaluation of Extended Queueing Network Models With LINE. In: Proc. Winter Simulation Conference, pp. 2377-2388 (2020).
5. Chandy, K.M., Neuse, D.: Linearizer: A heuristic algorithm for queueing network models of computing systems. Comm. of the ACM 25(2), 126-134 (1982).

6. Conway, A.E.: Fast approximate solution of queueing networks with multi-server chain-dependent FCFS queues. In: Modeling Techniques and Tools for Computer Performance Evaluation. pp 385-396, Plenum, New York (1989).
7. Dowdy, L.W., Carlson, B.M., Krantz, A.T., Tripathi, S.K.: Single-class bounds of multi-class queuing networks, J. ACM 39(1) pp 188–213 (1992).
8. Franks, G.: Performance analysis of distributed server systems. PhD thesis, Carleton University (1999).
9. G. Franks, G.: lqngen − generate layered queueing network models, manual page at https://github.com/layeredqueuing/V5, last accessed Feb 10, 2022.
10. Franks, G., Al-Omari, T., Woodside, M, Das, O., Derisavi, S.: Enhanced Modeling and Solution of Layered Queueing Networks, IEEE Trans. Software Engineering, 35(2) 148-161 (2009).
11. Franks, G. et al: Layered Queueing Network Solver and Simulator User Manual, Carleton University, at http://www.sce.carleton.ca/rads/lqns/userman22.pdf, accessed Jan 20, 2022.
12. Gias, A.U., Casale, G., Woodside, M.,: ATOM: Model-Driven Autoscaling for Microservices. In: 39th Int. Conf. Distrib. Computing Systems (ICDCS), pp 1994-2004. (2019).
13. Legato, P., Mazza, R.M.: Class Aggregation for Multi-class Queueing Networks with FCFS Multi-server Stations, Queueing Theory with Network Applications. LNCS vol. 11688, pp. 221–239 (2019).
14. Reiser, M., Lavenberg, SS.: Mean-value analysis of closed multichain queueing networks, J. of ACM 27(2) 312–322, 1980.
15. Rolia, J.A., Sevcik, K.C.: The Method of Layers, IEEE Trans. Software Engineering 21 pp 689 – 700, 2015.
16. Ruth, A.: Entwicklung, Implementierung und Validierung neuer Approximationsverfahren fur die Mittelwertanalyse (MWA) zur Leistungsberechnung von Rechnersystemen. Diplomarbeit am IMMD der Friedrich-Alexander-Universitat Erlangen-Nurnberg (1987).
17. Schmidt, R.: An approximate MVA algorithm for exponential, class-dependent multiple servers. Performance Evaluation 29 245-254. (1997).
18. Schweitzer, P.J.: Approximate anaIysis of multiclass closed net works of queues. In: Proc. Int. Conf. on Stochastic Control and Optimization pp 25-29, Amsterdam (1979).
19. de Souza e Silva, E., Muntz, R.R.: Approximate solutions for a class of non-product form queueing network models, Performance Evaluation 7 221-242. (1987).
20. Zhang, Q., Xiao, Y., Liu, F., Lui, J.C.S., Guo, J., Wang, T.: Joint Optimization of Chain Placement and Request Scheduling for Network Function Virtualization. In: Int. Conf. Distrib. Computing Systems (ICDCS), pp 731-741. (2011).
21. Q. Zhang, Q. Zhu, M.F. Zhani, R. Boutaba, J.L. Hellerstein, “Dynamic Service Placement in Geographically Distributed Clouds”, IEEE J. on Selected Areas in Communications, v 31, n 10, Oct. 2013.
22. Zhou, S.: A New Approximation for Multiserver Waiting Time for Layered Queueing Systems. MASc thesis, Carleton University (2021).
23. Zhou, S., Woodside, M.: A Multiserver Approximation for Cloud Scaling Analysis. In: Proc. Workshop on Challenges in Software Performance (WOSPC-22), in the Companion Volume to the Int. Conf. on Performance Engineering (ICPE22), ACM, New York (2022).