Topological Network Design of General, Finite, Multi-Server Queueing Networks

J. MacGregor Smith* F. R. B. Cruz† ‡ T. van Woensel§
jmsmith@ecs.umass.edu fcruz@est.ufmg.br T.v.Woensel@tm.tue.nl

November 22, 2008

Abstract — **The topological network design of general service, finite waiting room, multi-server queueing networks is a complex optimization problem. Series, merge, and split topologies are examined using an approximation method to estimate the performance of these queueing networks and an iterative search methodology to find the optimal buffer allocation within the network. The coefficient of variation is shown to be a significant factor in the buffer allocation for multiple servers in uniform and bottleneck server networks. Extensive computational results are included to illustrate the symmetries and asymmetries in the buffer patterns which emerge from the series, merge, and splitting topologies.**

Keywords — Manufacturing; multi-server systems; network design; buffer allocation.

1 Motivation

MULTI-SERVER, open finite queueing networks occur throughout most physical systems such as manufacturing facilities, telecommunications, and transportation. Traditionally, many of the models presented have been the subject in manufacturing environments (Dallery and Gershwin, 1992). Even today, stochastic models are used to analyze and optimize such systems (see, for instance, Eklin et al., 2009). Consider the manufacturing example presented in Fig. 1 representing the conceptual design of an automotive assembly system (Spieckermann et al., 2000) where finite buffers are necessary to avoid breakdowns in one area of the plant and to de-couple the assembly process. How one allocates the optimal buffers is the essential question for this assembly process.

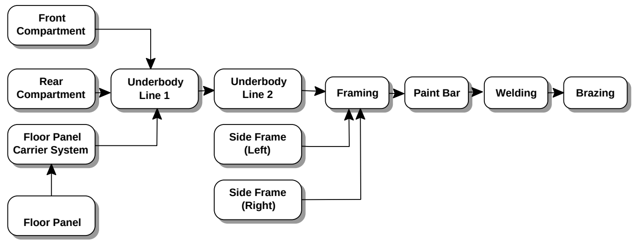

Figure 1: Queueing Network of Automotive Assembly System (Spieckermann et al., 2000)

Different types of operating systems could benefit from the application of the models presented in this article. As pointed out earlier in this section, we consider manufacturing facilities, telecommunications, and transportation, as the prime application areas. More specifically, manufacturing environments such as flow lines, or assembly lines as depicted in Fig. 1. Similarly, transportation infrastructures could be designed in such a way that the network flows are maximized taking into account congestion obtained via the queueing networks worked with in this paper (see, for instance, Cruz et al., 2008b). In telecommunications, one could consider the buffer allocation space in call centers, *i.e.* how many calls do we accept waiting such that a minimum number of serviced calls (or throughput) can be guaranteed. Slightly different are web applications which typically have a number of tiers which a request needs to traverse before finishing its processing. The Quality of Service (QoS) is often measured in terms of throughput (defined as the rate at which a service can process requests), next to server availability or response times (see Menasce, 2002). It is extremely important to identify this specific configuration that minimizes such an objective. Many of these application areas have a few things in common: (1) the performance is measured via the throughput rate which is hard to obtain in a closed-form equation; (2) this throughput is deteriorated due to significant variability in the service times and in the arrival rates; (3) the optimization involves a non-linear structure, making the problem in itself hard. In this research article,

*Department of Mechanical and Industrial Engineering, University of Massachusetts Amherst, Massachusetts 01003, USA.
†School of Computer Science and Information Technology, University of Nottingham, Jubilee Campus, Wollaton Road, Nottingham NG8 1BB, UK. On sabbatical leave from the Department of Statistics, Federal University of Minas Gerais, 31270-901 - Belo Horizonte - MG, Brazil.
‡Corresponding author. E-mail: fcruz@est.ufmg.br. P: +55 31 3409 5929. F: +55 31 3409 5924.
§Department of Technology Management, Operations, Planning, Accounting & Control, Eindhoven University of Technology, Eindhoven, The Netherlands.

we will confine ourselves to the general structure: a finite queueing network. Obviously, this could be replaced by any of the described systems, which then is modeled as finite queueing network. Thus, making abstraction of the specific environment, wherever there is a flow of goods and uncertainty about the processing of these goods at the nodes of the network, then the allocation of resources to the processing of this flow results in a finite queueing network of system resources. The allocation of resources we are concerned about here includes the buffers and the order of the servers and their interaction. The question posed in this research is how we can effectively design and model these systems and accurately predict their performance measures.

1.1 *Purpose*

In this paper, we seek to characterize and optimize the topology of finite queueing systems. We seek properties that allow us to both model and optimize these systems and construct algorithms for their solution. This paper is an extension of previous work in which we only considered single-server, finite buffer systems (Smith and Cruz, 2005). As such, with multi-server systems, we need to see how multi-servers affect optimal buffer allocation and additionally how various topologies and systematic variations in the general service times coefficient of variation play out.

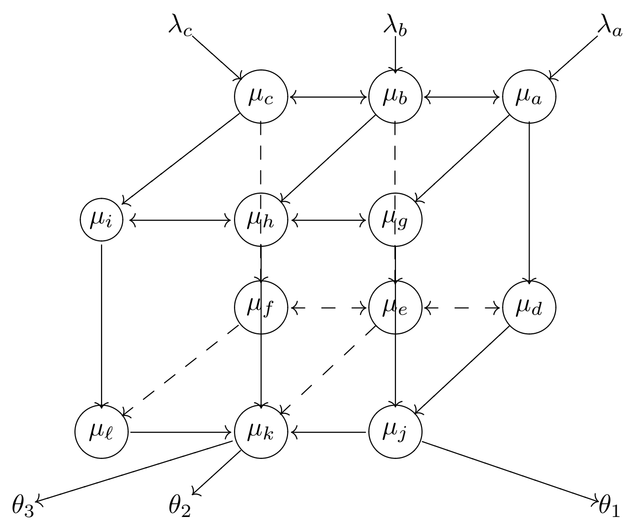

Figure 2: 3-d Cubic Network Topology

We are given a finite network $G(N, A)$ of a specified topology, with N nodes and corresponding arc pairs with general service at the nodes and routing probabilities on the arcs. See Fig. 2 for a possible 3-d cubic example in (x, y, z) Cartesian-coordinate space with variable input arrival rates $\lambda_i, i = a, b, c$ and service rates $\mu_a \ldots \mu_\ell$ and throughput outputs $\theta_j, j = 1, 2, 3$. We seek to determine the optimal performance measures of this network such as throughput, work-in-process, utilization, and optimal costs or profits. Since the network has finite capacity, there is blocking in the network that consequently gives rise to non-product form characteristics which makes the problem very difficult to easily derive the probability distribution of the number of customers within the network. Thus, we are forced to seek effective ways to decompose the problem to assess the performance measures of the system. These cubic architectures occur in communication networks where they employ 3-d mesh networks (of two or more layers) for routing traffic. Software for parallel algorithms in computer science applications often use hypercube topologies (Leighton, 1991).

1.2 *Outline of Paper*

In Sec. 2 of the paper, the problem background and related works are described. The mathematical models appropriate for the optimization approach are presented in Sec. 3, and in Sec. 4, the performance algorithms employed are discussed. In Sec. 5 the experimental results are described and in Sec. 6 the overall results and open questions for future research are presented.

2 Problem Background

As mentioned in Sec. 1, the problem is quite difficult and there have been limited published approaches in the literature regarding this problem. Exact approaches have been limited to the assumptions of exponential distributions, but these continuous time Markov Chain (CTMC) approaches may be limited to moderate sized networks since the state space explodes and often there are complex probabilistic relationships which are not easily understood. However, it is worthwhile mentioning that advances in solving huge Markov Chains were reported recently by Carrasco (2006). Additionally, non-exponential service times within networks are very difficult to analyze exactly, since the memoryless property of exponential distributions no longer applies. Perhaps with few exceptions, such as using Markovian arrival processes (MAP) to model service time distributions, for which one can use simple results to obtain exact blocking probabilities in an $M/MAP/1/K$ system by using matrix-geometric methods (see, for instance, Neuts, 1995), or one of their recent improvements (see Pla and Casares-Giner, 2005), approximations are both reasonable and practical.

Two-moment approximations have been very successful in the past and we shall also follow this approach since it will yield a powerful methodology to approximate the blocking probability for these general networks. A general overview of the literature in this area is given in the tree diagram presented in Fig. 3. In the diagram, we have subdivided the research into those looking at the development of expressions for the blocking probability, p_K, and those studying the buffer allocation problem (BAP) for single and multiple server systems. This paper seeks to unite these two research areas.

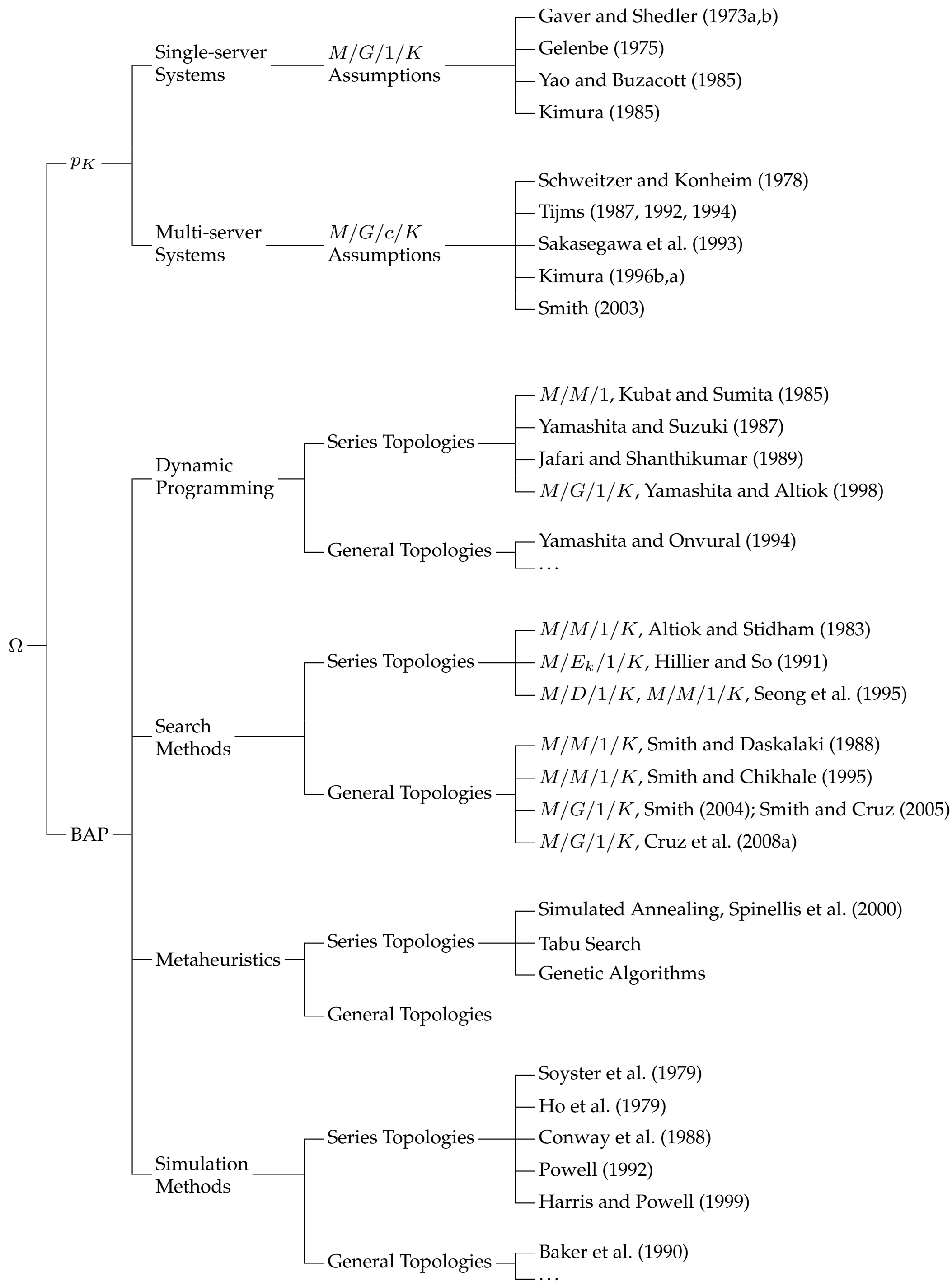

Figure 3: Morphological Diagram of $M/G/c/K$ BAP Approaches

2.1 p_K approaches

Methodologies for approximating the blocking probability, p_K, in $M/G/1/K$ and $M/G/c/K$ systems have a long and detailed history. Exact methods are not feasible for large c and K since the memoryless property of the exponential distribution no longer applies and one must account for how long a customer remains in the queue. Approximations essentially begin with Gelenbe's approach which is based on a diffusion approximation (Gelenbe, 1975). Also, formulas based on the steady-state probabilities of the infinite system by Schweitzer and Konheim (1978), Tijms (1987), and Sakasegawa et al. (1993) have been propagated and, finally, two-moment approximations by Tijms (1992, 1994), Kimura (1996a,b), and Smith (2003, 2004).

2.2 *Buffer Allocation*

The buffer allocation problem also has a long and detailed history as is witnessed by the wealth of authors in Fig. 3. Many optimization approaches have been proposed, based on dynamic programming, search methods, metaheuristics, and simulation-based methods. Since the buffer allocation problem is a solution to an integer, stochastic problem with a nonlinear objective function and constraints (not found in closed form), heuristic approaches have dominated optimal ones. For this problem, then, one needs a robust and accurate optimization procedure with an effective way to measure system performance. This is what is proposed in this paper.

3 Mathematical Models

In this part of the paper we present the optimization model along with the performance models underlying the stochastic measures used in the objective function and constraints of the optimization model. One of the key performance measures is the blocking probability estimation used in the two-moment $M/G/c/K$ queueing model.

3.1 *Assumptions*

We will confine ourselves to Poisson arrival processes because exact results can be derived for these systems. Besides, results for general arrivals are scarce and limited to single servers (see the papers by Choi et al., 2005; Kim and Chae, 2003, for instance).

We also are assuming blocking after service (BAS), sometimes referred to as *production or transfer blocking*, which is a typical protocol in manufacturing and facility planning applications. Although communication networks often assume blocking before service (BBS), or *service blocking*, and sometimes the protocol repetitive blocking (RPB), or *rejection blocking*, the methodology we use assumes BAS. BAS has also been used to model computer systems and disk I/O subsystems. The interested reader is referred to Onvural (1990) and de Nitto Persone (1994) for a good discussion of these blocking mechanisms.

3.2 *Notation*

This section presents most all of the notation we need for the paper.

Variable	Description
Λ	External Poisson arrival rate to the network;
λ_j	Poisson arrival rate to node j;
μ_j	Exponential mean service rate at node j;
c	Number of servers;
$\epsilon \in (0,1)$	Threshold for the blocking probability;
B_j	Buffer capacity at node j *excluding* those in service;
K_j	Buffer capacity at node j *including* those in service;
N	Number of stations in the network;
p_K	Blocking probability of finite queue of size K;
p_0^j	Unconditional probability that there is no customer in the service channel at node j (either being served or being held after service);
$\rho = \lambda/(\mu c)$	Traffic intensity;
s^2	Squared coefficient of variation, $\mathrm{Var}(T_s)/\mathrm{E}(T_s)^2$, of the service time, T_s;
$\mathbf{x}$	Buffer vector of decision variables in optimization routine;
Θ	Mean throughput rate;
Θ^τ	Threshold Mean throughput rate.

3.3 *Mathematical Formulation*

In this paper, we will consider the following type of optimization problem which also was the central objective used in Smith and Cruz (2005)

$$Z = \min f(\mathbf{x}), \tag{1}$$

subject to

$$\begin{aligned} \Theta(\mathbf{x}) &\geq \Theta^\tau, & (2)\\ x_i &\in \{1,2,3,\ldots\},\ \forall i, & (3) \end{aligned}$$

that minimizes the total buffer allocation, $f(\mathbf{x}) = \sum_i x_i$, constrained to provide the minimum throughput Θ^τ. In the above formulation Θ^τ is a threshold throughput value and $x_i \equiv K_i$ is the decision variable, which is the total buffer capacity at the i-th queue *including* those in service.

3.4 *Optimization Search Algorithm*

The primal optimization problem with $M/M/c/K$ and $M/G/c/K$ systems that will be examined here is given by Eq. (1)–Eq. (3). One way to incorporate the throughput constraint, Eq. (2), is through a penalty function approach, such as the Lagrangian relaxation (a recently published tutorial can be found in Lemaréchal, 2007).

Thus, defining a dual variable α and relaxing constraint (2), the following penalized problem is given

$$Z_\alpha = \min \left[\sum_{i=1}^{N} x_i + \alpha \underbrace{\left(\Theta^\tau - \Theta(\mathbf{x})\right)}_{\leq 0} \right], \tag{4}$$

subject to

$$\begin{aligned} x_i &\in \{1,2,\ldots\},\ \forall i, & (5)\\ \alpha &\geq 0. & (6)\end{aligned}$$

Notice that for any feasible vector $\mathbf{x}$ — that is, Eq. (2) and (3) must hold — the term $\alpha\Big(\Theta^\tau - \Theta(\mathbf{x})\Big)$ will be always non-positive and is a penalty of the objective function related to the difference between the threshold throughput, Θ^τ, and the effective throughput, $\Theta(\mathbf{x})$. Thus, it follows that $Z_\alpha \leq Z$, that is, Z_α is an inferior limit for Z, the optimal solution for the primal problem, given by Eq. (1)–Eq. (3).

The Lagrangian relaxation of the primal problem, Z_α, plus an additional relaxation of the integrality constraints for x_i, is a classical unconstrained optimization problem. In the particular formulation of the problem the $x_i, \forall i$ become the decision variables under optimization control. While these are essentially integer variables, they will be approximated by round off from the nonlinear programming solver.

In order to couple the optimization problem with the performance algorithm, which is the Expansion Method described in Sec. 4, Powell algorithm will be used to search for the optimal buffer vector(s) while the Expansion Method computes the performance measure of throughput. Powell's method, as presented in Himmelblau (1972), locates the minimum of $f(\mathbf{x})$ of a non-linear function by successive unidimensional searches from an initial starting point $\mathbf{x}^{(0)}$ along a set of conjugate directions. These conjugate directions are generated within the procedure itself. Powell's method is based on the idea that if a minimum of a non-linear function $f(\mathbf{x})$ is found along p conjugate directions in a stage of the search, and an appropriate step is made in each direction, the overall step from the beginning to the p^{th} step is conjugate to all of the p subdirections of the search. We have had remarkable success in the past with coupling Powell's algorithm and the Expansion Method (Smith and Cruz, 2005).

3.5 Remark on the Lagrange Multiplier

Notice that in order to solve the buffer allocation problem we will set the threshold throughput Θ^τ to the external arrival rate Λ, which will then serve as the input to the approximate performance measure program, the Expansion Method (Kerbache and Smith, 1988), that will compute the corresponding throughput $\Theta(\mathbf{x})$.

Thus, the best (highest) possible inferior limit is given by

$$Z_\alpha^* = \max_{\alpha\geq 0} Z_\alpha,$$

which is achieved for $\alpha^* \to \infty$, which follows from $\Theta(\mathbf{x})$ being a non-decreasing function of $\mathbf{x}$, the input arrival rate λ being exactly the threshold throughput Θ^τ, and, finally, from the property of the Lagrangian function, Z_α, being the minimum of linear functions of α,

$$Z_\alpha = \min\left(\underbrace{\sum_{i=1}^N x_i}_{\text{intercept}} + \alpha\overbrace{\Big(\Theta^\tau - \Theta(\mathbf{x})\Big)}^{\text{slope}}\right),$$

with non-negative intercepts and slopes with

$$\lim_{\mathbf{x}\to\infty}\Big(\Theta^\tau - \Theta(\mathbf{x})\Big) = 0.$$

The best Lagrange multiplier α defined previously is not practical because one would need that

$$\Big(\Theta^\tau - \Theta(\mathbf{x})\Big) = 0,$$

which yields $x_i \to \infty, \forall\ i$. On the other hand, if a small difference, say $\Big(\Theta^\tau - \Theta(\mathbf{x})\Big) = \varepsilon$, is acceptable, it must hold that

$$\alpha\Big(\Theta^\tau - \Theta(\mathbf{x})\Big) \in [0,1],$$

because, otherwise, it might be better to spend one more unity of buffer space to some i-th queue to increase the throughput (remind that $\Theta(\mathbf{x})$ is a non-decreasing function of $\mathbf{x}$). Thus, it is possible to define a corresponding α_ε as follows

$$\alpha_\varepsilon \leq 1/\Big(\Theta^\tau - \Theta(\mathbf{x})\Big),$$

which yields $\alpha_\varepsilon > 10^3$ for $\Big(\Theta^\tau - \Theta(\mathbf{x})\Big) \leq 10^{-3}$.

In the following subsections Sec. 3.6–3.8, we present our approach for estimating the blocking probabilities, since the blocking probabilities are central to the performance algorithm. We also illustrate a convexity property capacity of finite queues which is important in the buffer allocation process. Then in Sec. 4, we present the Expansion method for estimating the performance measures in the queueing networks.

3.6 $M/G/1/K$ p_K Expression

The blocking probability for an $M/M/1/K$ system with $\rho < 1$ is well-known

$$p_K = \frac{(1-\rho)\rho^K}{1-\rho^{K+1}}. \quad (7)$$

If we relax the integrality of K, we can express K in terms of ρ and p_K and arrive at a closed-form expression for the buffer size which is the largest integer as follows

$$K = \left\lceil \frac{\ln\left(\frac{p_K}{1-\rho+p_K\rho}\right)}{\ln(\rho)} \right\rceil,$$

in which $\lceil x \rceil$ is the smallest integer not inferior to x.

In two previous papers (Smith, 2003; Smith and Cruz, 2005), we showed that once we have the closed form expression for the optimal pure buffer $B^* = K^* - 1$ in an $M/M/1/K$ system (*excluding those on service*), we can use a two-moment approximation scheme based on Tijms' (Tijms, 1987, 1992, 1994) and Kimura's (Kimura, 1996a,b) works to calculate the optimal buffer size B^* for general service. For a single server, $c = 1$, and general squared coefficient of variation of service time s^2, we have an approximation to the optimal buffer size B^* for $M/G/1/K$ systems

$$B^* = -\frac{\left(-\ln\left(\frac{p_K}{1-\rho+p_K\rho}\right)+\ln(\rho)\right)\left(2+\sqrt{\rho}s^2-\sqrt{\rho}\right)}{2\ln(\rho)}.$$

If $s^2 = 1$, then the formula yields the same expression as for the $M/M/c/K$ $c = 1$ formula, when we subtract the space for the server.

As an added side benefit for developing the closed form expression for the optimal buffer, if we invert the last expression we can obtain the blocking probability for the $M/G/1/K$ system as

$$p_K = \frac{\rho^{\left(\frac{2+\sqrt{\rho}s^2-\sqrt{\rho}+2B}{2+\sqrt{\rho}s^2-\sqrt{\rho}}\right)}(-1+\rho)}{\rho^{\left(2\frac{2+\sqrt{\rho}s^2-\sqrt{\rho}+B}{2+\sqrt{\rho}s^2-\sqrt{\rho}}\right)}-1}.$$

One caveat of our approach, however, is that we do not have an explicit formula for the case when $\rho = 1$. This is because the original blocking probability formula for the $M/M/1/K$ formula does not include ρ in its calculation for the case when $\rho = 1$. Thus, we must linearly interpolate the value when $\rho = 1$, albeit a crude approximation, but not extraordinary difficult. This is not seen as a major issue, however, since as we shall describe below, $\rho = 1$ will be a limiting value for allocating buffers.

3.7 $M/G/c/K$ p_K Expressions

As one might expect, we can continue this process of developing p_K since one can obtain B^* and p_K for different values of c and thus develop closed form expressions of the buffer size and blocking probabilities for $M/G/c/K$ systems. Let us examine a multi-server Markovian system.

3.8 $M/M/c/K$ p_K Expression

Analytical results from the $M/M/c/K$ model provide the following expression for p_K

$$p_K = \frac{1}{c^{K-c}c!}\left(\frac{\lambda}{\mu}\right)^K p_0,$$

where for $\lambda/(c\mu) \neq 1$

$$p_0 = \left[\sum_{n=0}^{c-1}\frac{1}{n!}\left(\frac{\lambda}{\mu}\right)^n + \frac{(\lambda/\mu)^c}{c!}\frac{1-[\lambda/(c\mu)]^{K-c+1}}{1-\lambda/(c\mu)}\right]^{-1}.$$

The $M/M/c/K$ composite formula for the blocking probability is given by

$$p_K = \frac{\left(\frac{\lambda}{\mu}\right)^K (c!)^{-1}\left(c^{K-c}\right)^{-1}}{\left(e^{\frac{\lambda}{\mu}}\Gamma(c,\frac{\lambda}{\mu})\Gamma^{-1}(c)+\frac{\left(\frac{\lambda}{\mu}\right)^c\left(1-\left(\frac{\lambda}{c\mu}\right)^{K-c+1}\right)}{(c!)\left(1-\frac{\lambda}{c\mu}\right)}\right)}.$$

Since the blocking probability function is very complex, one cannot express the value of K without fixing c. If one fixes $c = 1$, the function becomes the same as the one which was derived from the previous formula for the $M/M/1/K$ system, Eq. (7).

3.9 Development and Rationale for Approximations

One of the keys linking the blocking probability models to the buffer allocation problem can be derived from the blocking probability formula for different values of c. Let us do this in some detail for the $M/G/c/K$ case with $c = 2$.

If we examine the optimal buffer size as a function of p_K, ρ, λ and μ, relaxing its integrality, we achieve for $c = 2$

$$B^* = \frac{\left(2+\sqrt{\frac{\rho}{e}}s^2-\sqrt{\frac{\rho}{e}}\right)\left(\ln\left(\frac{p_K(2\mu+\lambda)}{2(2\mu-\lambda+p_K\lambda)}\right)-2\ln(\rho)\right)}{2\ln(\rho)}.$$

Now let us graph this function of B^* vs. ρ with the parameters fixed at the following values: $p_K = 0.001$, $\lambda = 5$, $\mu = 10$ along with $s^2 = \{0, \frac{1}{2}, 1, \frac{3}{2}, 2\}$. Then we achieve the graph in Fig. 4. This graph illustrates the optimal buffer allocation as a function of the traffic intensity. The middle curve with $s^2 = 1$ is exact as it is based on the exact expression for the $M/M/2/K$ formula with fixed parameters $p_K = 0.001$, $\lambda = 5$, $\mu = 10$. The other curves, which parallel but do not overlap the middle curve, are the approximations to the optimal buffer size achieved with the two-moment approximation. Thus, in all that follows, when we vary c we will have this convex relationship between the approximations for s^2 and the blocking probabilities for c. This is a very important property that unifies our approach for the multi-server buffer allocation problem. We now need to describe how we will incorporate our expression for p_K into a procedure for modeling and optimizing finite queueing networks with multiple servers.

4 Performance Algorithm

The Expansion Method is a robust and effective approximation technique developed by Kerbache and Smith (1987) to analyze finite queueing networks. As described in previous papers, this method is characterized as a combination of repeated trials and node-by-node decomposition solution procedures. The Expansion Method uses BAS type blocking, which is prevalent

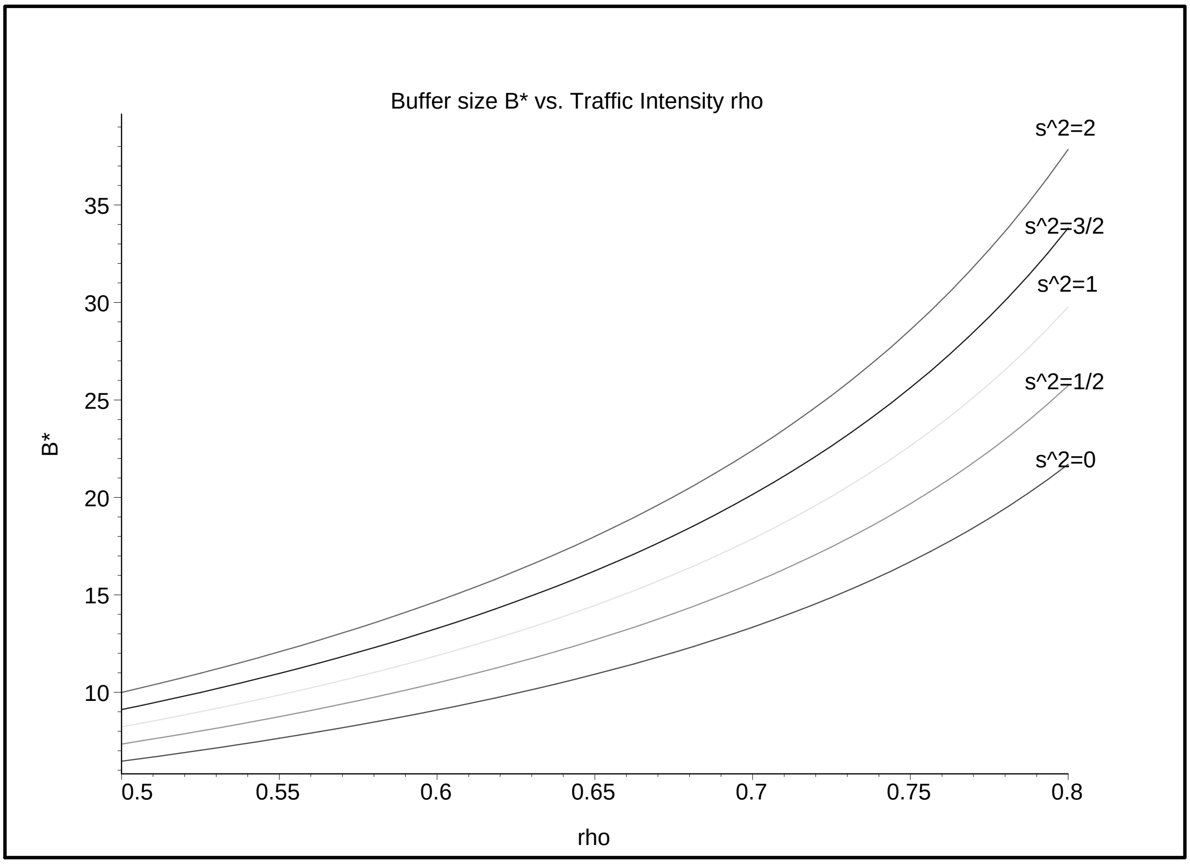

Figure 4: B^* vs. ρ for $c = 2$

in most production and manufacturing, transportation and other similar systems.

Consider a single node with finite capacity K (including service). This node essentially oscillates between two states — the saturated phase and the unsaturated phase. In the unsaturated phase, node j has at most $K - 1$ customers (in service or in the queue). On the other hand, when the node is saturated no more customers can join the queue. Refer to Fig. 5 for a graphical representation of the two scenarios.

The Expansion Method has the following three stages:

- *Stage I: Network Reconfiguration;*
- *Stage II: Parameter Estimation;*
- *Stage III: Feedback Elimination.*

The following additional notation defined by Kerbache and Smith (1987, 1988) shall be used in further discussion regarding this methodology.

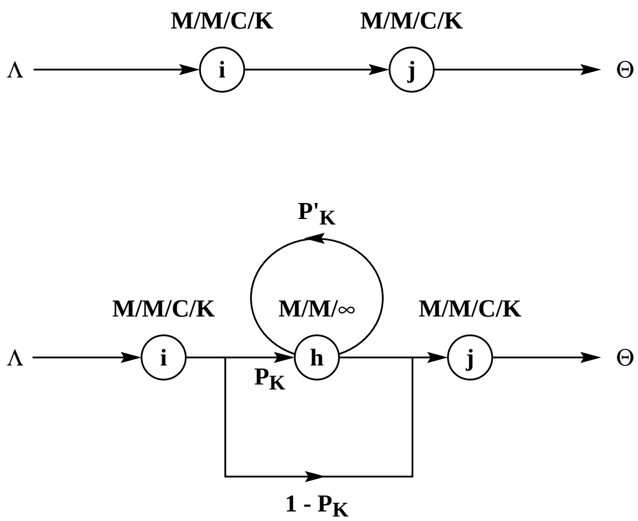

Figure 5: Type I Blocking in Finite Queues

Variable	Description
h	The holding node established in the Expansion Method;
$\tilde{\lambda}_j$	Effective arrival rate to node j;
$\tilde{\mu}_j$	Effective service rate at node j due to blocking;
p'_K	Feedback blocking probability in the Expansion Method.

4.1 Stage I: Network Reconfiguration

Using the concept of two phases at node j (saturated and unsaturated), an artificial node h is added for each finite node in the network to register blocked customers. Fig. 5 shows the additional delay, caused to customers trying to join the queue at node j when it is full, with probability p_K. The customers successfully join queue j with a probability $(1 - p_K)$. Introduction of an artificial node also dictates the addition of new arcs with p_K and $(1 - p_K)$ as the routing probabilities.

The blocked customer proceeds to the finite queue with probability $(1 - p'_K)$ once again after incurring a delay at the artificial node. If the queue is still full, it is re-routed with probability p'_K to the artificial node where it incurs another delay. This process continues until it finds a space in the finite queue. A feedback arc is used to model the repeated delays. The artificial node is modeled as an $M/M/\infty$ queue. The infinite number of servers is used simply to serve the blocked customer a delay time without queueing.

4.2 Stage II: Parameter Estimation

This stage essentially estimates the parameters p_K, p'_K and μ_h utilizing known results for the $M/M/c/K$ model.

- p_K: Utilizing our analytical results for the $M/G/2/K$ model provides the following expression for p_K (Smith, 2003)

$$p_K = \frac{2\rho^{2\left(\frac{(2+\sqrt{\rho/e}\,s^2-\sqrt{\rho/e}\,+B)}{(2+\sqrt{\rho/e}\,s^2-\sqrt{\rho/e})}\right)}(2\mu-\lambda)}{-2\rho^{2\left(\frac{(2+\sqrt{\rho/e}\,s^2-\sqrt{\rho/e}\,+B)}{(2+\sqrt{\rho/e}\,s^2-\sqrt{\rho/e})}\right)}\lambda+2\mu+\lambda}.$$

  Similarly, other expressions for $c = 3, \ldots, 10$ may be included here so that we will have a complete set of blocking probabilities for $c \in [1, 10]$;

- p'_K: Since there is no closed form solution for this quantity an approximation is used given by Labetoulle and Pujolle and obtained using diffusion techniques (Labetoulle and Pujolle, 1980)

$$p'_K = \left\{ \frac{\mu_j + \mu_h}{\mu_h} - \frac{\lambda\left[\left(r_2^K - r_1^K\right) - \left(r_2^{K-1} - r_1^{K-1}\right)\right]}{\mu_h\left[\left(r_2^{K+1} - r_1^{K+1}\right) - \left(r_2^K - r_1^K\right)\right]} \right\}^{-1},$$

  where r_1 and r_2 are the roots to the polynomial

$$\lambda - (\lambda + \mu_h + \mu_j)x + \mu_h x^2 = 0,$$

  and where $\lambda = \lambda_j - \lambda_h(1 - p'_K)$ and λ_j and λ_h are the actual arrival rates to the finite and artificial holding nodes respectively. Labetoulle and Pujolle (1980) illustrate in their paper a comparison of their method for computing p'_K with Erlang and hyper-exponential service time distributions and it is shown that the calculation for p'_K is very reasonable for these general service systems. Given these results, we felt comfortable in applying p'_K for the general service situation.

In fact, the arrival rate to the finite node j is given by

$$\lambda_j = \tilde{\lambda_i}(1 - p_K) = \tilde{\lambda_i} - \lambda_h.$$

Let us examine the following argument to determine the service time at the artificial node. If an arriving customer is blocked, the queue is full and a customer is being serviced. Thus the arriving customer to the holding node has to remain in service at the artificial holding node for the remaining service time interval of the customer in service. The delay distribution of a blocked customer at the holding node has the same distribution as the remaining service time of the customer being serviced. Using renewal theory, one can show that the remaining service has the following rate μ_h

$$\mu_h = \frac{2\mu_j}{1 + \sigma_j^2 \mu_j^2},$$

where σ_j^2 is the service time variance given by Kleinrock (1975). Notice that if the service time distribution at the finite queue doing the blocking is exponential with rate μ_j, then

$$\mu_h = \mu_j,$$

i.e., the service time at the artificial node is also exponentially distributed with rate μ_j. When the service time of the blocking node is not exponential, then μ_h will be affected by σ_j^2.

4.3 Stage III: Feedback Elimination

Due to the feedback loop around the holding node, there are strong dependencies in the arrival processes. Elimination of these dependencies requires reconfiguration of the holding node which is accomplished by recomputing the service time at the node and removing the feedback arc. The new service rate is given by

$$\mu'_h = (1 - p'_K)\mu_h.$$

The probabilities of being in each of the two phases (saturated or unsaturated) are p_K and $(1 - p_K)$. The mean service time at a node i preceding the finite node is μ_i^{-1} when in the unsaturated phase and $(\mu_i^{-1} + \mu_h'^{-1})$ in the saturated phase. Thus, on an average, the mean service time at the node i preceding a finite node is given by

$$\tilde{\mu}_i^{-1} = \mu_i^{-1} + p_K \mu_h^{-1}.$$

Similar equations can be established with respect to each of the finite nodes. Ultimately, we have simultaneous non-linear equations in variables p_K, p'_K, μ_h^{-1} along with auxiliary variables such as μ_j and $\tilde{\lambda}_i$. Solving these equations simultaneously we can compute all the performance measures of the network

$$\begin{aligned}
\lambda &= \lambda_j - \lambda_h(1 - p'_K), && (8)\\
\lambda_j &= \tilde{\lambda}_i(1 - p_K), && (9)\\
\lambda_j &= \tilde{\lambda}_i - \lambda_h, && (10)\\
p'_K &= \left\{ \frac{\mu_j + \mu_h}{\mu_h} - \right. && (11)\\
& \left. \frac{\lambda\left[\left(r_2^K - r_1^K\right) - \left(r_2^{K-1} - r_1^{K-1}\right)\right]}{\mu_h\left[\left(r_2^{K+1} - r_1^{K+1}\right) - \left(r_2^K - r_1^K\right)\right]} \right\}^{-1}, && (12)\\
z &= (\lambda + 2\mu_h)^2 - 4\lambda\mu_h, && (13)\\
r_1 &= \frac{\left[(\lambda + 2\mu_h) - z^{\frac{1}{2}}\right]}{2\mu_h}, && (14)\\
r_2 &= \frac{\left[(\lambda + 2\mu_h) + z^{\frac{1}{2}}\right]}{2\mu_h}, && (15)\\
p_K &= \frac{2\rho^{2\left(\frac{(2+\sqrt{\rho/e}\,s^2 - \sqrt{\rho/e} + B)}{(2+\sqrt{\rho/e}\,s^2 - \sqrt{\rho/e})}\right)}(2\mu - \lambda)}{-2\rho^{2\left(\frac{(2+\sqrt{\rho/e}\,s^2 - \sqrt{\rho/e} + B)}{(2+\sqrt{\rho/e}\,s^2 - \sqrt{\rho/e})}\right)}\lambda + 2\mu + \lambda}. && (16)
\end{aligned}$$

Equations (8) to (12) are related to the arrivals and feedback in the *holding* node. Equations (13) to (15) are used for solving Eq. (12) with z used as a dummy parameter for simplicity of the solution. Lastly, Eq. (16) gives the approximation to the blocking probability for the $M/G/2/K$ queue. Other expressions for p_K for $c = 3, 4, 5, 10$ are also utilized in the experiments to follow. Hence, we essentially have five equations to solve, *viz.* Eq. (8)–Eq. (12) and Eq. (16).

To recapitulate, we first expand the network; followed by approximation of the routing probabilities, due to blocking, and the service delay in the holding node and finally the feedback arc at the holding node is eliminated. Once these three stages are complete, we have an expanded network which can then be used to compute the performance measures for the original network. As a decomposition technique this approach allows successive addition of a holding node for every finite node, estimation of the parameters and subsequent elimination of the holding node.

Fig. 6 and 7 illustrate the process of expanding the network topologies for the merge and split topologies in the Expansion Method. An important point about this process is that we do not physically modify the networks, only represent the expansion process through the software.

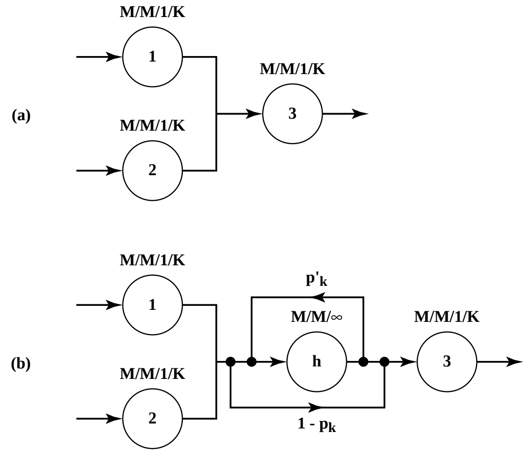

Figure 6: Merge Topologies

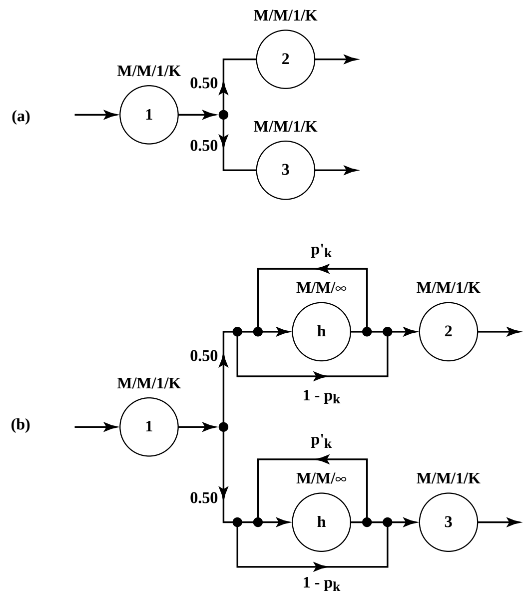

Figure 7: Split Topologies

5 Experimental Results

In this section of the paper, we provide computational results of the network design methodology for multi-server finite queueing networks. First, we will develop results for 2-node, 3-node. Then larger and more complex networks will be analyzed. We also examine the order of the servers and what effects the squared coefficient of variation s^2 has on symmetric, asymmetric, and bottleneck networks. Since the range of possible experiments is exponential itself, we have determined a select sample to present.

5.1 2-server/2-node Networks

The simplest network is a 2-server/2-node topology involving single and 2-servers arranged in a simple series connection, as seen in Fig. 8. One would like to test what are the buffers needed for this type of topology and whether one topology (*i.e.*, server order) is better than another.

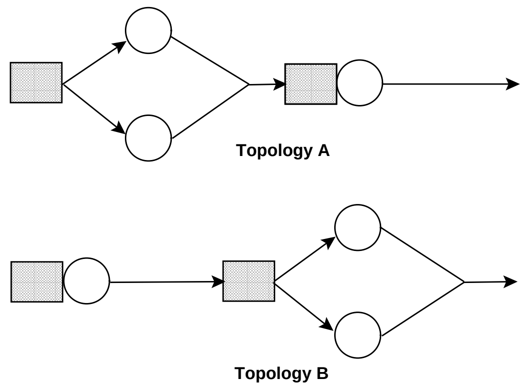

Figure 8: 2-server Network Topology

In our first experiment, we fix the arrival rate to the network with $\lambda = 5$ and service rates of the different servers to $\mu = 10$. We would like to examine what buffers are needed for these two alternative network topologies. We will also vary the coefficient of variation of the service times to see how the buffer is affected by the service time variability s^2. In order to evaluate the analytical results, simulation runs of 20 replications, with a warm up period of 20,000 time units (Robinson, 2007), and 100,000 time units for each run were carried out. These run lengths and number of replications were necessary in order to reduce the standard deviation of the statistics of the simulation output to a reasonably accurate level. In order to model the general service times for the $s^2 = \{1/2, 3/2\}$, Gamma distributions were used in the simulation model representation of the service time distributions. The computer was a CyberPower PC with AMD AthlonXP 1800+ 1.53 GHz with 512 MB of RAM, running Windows XP. In the tables that follow, the δ designation in the 7^{th} column of the simulation tables refers to the half-width of the 95% c.i.

Table 1: 2-server/2-node Results

					Simulation		
s^2	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$	$\theta(\mathbf{x})^s$	δ	Z_α^s
$1/2$	(2,1)	4.997	19.15	(8,8)	4.994	0.004	22.20
	(1,2)	4.997	19.15	(8,8)	4.997	0.002	19.40
1	(2,1)	4.997	21.84	(9,10)	4.994	0.004	25.50
	(1,2)	4.997	21.85	(10,9)	4.994	0.003	24.60
$3/2$	(2,1)	4.996	24.51	(10,11)	4.996	0.003	25.10
	(1,2)	4.996	24.51	(11,10)	4.996	0.002	24.60

The results in general are pretty encouraging, as seen in Tab. 1. In all cases, the analytical throughput value $\Theta(\mathbf{x})$ was within the 95% c.i. simulated value $\Theta(\mathbf{x})^s$. The buffer allocations are symmetric for all cases, and there is little difference in the optimal solution values for either topology. Thus, it is difficult to say whether one topology is better than another, simply because the optimization methodology made sure that the resulting buffer allocations were appropriate for each of the topologies. If one did not optimize the buffer allocations, then perhaps one topology might dominate the other. However, it is difficult to derive heuristic rules (*e.g.* always place the multi-servers first in the topology) prior to an optimization procedure to say which topology is better. The analytical throughput value $\theta(\mathbf{x})$ was within the 95% ci simulated value $\theta(\mathbf{x})^s$ in all cases.

Table 2: 2-server/2-node Bottleneck Results

					Simulation		
s^2	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$	$\theta(\mathbf{x})^s$	δ	Z_α^s
$1/2$	(2,1)	6.994	39.85	(20,14)	6.991	0.003	42.80
	(1,2)	6.994	39.84	(14,20)	6.991	0.004	43.10
1	(2,1)	6.993	46.58	(22,18)	6.990	0.004	49.60
	(1,2)	6.993	46.58	(18,22)	6.991	0.004	49.40
$3/2$	(2,1)	6.993	53.13	(25,21)	6.992	0.004	53.70
	(1,2)	6.993	53.12	(21,25)	6.989	0.005	56.60

In another experiment with 2-node networks, let us argue that the service time of the two-server node is smaller than the service time of the single server queue (see Fig. 8). This represents a bottleneck situation. Let us also argue that the service time of the 2-server queue has $\mu = 9$ while the service time at the single-server queue is $\mu = 10$. Finally, we will increase the arrival rate $\lambda = 7$ to force more buffers to be allocated. We get the experimental results presented in Tab. 2.

As in previous experimental results, Tab. 2 indicates that dramatically more buffer is allocated to the 2-server nodes rather than less since they represent the bottlenecks. Symmetric buffer allocations occur and little difference occurs in the objective function values of the topologies, so it is difficult to say which topology is better. Additionally, $\theta(\mathbf{x})$ is within the 95% c.i. of the $\theta(\mathbf{x})^s$ for all the simulation runs.

5.2 *3-node/2-server Networks*

Extending our approach to more complex network topologies, we examine a 3-node/2-server single queue network. Fig. 9 represents the possible topologies with one single server and two 2-server queues in a series topology.

Table 3: 3-node/2-server Results

					Simulation		
s^2	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$	$\theta(\mathbf{x})^s$	δ	Z_α^s
$1/2$	(1,2,2)	6.992	51.90	(14,15,15)	6.990	0.004	53.80
	(2,1,2)	6.992	51.90	(15,14,15)	6.991	0.003	53.40
	(2,2,1)	6.992	51.90	(15,15,14)	6.990	0.003	54.20
1	(1,2,2)	6.992	59.98	(18,17,17)	6.989	0.004	63.00
	(2,1,2)	6.992	59.98	(17,18,17)	6.993	0.004	59.20
	(2,2,1)	6.992	59.98	(17,17,18)	6.993	0.003	59.50
$3/2$	(1,2,2)	6.990	67.88	(20,19,19)	6.989	0.004	68.60
	(2,1,2)	6.990	67.88	(19,20,19)	6.993	0.003	65.30
	(2,2,1)	6.990	67.88	(19,19,20)	6.990	0.004	67.90

It is interesting that for the low s^2 in Tab. 3, the buffer at the single server is reduced in relation to the 2-server

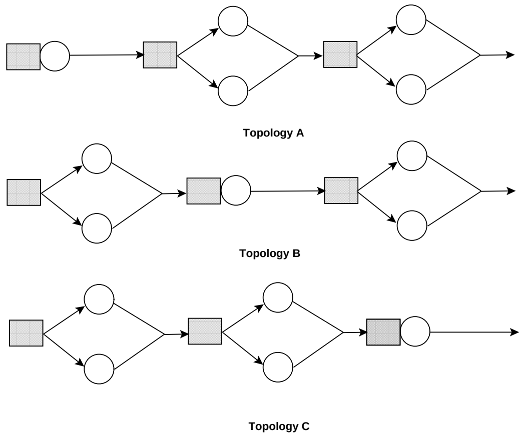

Figure 9: 3-node/2-server Network Topology

nodes while in the higher $s^2 = \{1, 3/2\}$, the buffer at the single node is increased over that of the 2-servers. Thus, the effects of variability through the s^2 are critically important in the buffer allocation.

Once again, it is difficult to say which topology is best since all the topologies are optimized, although, Topology B seems to fare a bit better than either A or C, especially when one examine the simulation results. In one sense, this seems intuitively correct as the single server is buffered by the two multi-server nodes. In all the experiments so far, all the analytical throughput results in Tab. 3 are within the 95% c.i. of the simulation results, which is very encouraging.

To recap what we have found with 2-servers, let us increase the number of servers in this topology for $c = 3, 4, 10$ and see how the buffer allocations evolve. These are presented in Tab. 4 without the corresponding simulations in order to examine the tradeoffs between the $\theta(\mathbf{x})$, Z_α, and the buffers, number of servers, and affects of s^2.

The results in Tab. 4 are interesting. In the first set of experiments, for $s^2 = 1/2$, when $c = 3$, the buffer at the single server is smaller than at the 3-server node. When $c = 4$, the result flips, so the buffer at the single-server node increases while the buffer at the four-server node decreases relative to the one at the single-server node. For $c = 10$, the buffer at the ten-server again is smaller than the single-server node, but not dramatically different.

In the second set of experiments with $s^2 = 1$, the buffer at the single server-node is always larger than the buffer for the four-server and ten-server nodes respectively. Finally, for $s^2 = 3/2$, the difference between the buffer at the single-server node and the multi-server node widens and this is because of the increased variability in the service times. In all cases, the objective function value decreases with increasing numbers of servers.

In order to more finely determine the effect of the s^2 on the buffer allocation, let us isolate one server configuration $c = (1, 3, 3)$ and vary the squared coefficient of variation s^2 to see how the buffer allocation changes with increasing $s^2 \in [0, 1]$. Tab. 5 illustrates this nonlinear changing process. When $s^2 = 0$, the buffer difference is 2 at the single server node in relation to the 3-server nodes, and then eventually changes between $s^2 = 0.80 - 0.90$ when the buffer at the single node flips and becomes greater than the 3-server nodes. This is very interesting and quite unpredictable. This is why the tool we have developed is so useful as opposed to pre-determined rules for allocating buffers since they are susceptible to slight changes in the system parameters.

One concern registered here is that we have not included the cost of the service. This is highly dependent on the application and will be a subject of future research.

Table 4: 3-node/c-server Variation Results

s^2	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$
1/2	(1,3,3)	6.993	51.14	(14,15,15)
	(3,1,3)	6.993	51.14	(15,14,15)
	(3,3,1)	6.992	51.14	(15,15,14)
	(1,4,4)	6.992	50.57	(15,14,14)
	(4,1,4)	6.992	50.57	(14,15,14)
	(4,4,1)	6.992	50.57	(14,14,15)
	(1,10,10)	6.993	47.22	(14,13,13)
	(10,1,10)	6.993	47.22	(13,14,13)
	(10,10,1)	6.993	47.22	(13,13,14)
1	(1,3,3)	6.993	59.17	(18,17,17)
	(3,1,3)	6.993	59.17	(17,18,17)
	(3,3,1)	6.993	59.17	(17,17,18)
	(1,4,4)	6.992	58.33	(15,14,14)
	(4,1,4)	6.992	58.33	(14,15,14)
	(4,4,1)	6.992	58.33	(14,14,15)
	(1,10,10)	6.991	54.66	(18,14,14)
	(10,1,10)	6.991	54.66	(18,14,14)
	(10,10,1)	6.991	54.66	(18,14,14)
3/2	(1,3,3)	6.990	66.87	(21,18,18)
	(3,1,3)	6.990	66.87	(18,21,18)
	(3,3,1)	6.990	66.87	(18,18,21)
	(1,4,4)	6.991	65.98	(21,18,18)
	(4,1,4)	6.991	65.98	(18,21,18)
	(4,4,1)	6.991	65.98	(18,18,21)
	(1,10,10)	6.991	61.83	(21,16,16)
	(10,1,10)	6.991	61.83	(16,21,16)
	(10,10,1)	6.991	61.83	(16,16,21)

Table 5: 3-node/c-server s^2 Variation Results

s^2	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$
0.00	(1,3,3)	6.994	42.847	(11,13,13)
0.10	(1,3,3)	6.995	44.718	(12,14,14)
0.20	(1,3,3)	6.994	46.267	(12,14,14)
0.30	(1,3,3)	6.994	48.006	(14,14,14)
0.40	(1,3,3)	6.992	49.553	(14,14,14)
0.55	(1,3,3)	6.993	51.910	(15,15,15)
0.60	(1,3,3)	6.992	52.704	(15,15,15)
0.70	(1,3,3)	6.994	54.417	(16,16,16)
0.80	(1,3,3)	6.992	55.921	(16,16,16)
0.90	(1,3,3)	6.992	57.496	(17,16,16)

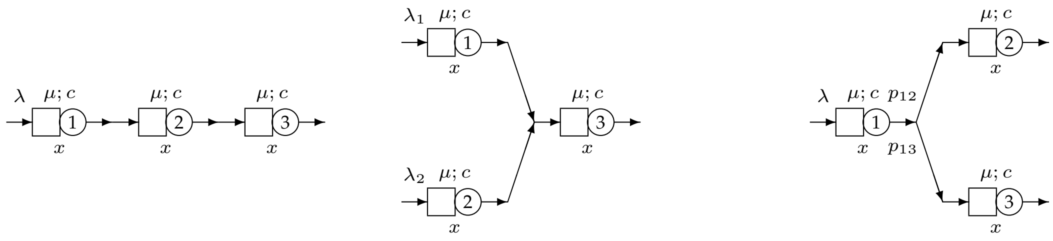

Figure 10: Basic Topologies Examined

Table 6: Symmetric Network Comparison Results

Topology	s^2	λ	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$	Simulation		
							$\theta(\mathbf{x})^s$	δ	Z_α^s
series	1/2	2.0	(2,2,2)	1.998	10.80	(3,3,3)	1.998	0.002	11.1
		4.0	(4,4,4)	3.996	19.10	(5,5,5)	3.995	0.003	20.5
		8.0	(10,10,10)	7.989	71.14	(20,20,20)	7.993	0.003	67.3
	1	2.0	(2,2,2)	1.997	11.55	(3,3,3)	1.996	0.002	13.2
		4.0	(4,4,4)	3.997	20.67	(6,6,6)	3.995	0.003	22.7
		8.0	(10,10,10)	7.986	80.32	(22,22,22)	7.991	0.003	75.0
	3/2	2.0	(2,2,2)	1.997	12.47	(3,3,3)	1.997	0.002	11.9
		4.0	(4,4,4)	3.996	22.31	(6,6,6)	3.996	0.003	21.9
		8.0	(10,10,10)	7.986	89.28	(25,25,25)	7.996	0.003	79.0
split	1/2	2.0	(2,2,2)	1.999	7.86	(3,2,2)	1.998	0.002	9.4
		4.0	(4,4,4)	3.999	14.40	(5,4,4)	3.997	0.003	15.9
		8.0	(10,10,10)	7.996	44.76	(21,10,10)	7.993	0.003	47.7
	1	2.0	(2,2,2)	1.999	8.18	(3,2,2)	1.997	0.002	8.4
		4.0	(4,4,4)	3.999	14.94	(6,4,4)	3.995	0.003	18.6
		8.0	(10,10,10)	7.995	46.85	(23,10,10)	7.992	0.003	51.3
	3/2	2.0	(2,2,2)	1.998	8.56	(3,2,2)	1.996	0.002	10.6
		4.0	(4,4,4)	3.998	15.52	(6,4,4)	3.996	0.003	17.9
		8.0	(10,10,10)	7.995	49.84	(25,10,10)	7.993	0.004	51.8
merge	1/2	2.0	(2,2,2)	1.999	7.86	(2,2,3)	1.996	0.002	11.0
		4.0	(4,4,4)	3.999	14.40	(4,4,5)	3.996	0.002	17.2
		8.0	(10,10,10)	7.996	43.76	(10,10,20)	7.989	0.004	50.9
	1	2.0	(2,2,2)	1.999	8.18	(2,2,3)	1.997	0.001	9.7
		4.0	(4,4,4)	3.999	14.94	(4,4,6)	3.995	0.002	19.2
		8.0	(10,10,10)	7.995	46.85	(10,10,22)	7.990	0.004	51.6
	3/2	2.0	(2,2,2)	1.998	8.56	(2,2,3)	1.996	0.002	10.6
		4.0	(4,4,4)	3.998	15.52	(4,4,6)	3.996	0.002	17.6
		8.0	(10,10,10)	7.995	49.84	(10,10,25)	7.989	0.004	55.7

5.3 Series, Merge, and Split networks

Now let us examine additional experiments for series, merge ($\lambda_1 = \lambda_2 = 0.5\lambda$), and splitting ($p_{12} = p_{13} = 0.5$) topologies of networks (see Fig. 10), in which we vary the number of servers $c = \{2, 4, 10\}$, along with $s^2 = \{1/2, 1, 3/2\}$ as before and λ which is allowed to vary from a low of $2 \rightarrow 8$. The ranges on the number of servers shows what is possible with our methodology. We will examine here uniform server allocations at all the nodes in the networks. Tab. 6 illustrates the range of buffer allocations for these network topologies. Simulation results for the different topologies are included to reveal the scope and limitations of this optimization approach. All the service times for the $s^2 = \{1/2, 3/2\}$ were from Gamma distributions. In all these tables, $\mu_i = 10\ \forall i$ nodes in the network and we restricted the buffer to be $\geq c_i, \forall i$ nodes. Thus, for the low traffic cases $\lambda = 2, 4$ and $c = 4, 10$ a zero buffer allocation is sometimes warranted. We will discuss this result in more detail in the following section of the paper.

In general, the results are intuitively appealing as they correspond to the number of servers, the traffic levels, and the effects of s^2. Symmetric uniform buffer allocations result in all the topologies, so it appears that the optimization procedures are working correctly and are unaffected by the different topologies. One surprising thing perhaps is that the effect of the multiple servers at the nodes is not as significant in improving the performance measures of the network as the buffers themselves. Witness the allocation of buffers to the single servers in relation to the buffers allocated to the multiple servers in Tab. 4 (for an interesting paper about symmetries in closed queueing networks with finite topologies and predicting their performance, see de Nitto Persone, 1994).

In Tab. 6 one can see that the buffer patterns are pretty uniform as might be expected. Once the topology of 3-nodes is established, the uniform allocation of buffers proceeds as expected. For the split topologies, similar buffer patterns emerge as in the series experiments. Since $\mu_i = 10\ \forall i$ nodes, the split node receives the most buffers since the traffic to the two leaf nodes is amply reduced. In the simulation experiments for the split topologies, similar results hold as for the series simulation experiments although the costs of supplying the buffers is reduced for the split topologies. Again, for the merging topologies, similar results occur as in the series and split topologies. The merge topologies objective function is a little less than for the split and series topologies and in a way this is surprising because the blocking levels in the merge networks are higher than either the series or split topologies.

5.4 Zero Buffer Networks

Note that in Tab. 6, where the buffer sizes in $\mathbf{x}$ are equal to the number of servers in $\mathbf{c}$, one obtains a zero buffer system. In such a system, no buffer space is allocated to the servers, so that a job can only enter a workstation when an empty server of that station is available. The reason for this result in the experimental results is that the traffic arrival rate does not justify the use of the multiple servers, so only zero buffers make sense.

For small tandem line networks, exact solutions are available: small asynchronous single server tandem lines with zero buffer and reliable machines are partly analyzed by Hildebrand (1968), Hillier and Boling (1967), Muth and Alkaff (1987), and Rao (1976a,b). Most of the exact results analysis are based on continuous time Markov processes. Consequently, the inter-arrival and service time need to exponentially or phase-type distributed to be able to solve the network. The analysis of large state space Markov process models, the computational time and the complexity for the techniques motivates the need for approximation techniques such as the Expansion method. As such, the presented results in this paper are an extension to any size of arbitrarily configured networks with general service times and zero buffers.

5.5 Asymmetric Networks

So far most of the cases considered were "symmetric" and someone might argue that such networks are not quite realistic. Thus we set up some asymmetric networks unbalancing the routing probabilities ($p_{12} = 0.6$ and $p_{13} = 0.4$, for the split topologies) and the arrival rates ($\lambda_1 = 0.6\lambda$, $\lambda_2 = 0.4\lambda$, for the merge topologies), and assuming different service rates, μ's, and number of servers, c's. The results are quite impressive and may be seen in Tab. 7. The service times were from Gamma distributions. The results show a close but not perfect agreement between the analytical and simulation results.

The results are sound because they show that the buffer allocation is stable and that the algorithms proposed also work for asymmetric networks. They also show we get zero buffer solutions if necessary (see in the tables those cases for which $x_i = c_i$ for some i). Additionally, some of the patterns observed for symmetric networks can be seen here for asymmetric networks too. For instance, under low arrival rate, $\lambda = 2.0$, the buffer allocation tends to be independent on the squared coefficient of variation of the service time, s^2, while under heavy traffic, i.e. $\lambda = 4.0$ and 8.0, the buffer allocation tends to increase as s^2 increases. Finally, it is noticeable the symmetry, almost perfectly, between the results from the split and merge topologies, as seen from Tab. 7, which is very encouraging.

5.6 Complex Bottleneck Networks

In order to model larger more complex networks, let us examine a combination of series, merge, and splitting networks which represent interesting combinations of these topologies. In fact, the first combination will be called a primal network as depicted in Fig. 11. In the primal network, nodes #1 and #6 are the bottlenecks since the blocking will be most severe for these nodes. Most of the buffers in the optimization will be allocated to these nodes.

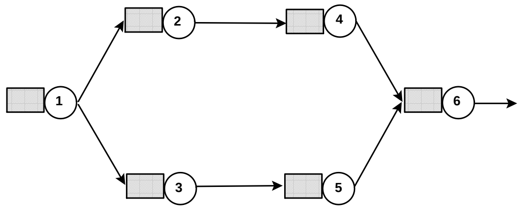

Figure 11: Primal Network Topology

The dual network derived from a dual graph representation of the primal network represents a tree topology (see Fig. 12). In the dual network, on the other hand, nodes #3 and #4 are the bottlenecks so again as in the primal network, the buffer allocation will be directed to these nodes. Since the bottleneck nodes in both the primal and dual network are critically important we will allocate additional servers to them in the experiments.

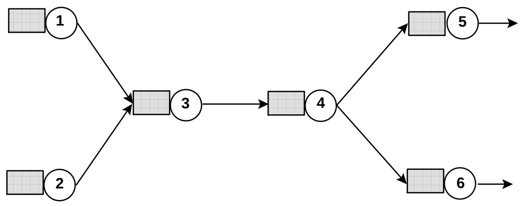

Figure 12: Dual Network Topology

For both of these networks, six simulation experiments were conducted across the s^2 values. In all the experiments, $\lambda = 7$ was chosen as an arrival rate to the

Table 7: Asymmetric Network Comparison Results

Topology	λ	μ	s^2	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$	Simulation $\theta(\mathbf{x})^s$	δ	Z_α^s
series	2.0	(12,11,10)	0.5	(2,4,10)	2.000	17.27	(3,4,10)	1.999	0.002	18.00
	4.0		1.0		3.998	25.71	(7,6,11)	4.000	0.002	24.00
	8.0		1.5		7.987	74.03	(17,22,23)	7.997	0.002	65.00
split	2.0	(12,11,11)	0.5	(2,4,4)	2.000	11.26	(3,4,4)	1.999	0.002	12.00
	4.0		1.0		3.999	14.67	(6,4,4)	4.000	0.002	14.00
	8.0		1.5		7.995	34.50	(18,7,4)	7.999	0.003	30.00
merge	2.0	(11,11,12)	0.5	(4,4,2)	2.000	11.26	(4,4,3)	2.000	0.002	11.00
	4.0		1.0		3.999	14.67	(4,4,6)	4.001	0.002	13.00
	8.0		1.5		7.995	35.49	(8,5,17)	8.001	0.003	29.00

network. The optimal buffer allocation $\mathbf{x}$ for the topology generated by the algorithm is in the middle of each of the two tables, Tab. 8 and Tab. 9. In Tab. 8 the analytical and simulation models are fairly close and the Z value are at most off by 8% (column $\Delta\%Z_\alpha$).

The buffer allocation is symmetric for the primal and dual network topologies which one might expect and that is primarily due to the bottleneck blocking mechanisms that arise within these two different topologies.

The comparisons of the simulation and analytical models for the dual network are surprisingly close which is very encouraging. Included in these tables is also the % deviation for the analytical results on the throughput, $\Delta\%\theta(\mathbf{x})$, and the objective function value, $\Delta\%Z_\alpha$. One notices that the % deviation from the throughput values for the analytical model is very encouraging ~ 0 and the % deviation from the objective function values is well within 10% which is very reasonable.

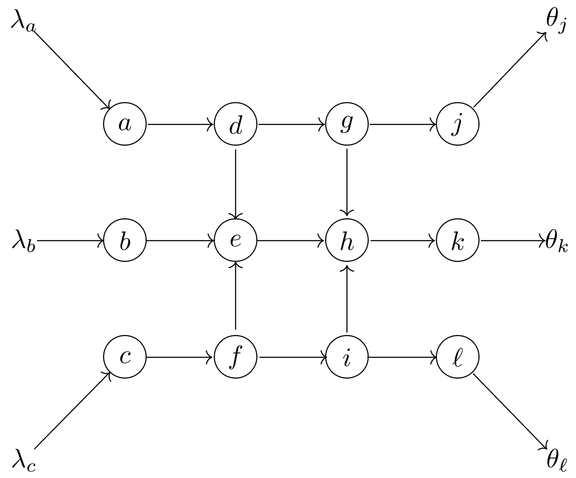

Figure 13: Cubic Sub-graph

5.7 *Larger Bottleneck Networks*

Let us take a subgraph decomposition of the 3d cube of Fig. 2, as seen in Fig. 13. This represents a complex series, merge, and split network of 12 nodes and 13 arcs. Flow is directed along the spine of the cube. Each of the bottleneck nodes $\{e, h, k\}$ will incur significant blocking and it will reverberate throughout the network. Intuition tells us that the bottleneck nodes will receive the most buffers, yet we also wish to factor into the equation the effect of s^2. For the first experiment, all the nodes have single servers.

Tab. 10 illustrates the resulting buffer allocations for the network as a function of the input traffic rate which varies from $\lambda = \{5, .., 9\}$ and $s^2 = \{1, 0, 2\}$ in this order. The nonlinear growth rate of the buffers is consistent with what one would think should happen to the bottlenecks servers. Fig. 14 illustrates the allocation of the buffers to the bottleneck nodes in the topology to nodes e, h, k respectively. While the allocations are not convex (since the buffers are discrete), they show that the monotonic curves are an interesting function of s^2 and the depth of the bottlenecks in the cubic topology. The results are also intuitively appealing in that the buffer under the deterministic service time $s^2 = 0$ is roughly half of the allocation under the $s^2 = 2$ variation.

In the simulation comparisons which follow, the idea is to see how close to optimal one might be given that the simulation is supposedly exact. For each s^2 variation experiment, we have perturbed the bottleneck buffer nodes e, h, k by ± 1 buffer to see how Z behaves. In Tab. 11 the general performance results are pretty encouraging. θ is within the 95% confidence intervals in the $s^2 = 0, 1$ results but not so for $s^2 = 2$. The percentage deviations in θ are very acceptable. The percentage deviations in Z are very acceptable in $s^2 = 0, 1$ but borderline for $s^2 = 2$. We seem to overestimate the buffers needed with our methodology based upon the $Z-$value simulation results.

Now, let us perform another experiment where we allocate more servers to the bottleneck nodes. Say that the number of servers are set to $c_e = 2, c_h = 3, c_k = 4$. Again for each s^2 variation experiment, we have perturbed the bottleneck buffer nodes e, h, k by ± 1 buffer to see how Z behaves. Tab. 12 illustrates the allocation for this new configuration. Again the buffer pattern results are consistent with what is expected from Tab. 10, although the bottlenecks receive fewer buffers because there are now multiple servers.

For the simulations in this multi-server case, Tab. 13,

Table 8: Primal 6-node Network Results

s^2	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$	Simulation			$\Delta\%\theta(\mathbf{x})$	$\Delta\%Z_\alpha$
					$\theta(\mathbf{x})^s$	δ	Z_α^s		
1/2	(1,1,1,1,1,1)	6.993	59.25	(14,6,6,6,6,14)	6.988	0.004	64.50	0.0716	-8.14
1	(2,1,1,1,1,2)	6.989	69.41	(17,6,6,6,6,17)	6.990	0.003	67.80	-0.0143	2.37
3/2	(4,1,1,1,1,4)	6.989	75.24	(18,7,7,7,7,18)	6.992	0.003	72.30	-0.0429	4.07

Table 9: Dual 6-node Network Results

s^2	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$	Simulation			$\Delta\%\theta(\mathbf{x})$	$\Delta\%Z_\alpha$
					$\theta(\mathbf{x})^s$	δ	Z_α^s		
1/2	(1,1,1,1,1,1)	6.993	59.25	(6,6,14,14,6,6)	6.993	0.004	55.40	0.0000	6.95
1	(1,1,2,2,1,1)	6.989	69.41	(6,6,17,17,6,6)	6.989	0.003	68.70	0.0000	1.03
3/2	(1,1,4,4,1,1)	6.989	75.24	(7,7,18,18,7,7)	6.990	0.004	74.00	-0.0143	1.68

Table 10: 12-node Cube Results

arrivals	λ	a	b	c	d	e	f	g	h	i	j	k	ℓ	θ	Z
(2,1,2)	5	4	2	4	4	6	4	2	7	2	2	7	2	4.992	54.435
(2,2,2)	6	4	4	4	4	7	4	2	10	2	2	10	2	5.992	63.456
(3,1,3)	7	6	2	6	6	7	6	3	11	3	2	11	2	6.990	74.927
(3,2,3)	8	6	4	6	6	10	6	3	15	3	2	15	2	7.990	88.008
(3,3,3)	9	6	6	6	6	13	6	3	21	3	2	21	2	8.987	107.635
(2,1,2)	5	3	2	3	3	4	2	2	5	2	1	5	1	4.992	42.480
(2,2,2)	6	3	3	3	3	5	4	2	7	2	1	7	1	5.992	48.105
(3,1,3)	7	4	2	4	4	5	4	3	8	3	2	8	2	6.993	55.807
(3,2,3)	8	4	3	4	4	8	4	3	10	3	2	10	2	7.993	63.374
(3,3,3)	9	4	4	4	4	9	4	3	13	3	2	13	2	8.991	74.315
(2,1,2)	5	5	3	5	5	7	5	3	9	3	2	9	2	4.993	65.514
(2,2,2)	6	5	5	5	5	9	5	3	13	3	2	13	2	5.992	77.904
(3,1,3)	7	7	3	7	7	9	7	4	15	4	2	15	2	6.989	93.044
(3,2,3)	8	7	5	7	7	13	7	4	20	4	2	20	2	7.987	111.444
(3,3,3)	9	7	7	7	7	17	7	4	28	4	2	28	2	8.981	138.673

Table 11: 12-node Cube Single-server Network Results

s^2	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$	Simulation			$\Delta\%\theta(\mathbf{x})$	$\Delta\%Z_\alpha$
					$\theta(\mathbf{x})^s$	δ	Z_α^s		
1	(1,. . . , 1)			(4,2,4,4,**5**,4,2,**6**,2,2,**6**,2)	4.9882	0.00320	54.80		
1	(1,. . . , 1)	4.992	54.435	(4,2,4,4,**6**,4,2,**7**,2,2,**7**,2)	4.9916	0.00350	55.30	0.0080	1.5
1	(1,. . . , 1)			(4,2,4,4,**7**,4,2,**8**,2,2,**8**,2)	4.9907	0.00302	57.40		
0	(1,. . . , 1)			(4,2,4,4,**4**,4,3,**7**,3,2,**7**,2)	6.9956	0.00301	50.40		
0	(1,. . . , 1)	6.993	55.81	(4,2,4,4,**5**,4,3,**8**,3,2,**8**,2)	6.9959	0.00333	53.10	-0.0400	5.1
0	(1,. . . , 1)			(4,2,4,4,**6**,4,3,**9**,3,2,**9**,2)	6.9950	0.00332	57.00		
2	(1,. . . , 1)			(7,7,7,7,**16**,7,4,**27**,4,2,**27**,2)	8.9956	0.00363	121.40		
2	(1,. . . , 1)	8.981	138.67	(7,7,7,7,**17**,7,4,**28**,4,2,**28**,2)	8.9945	0.00326	125.50	-0.1500	10.49
2	(1,. . . , 1)			(7,7,7,7,**18**,7,4,**29**,4,2,**29**,2)	8.9941	0.00314	128.90		

Table 12: 12-node Cube Multi-server Results

arrivals	λ	a	b	c	d	e	f	g	h	i	j	k	ℓ	θ	Z
(2,1,2)	5	4	2	4	4	5	4	2	6	2	2	6	2	4.992	50.821
(2,2,2)	6	4	4	4	4	7	4	2	9	2	2	8	2	5.992	60.196
(3,1,3)	7	6	2	6	6	7	6	3	10	3	2	10	2	6.991	71.731
(3,2,3)	8	6	4	6	6	9	6	3	14	3	2	14	2	7.990	85.034
(3,3,3)	9	6	6	6	6	12	6	3	20	3	2	19	2	8.986	104.810
(2,1,2)	5	3	2	3	3	4	2	2	5	2	1	5	1	4.992	42.111
(2,2,2)	6	3	3	3	3	6	4	2	7	2	1	7	1	5.992	49.287
(3,1,3)	7	4	2	4	4	6	4	3	8	3	2	8	2	6.992	57.555
(3,2,3)	8	4	3	4	4	8	4	3	11	3	2	11	2	7.991	67.549
(3,3,3)	9	4	4	4	4	10	4	3	15	3	2	15	2	8.988	81.802
(2,1,2)	5	5	3	5	5	6	5	3	7	3	2	7	2	4.994	58.824
(2,2,2)	6	5	5	5	5	8	5	3	10	3	2	10	2	5.993	70.452
(3,1,3)	7	7	3	7	7	8	7	4	12	4	2	11	2	6.989	85.155
(3,2,3)	8	7	5	7	7	11	7	4	16	4	2	16	2	7.986	101.965
(3,3,3)	9	7	7	7	7	15	7	4	24	4	2	24	2	8.984	126.490

Table 13: 12-node Cube Multi-server Network Results

s^2	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$	Simulation $\theta(\mathbf{x})^s$	Simulation δ	Simulation Z_α^s	$\Delta\%\theta(\mathbf{x})$	$\Delta\%Z_\alpha$
1	(1,1,1,1,2,1,1,3,1,1,4,1)			(4,2,4,4,**4**,4,2,**5**,2,2,**5**,2)	4.9913	0.00254	48.7		
1	(1,1,1,1,2,1,1,3,1,1,4,1)	4.992	50.82	(4,2,4,4,**5**,4,2,**6**,2,2,**6**,2)	4.9911	0.00265	51.9	0.018	2.08
1	(1,1,1,1,2,1,1,3,1,1,4,1)			(4,2,4,4,**6**,4,2,**7**,2,2,**7**,2)	4.9906	0.00295	55.4		

s^2	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$	Simulation $\theta(\mathbf{x})^s$	Simulation δ	Simulation Z_α^s	$\Delta\%\theta(\mathbf{x})$	$\Delta\%Z_\alpha$
0	(1,1,1,1,2,1,1,3,1,1,4,1)			(4,2,4,4,**5**,4,3,**7**,3,2,**7**,2)	6.9948	0.00399	52.2		
0	(1,1,1,1,2,1,1,3,1,1,4,1)	6.992	57.55	(4,2,4,4,**6**,4,3,**8**,3,2,**8**,2)	6.9948	0.00399	55.2	-0.040	4.25
0	(1,1,1,1,2,1,1,3,1,1,4,1)			(4,2,4,4,**7**,4,3,**9**,3,2,**9**,2)	6.9948	0.00399	58.2		

s^2	$\mathbf{c}$	$\theta(\mathbf{x})$	Z_α	$\mathbf{x}$	Simulation $\theta(\mathbf{x})^s$	Simulation δ	Simulation Z_α^s	$\Delta\%\theta(\mathbf{x})$	$\Delta\%Z_\alpha$
2	(1,1,1,1,2,1,1,3,1,1,4,1)			(7,7,7,7,**14**,7,4,**23**,4,2,**23**,2)	8.9907	0.00333	116.3		
2	(1,1,1,1,2,1,1,3,1,1,4,1)	8.984	126.49	(7,7,7,7,**15**,7,4,**24**,4,2,**24**,2)	8.9910	0.00389	119.0	-0.077	6.29
2	(1,1,1,1,2,1,1,3,1,1,4,1)			(7,7,7,7,**16**,7,4,**25**,4,2,**25**,2)	8.9919	0.00335	121.1		

the results for the confidence intervals are similar to the single-server case while the percentage deviations on θ and Z are even more acceptable. Again, we seem to overestimate the number of buffers needed with our methodology based upon the simulation results and the $Z-$values which resulted.

It is interesting that in the $s^2 = 0$ experiments in Tab. 12, adding the servers for the experiments in $\lambda = 6, 7, 8, 9$ is worse than adding the buffers in Tab. 10. This is somewhat counterintuitive, however, just reducing the variability and having the $s^2 = 0$ and adequate buffers is a cheaper solution than adding the servers for the objective function in question, whereas for the higher variability concerns $s^2 = \{1, 2\}$, adding the servers (assuming an equivalent cost to the buffers) is beneficial especially at the higher traffic levels. Therefore, one must be careful in arbitrarily adding servers to a network topology, since there may be other ways of improving performance such as reducing s^2 or adding buffers or both.

At this point we could continue to run more experiments with larger and more complex networks. We choose to stop here, since that process would be endless, and perhaps not reveal more than we have already shown.

6 Summary and Conclusions

We have provided a comprehensive approach to the buffer allocation problem of finite open queueing networks with general service and multiple-servers. Both the derivation of the blocking probability formulas used in the experiments as well as the optimization methodology have been described. Numerous experiments illustrating the scope and limitations of the approach have been shown.

Open Questions

Among the possible directions this research could evolve would be with the various applications of the algorithm such as in manufacturing and assembly problems, facility planning and layout design, telecommunication, and computer system network design problems. Since some of these applications require BBS and RPB blocking protocols, we would have to revise the Expansion method to do so. This is a valid research issue.

We have not examined in any detail the situation where we make the number of servers **c** a decision variable. This would require a possible re-structuring of the optimization approach and we decided not to carry out this activity at this stage of the research. One discouraging thing about the number of servers is that they do not seem as critical as the number of buffers which was evident in our results. Other researchers have found similar results about the importance of the number of buffers.

Another facet worth considering is that the material handling transfer time or people movement problems of these networks would require us to incorporate $M/G/\infty$ and $M/G/c/c$ travel times. This modification would add additional nodes to the network and would make the optimization methodology even more comprehensive. We are carefully considering this option.

Another possible extension is to include networks with feedback loops as this is often found in manufacturing as well as other service sector problems. Feedback loops cause strong dependencies within the networks, so this needs careful consideration.

Improvements to the Expansion method can also occur so that it can handle higher traffic levels with even more variability and uncertainty, perhaps even with general arrival processes.

We hope that the reader senses the power of this approach and the ability we now have to tackle these complex network planning and design problems. One hopes that we can make further improvements to this

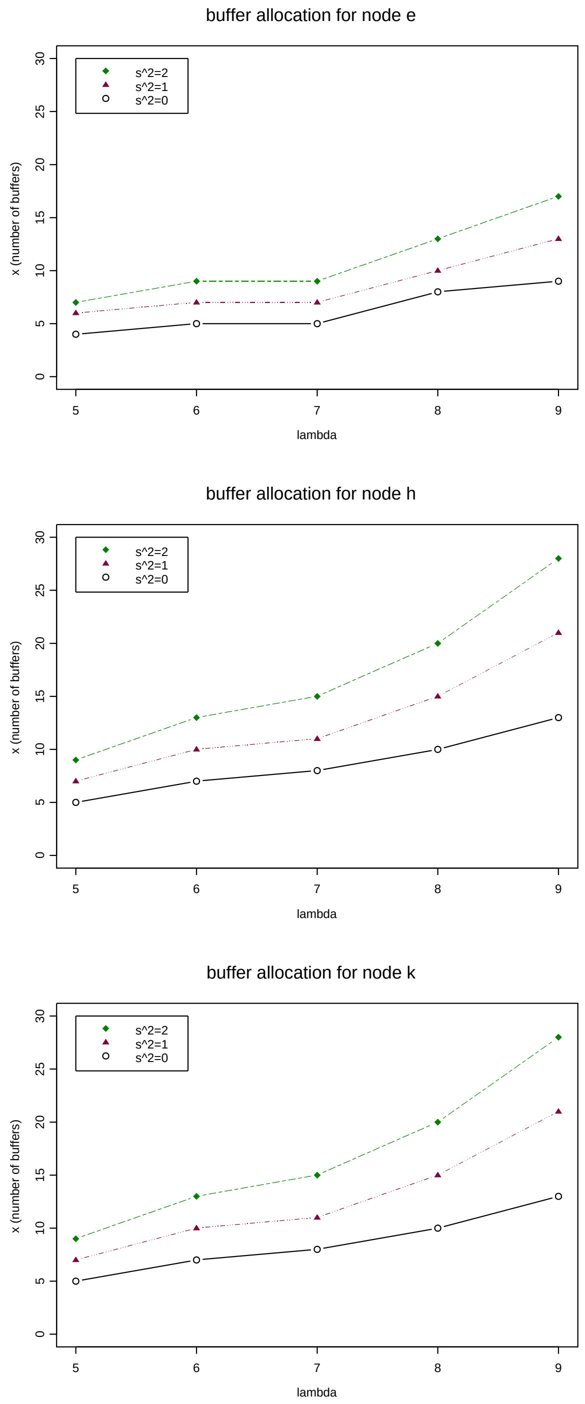

Figure 14: Bottleneck BAPs in cubic architecture

network design process in future research.

Acknowledgments

The research of Frederico Cruz has been partially funded by CNPq (*Conselho Nacional de Desenvolvimento Científico e Tecnológico*) of the Ministry for Science and Technology of Brazil, grants 201046/1994-6, 301809/1996-8, 307702/2004-9, 472066/2004-8, 472877/2006-2, and 304944/2007-6, by CAPES (*Coordenação de Aperfeiçoamento de Pessoal de Nível Superior*), grant BEX-0522/07-4, by FAPEMIG (*Fundação de Amparo à Pesquisa do Estado de Minas Gerais*), grants CEX-289/98, CEX-855/98, TEC-875/07, and CEX-PPM-00401/08.

References

Altiok, T., Stidham, S., 1983. The allocation of interstage buffer capacities in production lines. IIE Transactions 15, 251–261.

Baker, K. R., Powell, S., Pyke, D., 1990. Buffered and unbuffered assembly systems with variable processing times. Journal of Manufacturing and Operations Management 3, 200–223.

Carrasco, J. A., 2006. Two methods for computing bounds for the distribution of cumulative reward for large Markov models. Performance Evaluation 63 (12), 1165–1195.

Choi, D. W., Kim, N. K., Chae, K. C., 2005. A two-moment approximation for the $GI/G/c$ queue with finite capacity. INFORMS Journal on Computing 17 (1), 75–81.

Conway, R., Maxwell, W. L., McClain, J. O., Thomas, L. J., 1988. The role of work-in-process inventories in serial production lines. Operations Research 36, 229–241.

Cruz, F. R. B., Duarte, A. R., van Woensel, T., 2008a. Buffer allocation in general single-server queueing network. Computers & Operations Research 35 (11), 3581–3598.

Cruz, F. R. B., van Woensel, T., Smith, J. M., Lieckens, K., 2008b. On the system optimum of traffic assignment in $M/G/c/c$ state-dependent queueing networks. European Journal of Operational Research (under review). URL `ftp://ftp.est.ufmg.br/pub/fcruz/publics/assign.pdf`

Dallery, Y., Gershwin, S. B., 1992. Manufacturing flow line systems: A review of models and analytical results. Queueing Systems 12 (1-2), 3–94.

Eklin, M., Arzi, Y., Shtub, A., 2009. Model for cost estimation in a finite-capacity stochastic environment based on shop floor optimization combined with simulation. European Journal of Operational Research 194 (1), 294–306.

de Nitto Persone, V., 1994. Topology related index for performance comparison of blocking symmetrical networks. European Journal of Operational Research 78 (3), 413–425.

M. Eklin, Y. Arzi, and A. Shtub, 2009. Model for cost estimation in a finite-capacity stochastic environment based on shop floor optimization combined with simulation. European Journal of Operational Research 194(1): 294-306

Gaver, D., Shedler, G. S., 1973a. Approximate models for processor utilization in multiprogrammed computer systems. SIAM Journal of Computing 2/3, 183–192.

Gaver, D., Shedler, G. S., 1973b. Processor utilization in multi-programming systems via diffusion approximations. Operations Research 21, 569–576.

Gelenbe, E., 1975. On approximate computer system models. Journal of the ACM 22 (2), 261–269.

Harris, J. H., Powell, S. G., 1999. An algorithm for optimal buffer placement in reliable serial lines. IIE Transactions 31, 287–302.

Hildebrand, D., 1968. On the capacity of tandem server, finite queue, service systems. Operations Research 16, 72–82.

Hillier, F. S., Boling, R. W., 1967. Finite queues in series with exponential or Erlang service times: A numerical approach. Operations Research 15, 286–303.

Hillier, F. S., So, K. C., 1991. The effect of the coefficient of variation of operation times on the allocation of storage space in production line systems. IIE Transactions 23 (2), 198–206.

Himmelblau, D. M., 1972. Applied Nonlinear Programming. McGraw-Hill Book Company, New York.

Ho, Y. C., Eyler, M. A., Chien, T., 1979. A gradient technique for general buffer storage design in a production line. International Journal of Production Research 17 (6), 557–580.

Jafari, M. A., Shanthikumar, J. G., 1989. Determination of optimal buffer storage capacities and optimal allocation in multistage automatic transfer lines. IIE Transactions 21 (2), 130–135.

Kerbache, L., Smith, J. M., 1987. The generalized expansion method for open finite queueing networks. European Journal of Operational Research 32, 448–461.

Kerbache, L., Smith, J. M., 1988. Asymptotic behavior of the expansion method for open finite queueing networks. Computers & Operations Research 15 (2), 157–169.

Kim, N. K., Chae, K. C., 2003. Transform-free analysis of the $GI/G/1/K$ queue through the decomposed Little's formula. Computers & Operations Research 30 (3), 353–365.

Kimura, T., 1985. Refining diffusion approximations for GI/G/1 queues: A tight discretization method. Teletraffic Congress , 317–323.

Kimura, T., 1996a. Optimal buffer design of an $M/G/s$ queue with finite capacity. Communications in Statistics - Stochastic Models 12 (1), 165–180.

Kimura, T., 1996b. A transform-free approximation for the finite capacity $M/G/s$ queue. Operations Research 44 (6), 984–988.

Kleinrock, L., 1975. Queueing Systems. Vol. I: Theory. John Wiley & Sons, New York, NY, USA.

Kubat, P., Sumita, U., 1985. Buffers and backup machines in automatic transfer lines. International Journal of Production Research 23 (6), 1259–1280.

Labetoulle, J., Pujolle, G., 1980. Isolation method in a network of queues. IEEE Transactions on Software Engineering SE-6 (4), 373–381.

Leighton, F. T., 1991. Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Lemaréchal, C., 2007. The omnipresence of Lagrange. Annals of Operations Research 153 (1), 9–27.

Menasce, D. A., 2002. QoS issues in web services. IEEE Internet Computing 6 (6), 72–75.

Muth, E., Alkaff, A., 1987. The throughput rate of three-stationproduction lines: A unifying solution. International Journal of Production Research 25, 1405–1413.

Neuts, M. F., 1995. Matrix Geometric Solution in Stochastic Models: An Algorithmic Approach. Dover Publications, New York, NY.

Onvural, R. O., 1990. Survey of closed queueing networks with blocking. ACM Computing Surveys 22 (2), 83–121.

Pla, V., Casares-Giner, V., 2005. Analysis of priority channel assignment schemes in mobile cellular communication systems: A spectral theory approach. Performance Evaluation 59 (2-3), 199–224.

Powell, S. G., 1992. Buffer allocation in unbalanced serial lines. Working Paper #289, The Amos Tuck School of Business Administration, Dartmouth College, Hanover, N.H. 03755.

Rao, N., 1976a. A generalization of the bowl phenomenon in series production systems. International Journal of Production Research 14 (4), 437–443.

Rao, N., 1976b. A viable alternative to the method of stages solution of series production systems with Erlang service times. International Journal of Production Research 14 (6), 699–702.

Robinson, S., 2007. A statistical process control approach to selecting a warm-up period for a discrete-event simulation. European Journal of Operational Research 176 (1), 332–346.

Sakasegawa, H., Miyazawa, M., Yamazaki, G., 1993. Evaluating the overflow probability using the infinite queue. Management Science 39 (10), 1238–1245.

Schweitzer, P. J., Konheim, A. G., 1978. Buffer overflow calculations using an infinite-capacity model. Stochastic Processes and their Applications 6 (3), 267–276.

Seong, D., Chang, S. Y., Hong, Y., 1995. Heuristic algorithms for buffer allocation in a production line with unreliable machines. International Journal of Production Research 33 (7), 1989–2005.

Smith, J. M., 2003. $M/G/c/K$ blocking probability models and system performance. Performance Evaluation 52 (4), 237–267.

Smith, J. M., 2004. Optimal design and performance modelling of $M/G/1/K$ queueing systems. Mathematical and Computer Modelling 39 (9-10), 1049–1081.

Smith, J. M., Chikhale, N., 1995. Buffer allocation for a class of nonlinear stochastic knapsack problems. Annals of Operations Research 58, 323–360.

Smith, J. M., Cruz, F. R. B. 2005. The buffer allocation problem for general finite buffer queueing networks. IIE Transactions 37 (4), 343–365.

Smith, J. M., Daskalaki, S., 1988. Buffer space allocation in automated assembly lines. Operations Research 36 (2), 343–358.

Soyster, A. L., Schmidt, J. W., Rohrer, M. W., 1979. Allocation of buffer capacities for a class of fixed cycle production lines. IIE Transactions 11 (2), 140–146.

Spieckermann, S., Gutenschwager, K., Heinzel, H., Voß, S., 2000. Simulation-based optimization in the automotive industry a case study on body shop design. Simulation 75 (5), 276–286.

Spinellis, D., Papadopoulos, C. T., Smith, J. M., 2000. Large production line optimization using simulated annealing. International Journal of Production Research 38 (3), 509–541.

Tijms, H. C., 1987. Stochastic Modelling and Analysis: A Computational Approach. John Wiley & Sons, New York.

Tijms, H. C., 1992. Heuristics for finite-buffer queues. Probability in the Engineering and Informational Sciences 6, 267–276.

Tijms, H. C., 1994. Stochastic Models: An Algorithmic Approach. John Wiley & Sons, New York.

Yamashita, H., Altiok, T., 1998. Buffer capacity allocation for a desired throughput in production lines. IIE Transactions 30 (10), 883–892.

Yamashita, H., Onvural, R., 1994. Allocation of buffer capacities in queueing networks with arbitrary topologies. Annals of Operations Research 48, 313–332.

Yamashita, H., Suzuki, S., 1987. An approximate solution method for optimal buffer allocation in serial n-stage automatic production lines. Transactions of the Japanese Society of Mechanical Engineers 53-C, 807–817.

Yao, D. D. W., Buzacott, J. A., 1985. Queueing models for a flexible machining station Part I: The diffusion approximation. European Journal of Operational Research 19 (2), 233–240.