Performance Bounds and Differential Cost Approximations for Queueing Networks

Stephen Rizzo
Michael Veatch

August 1, 2008

Abstract

A new bound is proposed relating the average cost sub-optimality of a myopic policy to its differential cost error. The bound is an expectation over the stationary distribution for this policy. For small queueing networks, we compute or approximate the distribution using the matrix analytic method and numerically test the bound. Product form approximations to the stationary distribution are also considered. Applications to approximate dynamic programming are also discussed.

1 Introduction

Queueing network optimization problems are important models in various areas, including manufacturing systems and engineering control theory. Unfortunately, the optimal solution to the Bellman equation is often intractable; without knowing the optimal differential cost, it is impossible to calculate the associated optimal average cost. Iterative dynamic programming methods, which are capable of finding the optimal average cost for smaller problems, are often computationally very difficult or impossible to compute for larger queueing networks. A common solution to this problem is to approximate the differential cost of the system. By making assumptions about the form of h, the size of the problem can be reduced considerably. Often we consider h to be a linear function of a set of basis functions,

$$h = \sum_{i=0}^{k} r_i \phi_i$$

and optimize over the smaller set of variables r_i. However, this approximation introduces error into the system, which appears as a difference in the policy determined by the differential cost function. This sub-optimality of the myopic policy with respect to h results in an increase in the associated average cost J_h. We present an upper bound for this difference between J_h and J^* for a given differential cost function h.

Other approaches to bounding performance of queueing networks include the achievable region method [1], [8], [7], and approximate linear programming [5], [4], [6], [11], [12], [15], and [2]. The first two methods use linear programming. Most of these bounds are for total average cost. In contrast, [4] and our method bound average cost sub-optimality. Bounds on sub-optimality for discounted cost are given in [5] . Also, these methods give a bound on the total cost for any policy in some class of policies, while our bound is for a specific myopic policy.

The bound we use starts with a differential cost approximation and the myopic policy associated with this cost function. It relates the differential cost error to the sub-optimality of this policy. The primary motivation for this bound is approximate dynamic programming, where the differential cost is approximated by the linear form described above.

2 A performance bound in terms of the differential cost error

2.1 Derivation of the bound

Consider a differential cost approximation h for which the myopic policy $u_h(x) = \arg\min(P_u h)(x)$ is stabilizing. Let J_h, P_h and π_h denote the average cost, transition probability matrix, and stationary distribution under this policy. Using the ACOE,

$$\begin{aligned} J_h &= \pi_h g = \pi_h(J^* \mathbf{1} + h^* - P_{h^*} h^*) \\ J_h - J^* &= \pi_h(h^* - P_{h^*} h^*) \end{aligned} \tag{1}$$

Note that $P_h h = \min_u P_u h$. If h satisfies a suitable growth condition under π_h, then $\pi_h(P_h h - h) = 0$. Using these facts,

$$J_h - J^* = \pi_h(h^* - \min_u P_u h^*) - \underbrace{\pi_h(h - \min_u P_u h)}_{=0} \tag{2}$$

To obtain a bound in terms of $h^* - h$, we replace the two minimums in (2) with the optimal policy. Since $P_h h \leq P_{h^*} h$, we have

$$\begin{aligned} 0 \leq J_h - J^* &\leq \pi_h(h^* - P_{h^*} h^* - h + P_{h^*} h) \\ &= [\pi_h(I - P_{h^*})]\,(h^* - h) \\ &= [\pi_h(P_h - P_{h^*})]\,(h^* - h) \end{aligned} \tag{3}$$

On infinite state spaces, the first equality involves a limit interchange and so requires the assumptions that $\pi_h(P_{h^*} h - h) = [\pi_h(P_{h^*} - I)]\, h$ and similarly for h^*. Typically these equations hold under the growth condition on h used above.

This bound relates the performance of a policy to the differential cost error. A similar bound is proposed in [5] for cases where $h < h^*$. Under this assumption,

$$J_h - J^* \leq \pi_h(h^* - h) \tag{4}$$

We are interested in using h found by the ALP, where there is no guarantee that $h^* < h$. The bracketed expression in (3) involves π_h, which we will compute or bound in small examples, and P_{h^*}. Although we don't know the optimal policy, typically we have some information about P_{h^*}, so (3) is preferable. In fact, $P_h - P_{h^*}$ is non-zero in relatively few states.

2.2 Example: M/M/1 queue with controlled arrivals

Now let us consider an M/M/1 queue with a holding cost cx, service rate μ and arrival rate λ. We choose in any state either to accept ($u = 1$) or reject ($u = 0$) arrivals. Let r be the reward for the action $u = 1$. The optimal solution to this arrival control problem will necessarily give a hedging point z^* such that $u = 1$ for $x < z^*$ and $u = 0$ thereafter. In this section we adopt the notation $\Pi_z = [\pi_x]$ for a stationary distribution with a hedging point z. Under such a policy the system is a M/M/1/z queue, with a stationary probability distribution

$$\pi_x = \begin{cases} \frac{\rho^x}{\sum_{j=0}^{z} \rho^j} & \text{if } 0 \leq x \leq z; \\ 0 & \text{if } x > z. \end{cases}$$

Note that $\pi_x = \rho\pi_{x-1}$, with $1 \leq x \leq z$. The generator matrix for the optimal policy is

$$I - P_{z^*} = \begin{bmatrix} 1-\mu & -\lambda & 0 & 0 & \dots & 0 & \overbrace{0}^{z^*} & 0 & \dots \\ -\mu & 1 & -\lambda & 0 & \dots & 0 & 0 & 0 & \dots \\ 0 & -\mu & 1 & -\lambda & \dots & 0 & 0 & 0 & \dots \\ \vdots & \vdots & \ddots & \ddots & \ddots & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & -\mu & 1 & -\lambda & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & -\mu & 1 & -\lambda & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & -\mu & 1-\lambda & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & 0 & -\mu & 1-\lambda & \ddots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \ddots \end{bmatrix}$$

We will evaluate the coefficients of $h^* - h$ in (3). If $z = z^*$, the coefficients are 0. Next consider $0 < z < z^*$. For $x = 0$

$$[\Pi_z(I - P_{z^*})]_0 = (1-\mu)\pi_0 - \mu\pi_1 = \pi_0(\underbrace{\lambda - \mu\rho}_{=0}) = 0$$

For states $1 \leq x \leq z - 1$:

$$[\Pi_z(I - P_{z^*})]_x = -\lambda\pi_{x-1} + \pi_x - \mu\pi_{x+1} = \pi_{x-1}(-\lambda + \rho - \mu\rho^2) = \pi_x(\underbrace{1 - \mu - \lambda}_{=0}) = 0$$

For $x = z$,

$$[\Pi_z(I - P_{z^*})]_z = -\lambda\pi_{z-1} + \pi_z = \pi_{z-1}(-\lambda + \rho) = \lambda\pi_z$$

Since $\pi_x = 0$, $x > z$, for $x = z + 1$

$$[\Pi_z(I - P_{z^*})]_{z+1} = -\lambda\pi_z$$

and the coefficient is 0 for $x > z + 1$.

Now consider $0 < z^* < z$. The same analysis gives a coefficient of 0 in states $x < z^*$. For $x = z^*$,

$$[\Pi_z(I - P_{z^*})]_{z^*} = -\lambda\pi_{z^*-1} + (1-\lambda)\pi_{z^*} - \mu\pi_{z^*+1} = \pi_{z^*-1}(-\lambda + \frac{\lambda}{\mu} - 2\frac{\lambda^2}{\mu}) = -\lambda\pi_{z^*}$$

For $z^* + 1 \leq x < z$

$$[\Pi_z(I - P_{z^*})]_x = (1-\lambda)\pi_x - \mu\pi_{x+1} = \pi_x(\underbrace{1-\lambda}_{=\mu} - \underbrace{\mu\rho}_{=\lambda}) = (\mu - \lambda)\pi_x$$

For $x = z$,

$$[\Pi_z(I - P_{z^*})]_z = (1-\lambda)\pi_z = \mu\pi_z$$

The cases where $z = 0$ or $z^* = 0$ are similar, but usually not of interest. One could use the aforementioned coefficients to compute $J_h - J^*$ using (1). Instead, we note that there is no sensitivity in states below both hedging points. When too small a hedging point is used, there is a sensitivity to h^* at the hedging point. When too large a hedging point is used, there is a sensitivity to h^* between the two hedging points. We will explore more about these sensitivities in the next section.

2.3 Performance bound sensitivities in series 2

A closer examination of the $(h^* - h)$ bound (3) allows us to make certain comments about the sensitivities of the bound to the state space of our problem. Let us define the switching curves s_h and s^* associated with h and h^* respectively such that $u_1(x) = 1$ if $x_2 \leq s(x_1)$. Assume that s_h and s^* are non-decreasing. There are 4 distinct cases:

1. Suppose that for some x_1, $s^*(x_1) \geq s_h(x_1 + 1) + 1 > s_h(x_1)$. In all states (x_1, x_2) with $x_2 \leq s_h(x_1)$, the transitions are the same under both policies, and therefore $P_h - P_{h^*}$ cancels out all of those sensitivities. For states such that $x_2 > s_h(x_1 + 1)$, those states are transient under u_h, therefore $\pi_h(x_1, x_2) = 0$, canceling out their sensitivities. Therefore non-zero sensitivities can only exist in the states (x_1, x_2) where $s_h(x_1) < x_2 \leq s_h(x_1 + 1) + 1$.

2. Suppose that for some x_1, $s_h(x_1 + 1) + 1 > s^*(x_1) > s_h(x_1)$. In all states (x_1, x_2) with $x_2 \leq s_h(x_1)$ or $x_2 > s^*(x_1)$, the transitions are the same under both policies, and therefore $P_h - P_{h^*}$ cancels out all of those sensitivities. Consequently non-zero sensitivities only exist in states (x_1, x_2) such that $s_h(x_1) < x_2 \leq s^*(x_1)$.

3. Suppose that for some x_1, $s_h(x_1 + 1) + 1 \geq s_h(x_1) = s^*(x_1)$. Because the transitions are the same under both policies for all states (x_1, x_2), the bound has no sensitivities to any of these states.

4. Suppose that for some x_1, $s_h(x_1 + 1) + 1 \geq s_h(x_1) > s^*(x_1)$. Sensitivities cancel out for $x_2 > s_h(x_1)$ and for $x_2 \leq s^*(x_1)$ because of identical transitions. Therefore the only non-zero sensitivities are for the states with $s_h(x_1) \geq x_2 > s^*(x_1)$.

Let us note that in the first case the states do not depend on s^*. This has the important consequence that a bound can be calculated without knowing exactly where the optimal switching curve is, as long as it is bounded below by $s_h(x_1 + 1) + 1$.

3 Matrix analytic method

Computing the performance bound (3) generally requires knowing both the transition probabilities under the optimal policy as well as the stationary distribution under the myopic policy for h. While in some smaller examples we can either guess or calculate the exact optimal policy, larger examples force us to bound P_{h^*}. The computation of π_h from the balance equation requires truncating the state space and solving a linear system that grows quickly with the problem size. However, under certain policies the system can be modeled as a quasi-birth-death process and the matrix analytic method [9] can be used to more efficiently compute π_h. We will start be discussing this method's applicability, then introduce it using a small example.

3.1 Applicable policies

The matrix analytic method requires that the transition probability matrix have a particular repetitive block structure. This requires that our policy keep finite buffer sizes for all buffers other than the first, restricting the use of this method to quasi-birth-death processes. This condition restricts us to reentrant lines, as uncontrolled arrivals at multiple classes would result in the inability to control the maximum buffer size of two different classes. A convenient method for generating such policies is to use an ALP to generate h functions. Results from [14] allow us to select parameters for the ALP. However, we must first ensure that the ALP admits feasible solutions of the form needed for the matrix analytic method.

For a reasonable h, the myopic policy is defined by switching surfaces associated with the control variables of the problem. In a series line, we define these switching surfaces $s_i(x)$ such that $u_i = 1$ if $x < s_i(x)$. These surfaces partition the state space into regions, each corresponding to a different action. In this section we will specify conditions on Q such that the switching surfaces are parallel to certain coordinate axes, imposing a maximum buffer size so that the matrix analytic method can be applied.

Let us first consider the example of series 2. The equation for the machine 1 switching curve is given in [3]:

$$s_1(x): \quad x_2 = \frac{q_{12} - q_{11}}{q_{12} - q_{22}}\, x_1 + \frac{\frac{1}{2}q_{11} + \frac{1}{2}q_{22} + p_2 - p_1 - q_{12}}{q_{12} - q_{22}}$$

To have a finite x_2 buffer size the slope of the above switching curve must be zero. This requires $q_{12} - q_{11} = 0 \Rightarrow q_{12} = q_{11}$ or the approximation $|q_{12} - q_{11}| \ll |q_{12} - q_{22}|$.

Now let us consider series 3. Similarly to series 2, the switching surfaces must be invariant with respect to x_1, thus $s_i(x) = s_i(x_2, x_3)$ for $i = 1, 2, 3$. The equations for the switching surfaces for each machine are:

$$s_1(x): \quad (q_{12} - q_{11})\, x_1 + (\frac{1}{2}q_{22} - q_{12})\, x_2 + (q_{23} - q_{13})\, x_3 = \frac{1}{2}q_{11} - q_{12} + q_{22} - p_1 + p_2$$

$$s_2(x): \quad (q_{13} - q_{12})\, x_1 + (q_{23} - q_{22})\, x_2 + (q_{33} - q_{23})\, x_3 = \frac{1}{2}q_{22} - q_{23} + \frac{1}{2}q_{33} - p_2 + p_3$$

Invariance with respect to x_1 requires that $q_{11} = q_{12} = q_{13}$, which is analogous to the series 2 condition.

3.2 Series queue with a buffer size of one

Suppose we have a series queue with arrival rate λ and service rates μ_1, μ_2 for the servers 1 and 2 respectively. Consider the policy $u_1(x) = 1$ if $x_2 = 0$ and 0 otherwise, so that the queue for the second server has a maximum queue length of 1. Let $\Lambda = \lambda + \mu_1 + \mu_2$ be the total transition rate. To simplify notation we will set $\Lambda = 1$. We order the states lexicographically: $(0,0), (0,1), (1,0), (1,1), (2,0), \ldots$

The generator matrix for this queuing network is

$$I - P = \begin{bmatrix} \lambda & 0 & -\lambda & 0 & 0 & \ldots \\ -\mu_2 & \lambda+\mu_2 & 0 & -\lambda & 0 & \ldots \\ 0 & -\mu_1 & \lambda+\mu_1 & 0 & -\lambda & \ldots \\ 0 & 0 & -\mu_2 & \lambda+\mu_2 & 0 & \ldots \\ 0 & 0 & 0 & -\mu_1 & \lambda+\mu_1 & \ldots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Here we notice that the matrix has the following block structure

$$I - P = \left[\begin{array}{c|cc|cc|c} \lambda & 0 & -\lambda & 0 & 0 & \ldots \\ \hline -\mu_2 & \lambda+\mu_2 & 0 & -\lambda & 0 & \ldots \\ 0 & -\mu_1 & \lambda+\mu_1 & 0 & -\lambda & \ldots \\ \hline 0 & 0 & -\mu_2 & \lambda+\mu_2 & 0 & \ldots \\ 0 & 0 & 0 & -\mu_1 & \lambda+\mu_1 & \ldots \\ \hline \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{array}\right]$$

Thus if we define the matrices

$$B_{00} = \begin{bmatrix} \lambda \end{bmatrix} \quad B_{01} = \begin{bmatrix} 0 & -\lambda \end{bmatrix} \quad B_{10} = \begin{bmatrix} -\mu_2 \\ 0 \end{bmatrix}$$

$$A_0 = \begin{bmatrix} -\lambda & 0 \\ 0 & -\lambda \end{bmatrix} \quad A_1 = \begin{bmatrix} \lambda+\mu_2 & 0 \\ -\mu_1 & \lambda+\mu_1 \end{bmatrix} \quad A_2 = \begin{bmatrix} 0 & -\mu_2 \\ 0 & 0 \end{bmatrix}$$

the generator matrix has the form

$$I - P = \left[\begin{array}{cc|cccc} B_{00} & B_{01} & 0 & 0 & 0 & \ldots \\ B_{10} & A_1 & A_0 & 0 & 0 & \ldots \\ \hline 0 & A_2 & A_1 & A_0 & 0 & \ldots \\ 0 & 0 & A_2 & A_1 & A_0 & \ldots \\ 0 & 0 & 0 & A_2 & A_1 & \ddots \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots \end{array}\right] \tag{5}$$

with the right portion of the matrix being repetitive. This repetitive structure implies that the stationary distribution will have a matrix geometric form. Let $\Pi = [\Pi_0, \Pi_1, \Pi_2, \ldots]$ be the stationary distribution, where the blocks Π_i correspond to the blocks in (5), i.e. the number of elements in Π_0 is the number of rows in B_{00} and the number of elements in $\Pi_j, j = 1, 2, 3, \ldots$ is the number of rows in A_0. Then, for the repetitive part of the matrix,

$$\Pi_j = \Pi_{j-1} R \quad \Rightarrow \quad \Pi_j = \Pi_1 R^{j-1} \quad \forall j \geq 2 \tag{6}$$

We can now write the balance equation for the repetitive part of the matrix as

$$\Pi_{j-1} A_0 + \Pi_j A_1 + \Pi_{j+1} A_2 = 0 \qquad \forall j \geq 2 \tag{7}$$

Inserting (6) in (7) and choosing j, we get the quadratic matrix equation

$$R^2 A_2 + R A_1 + A_0 = 0 \tag{8}$$

Once we determine a solution for R, we can examine the boundary section of the generator matrix. Using the balance equation again, we can write

$$\begin{bmatrix} \Pi_0 & \Pi_1 \end{bmatrix} \begin{bmatrix} B_{00} & B_{01} \\ B_{10} & A_1 + R A_2 \end{bmatrix} = \mathbf{0}^T$$

We now use the fact that Π is normalized, that is $\Pi_0 \mathbf{1} + \Pi_1 (I - R)^{-1} \mathbf{1} = 1$ and replace the first column of $\begin{bmatrix} B_{00} \\ B_{10} \end{bmatrix}$ with the above expression, giving us

$$\begin{bmatrix} \Pi_0 & \Pi_1 \end{bmatrix} \left[\begin{array}{c|c|c} \mathbf{1} & B_{00}^* & B_{01} \\ (I - R)^{-1} \mathbf{1} & B_{10}^* & A_1 + R A_2 \end{array} \right] = \left[\begin{array}{c|c} 1 & \mathbf{0}^T \end{array} \right] \tag{9}$$

Consequently, solving (8) for R (determining Π_k for $k \geq 2$) and then using it to solve (9) for Π_0 and Π_1 fully determines the stationary distribution. In this example, B_{00} only had one column, and thus is entirely replaced by the normalization, leaving us with

$$\begin{bmatrix} \Pi_0 & \Pi_1 \end{bmatrix} \left[\begin{array}{c|c} \mathbf{1} & B_{01} \\ (I - R)^{-1} \mathbf{1} & A_1 + R A_2 \end{array} \right] = \left[\begin{array}{c|c} 1 & \mathbf{0}^T \end{array} \right]$$

3.3 Series queue with a bounded switching curve

Consider a series queue with a switching curve $s(x)$ given by $u_1(x) = 1$ if $x_2 < s(x_1)$. Assume that s is non-decreasing and bounded. The recurrent states are $\{x : x_2 \leq s(x_1 + 1)\}$. Arrange the states of the system lexicographically in the generator matrix, numbering them $k := 1 \rightarrow (0,0), 2 \rightarrow (0,1),\ \ldots\ ,\ s(1) - 1 \rightarrow (0, s(1)),\ s(1) \rightarrow (1,0),\ s(1) + 1 \rightarrow (1,1),\ \ldots$ Let us also define the coordinate functions $x_1(k)$, $x_2(k)$ such that if $k \rightarrow (i,j)$, $x_1(k) := i$, $x_2(k) := j$.

We know that in a series queue with 2 servers, there are at most three different possible transitions from each state to a different state: a new element could arrive in the queue, server 1 could finish processing a job, or server 2 could finish processing. These three transitions are represented in the generator matrix by

$$\begin{aligned} &[I - P]_{k, k + s(x_1(k)) + 1} = -\lambda \quad \forall\ k \\ &[I - P]_{k, k-1} = -\mu_2 \quad \forall\ k \text{ s.t. } x_2(k) \neq 0 \\ &[I - P]_{k, k - s(x_1(k)) + 1} = -\mu_1 \quad \forall\ k \text{ s.t. } x_1(k) \neq 0 \text{ and } x_2(k) < s(x_1(k) - 1) \end{aligned}$$

Adding the appropriate self-transitions,

$$[I-P]_{k,k} = -\sum_j [I-P]_{k,j}$$

If n is the maximum buffer size of the buffers, the A_0, A_1, and A_2 blocks in the generator matrix are square $(n+1)\times(n+1)$ matrices with non-zero elements

$$A_0 = -\lambda I_{n+1}$$

$$A_1(1,1) = \lambda + \mu_2 \qquad A_1(2,2) = \lambda + \mu_1 \qquad A_1(i,i) = \lambda + \mu_1 + \mu_2 = 1 \text{ if } \Lambda = 1, \quad 3 \le i \le n+1$$
$$A_1(i+1,i) = -\mu_2, \quad 2 \le i \le n$$
$$A_1(n+1,1) = -\mu_1$$

$$A_2(1,n+1) = -\mu_2$$
$$A_2(i,i+1) = -\mu_1, \quad 2 \le i \le n$$

4 Performance bounds using the matrix analytic method

The matrix analytic method provides us with a convenient way to calculate π_h for reentrant line queueing networks under specific policies. However, computing (3) requires the generator matrix under the optimal policy. One possible approximation is to write $I - P_{h^*}$ as the sum of $n+1$ component matrices (where n is the number of servers) T_{μ_i}, each one associated with a transition, and only retain the components which are unaffected by the chosen policy. If we define $\lambda = \mu_0$, (1) gives

$$J_h - J^* = \pi_h(I - P_{h^*})h^* = \pi_h(I - \sum_{i\in C} T_{\mu_i} - \sum_{j\notin C} T_{\mu_j})h^* \le \pi_h(I - \sum_{j\notin C} T_{\mu_i})h^* \tag{10}$$

where $C := \{i : \mu_i \text{ is controlled}\}$. The following section attempts to compare the efficiency of the approximations from Section 2 with the one above.

4.1 Example: Series 2 with a buffer size of one

Consider a series 2 queueing network. It is convenient to rewrite (1) in its summation form. Let R be the set of recurrent states under u_h. Then, if $p^*(x,y)$ is the transition probability from state x to state y under the optimal policy,

$$J_h - J^* = \sum_{x\in R} \pi(x) \sum_y p^*(x,y)h^*(y)$$

Let us define $u_i := 1$ if an action is allowed in state (x_1, x_2) under u_h and 0 otherwise. We can similarly define u_i^* for the policy u_{h^*}. In our example, sub-optimality is given by

$$J_h - J^* = \sum_{x_1=0}^{\infty} \sum_{x_2=0}^{2} \pi_h(x_1,x_2) \left[h^*(x_1,x_2) - \lambda h^*(x_1+1,x_2) - \mu_1 h^*(x_1-u_1, x_2+u_1) - \mu_2 h^*(x_1, x_2-u_2)\right]$$

Equation (3) can also be put into summation form,

$$J_h - J^* \leq \sum_{x_1=0}^{\infty} \sum_{x_2=0}^{2} \pi_h(x_1,x_2)\mu_1([h^*(x_1-u_1,x_2+u_1) - h^*(x_1-u_1^*,x_2+u_1^*)]$$
$$- [h(x_1-u_1,x_2+u_1) - h(x_1-u_1^*,x_2+u_1^*)])$$

Removing the sub-stochastic matrix T_{μ_1} from P_{h^*}, the bound (1) becomes

$$J_h - J^* \leq \pi_h(I - T_\lambda - T_{\mu_2})h^*$$

which, in summation form, gives us

$$J_h - J^* = \sum_{(x_1,x_2)\in R} \pi_h(x_1,x_2)\left[h^*(x_1,x_2) - \lambda h^*(x_1+1,x_2) - \mu_2 h^*(x_1,x_2-u_2)\right] \tag{11}$$

In order to use the matrix analytic method to calculate π_h, we used a quadratic h as in Section 3.1. To be bounded, the switching curve must be horizontal, i.e. $q_{12} = q_{11}$. We used an approximate linear program to generate h, setting parameters determined in [14] to meet the criteria on Q. J^* and h^* were calculated numerically using dynamic programming algorithms. We compared the two bounds with $J_h - J^*$ for four different cases. The cases were chosen so that their optimal switching curves had different shapes – in cases 1 and 2, the optimal switching curve appears quasi-logarithmic, very close to s_h. In case 3 the switching curve appears approximately linear, while in case 4 the optimal switching curve is far above s_h. The results are displayed in Figure 4.1.

Cases	λ	μ_1	μ_2	c_1	c_2	J_h	J^*	(3)	(3)/Δ_J
1	0.18182	0.36364	0.45454	1	5	1.26769	1.24144	.09312	355%
2	.22535	.42254	.35211	1	2	2.07874	1.26916	1.36509	169%
3	0.125	0.25	0.625	1	5	.47226	.46665	.01891	337%
4	0.13793	0.17241	0.68966	1	1.2	.71163	.59315	.1419	120%

Figure 1: Numerical results for series 2

In all four numerical experiments, the approximation (3) performed better than (11) by multiple orders of magnitude, therefore the numerical computations of (11) were not included in this paper. However, let us note that (3) performed reasonably well, with an error under 400%.

5 Summary and areas for future research

In this paper we have developed two performance bounds and tested their accuracy for a specific class of h functions. Empirical tests showed that our first performance bound, (3), was reasonably accurate, consistently being within 400% of the exact value. However, our approximation of a performance bound, (10), was less accurate by multiple orders of magnitude. Unfortunately, we were unable to compare these two bounds with the bound from [5] because the h functions that we chose to consider did not satisfy the necessary assumptions. Nevertheless, it can be easily shown that (3) is tighter than (4).

We also discussed the sensitivities of our bound to different states as a function of the two switching curves. Provided that the maximum buffer size under our policy is less than the optimal switching curve and that h is chosen such that all buffers but the first are finite, one can calculate a bound for the sub-optimality of the h approximation as a function of $(h^* - h)$ on a very small section of the state space. If the aforementioned condition is not satisfied, computing the bound would require additional knowledge of the optimal switching curve. Further research could test the accuracy of this bound in numerical examples, varying either the size of the problem or the height of the switching curve associated with h. The matrix analytic method and the performance bound both still work for arbitrarily large problems, as long as the conditions from Section 3.1 are satisfied. Also, it would be of interest to reevaluate the bound using other approximations or bounds on π_h that weren't quite as restrictive on the u_h policy.

References

[1] Bertsimas, D., Paschaladis, I. Ch., & Tsitsiklis, J. N. (1994). *Optimization of multiclass queueing networks: Polyhedral and non-linear characterizations of achievable performance.* Ann. Appl. Prob., 4, 43-75.

[2] Bertsimas, D., Gamarnik, D., & Tsitsiklis, J. N. (2001). *Performance of multiclass queueing networks via piecewise linear Lyapunov functions.* Ann. Appl. Prob., 11, 1385-1428.

[3] Carter, L. (2007). *Approximate Linear Programming and Upper Bounds.* Gordon College.

[4] de Farias, D. P., & Van Roy, B. (2003). *Approximate linear programming for average cost dynamic programming.* Advances in Neural Information Processing Systems 15, MIT Press.

[5] de Farias, D. P., & Van Roy, B. (2003). *The linear approach to approximate dynamic programming.* Oper. Res., 51, 850-865.

[6] de Farias, D. P., & Van Roy, B. (2005). *A linear program for Bellman Error minimization with performance guarantees.* Advances in Neural Information Processing Systems 17, MIT Press.

[7] Kumar, P. R., Meyn, S.P. (1996). *Duality and linear programs for stability and performance analysis of queueing networks and scheduling policies.* IEEE Trans. Automat. Control, 41, 4-17.

[8] Kumar, S., & Kumar, P. R. (1994). *Performance bounds for queueing networks and scheduling policies.* IEEE Trans. Automat. Control, AC-39, 1600-1611.

[9] Latouche, G., & Ramaswami, V. (1987). *Introduction to Matrix Analytic Methods in Stochastic Modeling.* Society for Industrial Mathematics.

[10] Lui, John C.S. (2005). *Matrix-Geometric Analysis and Its Applications.* The Chinese University of Hong Kong. PowerPoint presentation.

[11] Morrison, J. R., Kumar, P. R. (1999). *New linear program performance bounds for queueing networks.* J. Optim. Theory Appl., 575-597.

[12] Morrison, J. R., Kumar, P. R. (2002). *Linear programming and performance bounds for Markov chains with polyhedrally translation invariant transition probabilities and applications to unreliable manufacturing systems and enhanced wafer fab models.* Proceedings of IMECE2002, New Orleans, LA.

[13] Nasman, J. (2007). *Approximate Linear Programming and Bellman Error.* Gordon College.

[14] Russell, M. (2008). *Series 2 Quadratic ALP: Analytic Solutions and Bellman Error.* Gordon College.

[15] Veatch, M. (2005). *Approximate dynamic programming for networks: fluid models and constraint reduction.* Gordon College.

[16] Veatch, M. (2008). *Notes on Numerous Different Topics.* Gordon College.