Probability in the Engineering and Informational Sciences, **21**, 2007, 19–46. Printed in the U.S.A.

ADMISSION CONTROL WITH INCOMPLETE INFORMATION TO A FINITE BUFFER QUEUE

DOROTHÉE HONHON*
Department of Information, Risk and Operations Management
McCombs School of Business
The University of Texas at Austin, Austin, TX 78712
E-mail: dorothee.honhon@mccombs.utexas.edu

SRIDHAR SESHADRI
Department of Information
Operations and Management Science
Leonard N. Stern School of Business
New York University, New York, NY 10012

We consider the problem of admission control to a multiserver finite buffer queue under partial information. The controller cannot see the queue but is informed immediately if an admitted customer is lost due to buffer overflow. Turning away (i.e., blocking) customers is costly and so is losing an admitted customer. The latter cost is greater than that of blocking. The controller's objective is to minimize the average cost of blocking and rejection per incoming customer. Lin and Ross [11] studied this problem for multiserver loss systems. We extend their work by allowing a finite buffer and the arrival process to be of the renewal type. We propose a control policy based on a novel state aggregation approach that exploits the regenerative structure of the system, performs well, and gives a lower bound on the optimal cost. The control policy is inspired by a simulation technique that reduces the variance of the estimators by not simulating the customer service process. Numerical experiments show that our bound varies with the load offered to the system and is typically within 1% and 10% of the optimal cost. Also, our bound is tight in the important case when the cost of blocking is low compared to the cost of rejection and the load offered to the system is high. The quality of the bound degrades with the degree of state aggregation, but the computational effort is comparatively small. Moreover, the control policies that we obtain perform better compared to a heuristic suggested by Lin and Ross. The state aggregation technique developed in this article can be used more generally to solve problems in which the objective is to

*This work was done while the first author was at the Stern School of Business, NYU.

 0269-9648/07 $16.00

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

control the time to the end of a cycle and the quality of the information available to the controller degrades with the length of the cycle.

1. INTRODUCTION

We consider the problem of admission control to a finite buffer queue under partial information. In this problem, a controller decides whether to admit a customer. The controller cannot see the queue but is informed immediately if an admitted customer is rejected due to buffer overflow. Turning away (i.e., blocking) customers is costly and so is losing an admitted customer due to buffer overflow. The latter cost is greater than that of blocking. The controller's objective is to minimize the average cost of blocking and rejection.

This problem was originally posed in the context of computer and telecommunications networks. Lin and Ross [11] established that the optimal policy is of a threshold type for single-server loss systems (i.e., a queue with no buffer). In this policy, arrivals are blocked for a fixed period of time following a rejection, after which all arrivals are admitted until a new rejection takes place. They showed that the optimal policy does not have a similar structure for multiserver systems but that it can be obtained by solving a dynamic program (DP). Unfortunately, the state space of the DP grows exponentially with the number of servers and the size of the buffer.

We consider an extension suggested in Lin and Ross [11] to the case when the queue has a finite buffer and the arrival process is of the renewal type. It is extremely difficult to solve this problem to optimality. For example, to determine the optimal policy to a reasonable accuracy for a system in which the combined size of the buffer and the number of servers is equal to 10, the number of states in the corresponding DP can exceed 30 million. We therefore propose to solve the DP using a new state aggregation technique. The novelty of our technique is that it yields both a control policy and a lower bound on the average cost. Moreover, the computational effort does not grow with the size of the buffer or the number of servers. We find in our experiments that the bound given by this technique is within 1%–10% of the optimal cost. In many instances, our method improves on the performance of the heuristic proposed by Lin and Ross. The bound is tight when the cost of blocking is low compared to the cost of rejection and the control policy performs especially well when the cost of blocking is high and the offered load is high. Finally, we also provide an efficient simulation technique to compute the average cost per customer without simulating the customer service process.

The key observation we use in this article is that the state space for the controlled Markov chain can be either the set of events since the last rejection or the probability distribution of the number of customers in the system, computed at arrival epochs of customers. The latter, although convenient to numerically determine the optimal control to any given accuracy, grows very quickly with the size of the buffer and the number of servers. The former grows with the number of events since the last rejection. However, if we retain only the information about the last few admit-

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

ted customers, we can curtail the growth of the state space. If we then reconstruct the events based on this limited information (i.e., reconstruct the entire set of admission events since the last rejection) in a careful manner, taking into account the structure of the stochastic system, we show that we can achieve a lower bound on the minimum average cost per incoming customer, as well as a good control policy. The main contribution of this article is the idea of state reconstruction using limited information as a basis for state aggregation.

The method developed by us can be used to bound the average cost in systems in which the objective is to control the time to the end of a cycle and the quality of the information available to the controller degrades with the length of the cycle. Examples of such applications include equipment maintenance, cash management problems, and inventory management problems. In the case of equipment maintenance, the problem is to minimize the average cost of maintenance and breakdown repair. One might have to schedule preventive repairs on a machine with imperfect information about the state of the equipment. The repairs have the effect of delaying failure and a cost that is lower than that of a breakdown repair. Regeneration occurs when the machine finally breaks down and has to be replaced. The problem is to minimize the average cost of maintenance and breakdown repair. In the cash management setting, one might have to decide to give loans without knowing the exact amount of resources available (i.e., without information about the actual cash inflows from interest and repayment of the principal). By refusing to grant certain loans, the controller can extend the time until the firm runs out of resources. If there are inadequate funds and a loan is sanctioned, the firm incurs a significant cost and has to add assets to the balance sheet. The problem is to determine the loans to sanction until resources run out. Finally, one can apply our solution technique to decide whether to fill an order without knowing the actual number of items in stock. By refusing to take an order, the controller is able to delay the time until a customer whose order was taken experiences shortage. In this case, a fixed cost is incurred to replenish the inventory.

The rest of the article is organized as follows. A review of the relevant literature is given in Section 2. Section 3 presents the model. Section 4 is devoted to analyzing the optimal policy. Sections 5–8 present our main results. Numerical examples are given in Section 9.

2. LITERATURE REVIEW

There is a growing literature on control of queues with delayed or incomplete information on the state of the system. These articles consider either routing or admission (or both) control issues. In this article we focus on admission control.

Altman, Marquez, and Yechiali [4] considered the optimal routing and admission control in a multiserver loss system when the controller has delayed information or no information at all. With delayed information, they compare a "round-robin" policy, where customers are sent to servers before the information about their state becomes available, with a "wait" policy, where customers are first

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

grouped in a finite buffer then sent to a server once information becomes available. They established the existence of a threshold on delays below which the wait policy gives a higher throughput. When the controller has no information at all, they proposed a timer mechanism where the interadmission time is set equal to the maximum of the interarrival time and some random variable and customers are assigned to servers in a cyclic fashion. For a given distribution of the random variable, they show how to choose the parameters of the timer to achieve the maximum throughput. Litvak and Yechiali [13] performed the same type of analysis in a system with limited buffer and the objective of minimizing the average queue length. In their system, the information on service completions is delayed by a random time. They showed that there exists a threshold value of the average information delay below which the wait policy performs better.

The problem of admission control with delayed information in a single-server discrete-time queue has been studied by Altman, Kofman, and Yechiali [2], Altman and Stidham [6], Altman and Nain [5], Altman and Basar [1], as well as Kuri and Kumar [9]. In all these papers the information on queue lengths is delayed by an arbitrary number of periods. Altman, Kofman, and Yechiali [2] focus on computing the distribution of the queue length in steady state. Altman and Stidham [6], Altman and Koole [3], and Altman and Nain [5] prove, in different contexts, that the optimal policy is characterized by a threshold on the queue length. Altman and Koole [3] consider the control of a random walk where the controller has noisy information on the current state of the queue but full information on previous states. They also establish the optimality of a threshold policy in this setting.

Kuri and Kumar [10] studied the control of arrivals to a queue with delayed information. They argued that the standard approach for solving a partially observable controlled Markov chain is to define the "state" of the system to be the conditional probability measure on the space of the underlying unobservable system. For example, in their problem (and in ours), the state is the distribution of the number of customers in the system. They suggested that an alternate and direct formulation is to appropriately define an enlarged state such that it captures all of the relevant information. They gave several references in which such an approach has been successfully used. For example, the pattern of arrivals to the queue along with the delay in information is sufficient to predict the state of the queue in their (and our) model. Therefore, this enlarged state space can serve to develop the optimal control. Their model differed from ours in several ways. They modeled a discrete time system with a single server, whereas we consider multiple servers. They modeled a system with delayed information, whereas we model one with partial information. Their main insight was also different. They show that even though there are many different arrival patterns, the control corresponding to subsets of patterns is the same. They sought to exploit this property; therefore, they explored state-space reduction, not state-space aggregation. Finally, they provided a lower and upper bound on the optimal cost based on a policy that never accepts customers. Our lower bound is based on changing the transition probability in each aggregate state and is obtained by solving a DP. Despite these differences, we believe that their

insight that the direct formulation will give valuable results is the key to our results as well.

The work of Lin and Ross described in Section 1 is different from all of the work cited above because it deals with incomplete and not delayed information. In Lin and Ross [12], they extended their work and showed that a threshold policy, which is optimal in the case of a single server with exponential service time distribution, is not necessarily optimal with general service time distributions. In this latter work, they also provided a sufficient condition for the existence of an optimal policy that is of the threshold type.

3. PROBLEM SETTING

We consider a multiserver queue with finite buffer that has s identical servers. The size of the buffer is b, so that the system is full when there are $b + s$ customers. The arrival process of customers to this queuing system is an ordinary renewal process such that interarrival times are independent and identically distributed (i.i.d.) with distribution F and mean $1/\lambda$. The service time distribution is exponential with mean $1/\mu$.

A controller observes the arrival process. He knows the parameters of the arrival and service processes. However, he cannot observe the number of customers in the system and cannot observe departures from the system. A control is exercised at each arrival instance: The controller has to decide whether to admit or block an arriving customer. The cost of blocking a customer is $c < 1$ and that of admitting a customer is zero. If the customer is blocked, she leaves the system immediately without being served. If the customer is admitted and does not find the buffer full, she is *accepted*; that is, she joins the queue or starts service if a server is free. If the customer finds the system full, she is *rejected* at a cost of 1 and instantly leaves the system. The controller is informed immediately of the rejection. The objective of the controller is to minimize the average cost of blocking and rejection per incoming customer.

We consider systems in which it is not optimal to block all incoming customers and restrict our attention to control policies under which the expected time until a customer is rejected is finite. When the controller is informed of a rejection, he knows that at that instant there are exactly $b + s$ customers in the system. In particular, all servers are busy, and because service times are exponential, their remaining processing times are exponentially distributed with mean $1/\mu$, independent of one another and of the past. Also, the next arrival takes place after one interarrival time, which is independent of the past as well. Given these observations, the controller's estimate of the distribution of the number of customers in the system does not depend on what happened before the rejection occurred. It follows that the only information the controller needs to consider is the set of events *since the last rejection*. Moreover, customers that are blocked do not impact the distribution of the number of customers in the system; therefore, the controller only needs to consider the *admission epochs* since the last rejection and the elapsed time since the last rejection. This information is sufficient to compute the probability distribution of the number of customers in the system, at every arrival epoch.

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

Therefore, we define a *state* to be the set of admission epochs since the last rejection along with the elapsed time since the last rejection, or, equivalently, the probability distribution of the number of customers in the system calculated from this information. These alternate definitions for the state space are examined in greater detail in Section 4. For now, let $\mathcal{S}$ denote the state space, under either definition.

Let Ψ be the set of control policies adapted to the processes of the number of arriving, blocked, and rejected customers since the last rejection. Let $\mathcal{A}(i)$ denote the action space for state $i \in \mathcal{S}$; we have

$$\mathcal{A}(i) = \mathcal{A} \equiv \{\text{“admit”}, \text{“block”}\}.$$

Because the cost is computed per customer arrival, we define stage k of the DP to be the decision epoch when the kth customer arrives to the system. Let i_k and a_k respectively denote the state visited and the action taken in stage k. Let $g(i_k, a_k)$ denote the cost incurred at stage k, in state $i_k \in \mathcal{S}$, and under action $a_k \in \mathcal{A}$. It is equal to

$$g(i_k, a_k) = \begin{cases} c & \text{if } a_k = \text{“block”} \\ f(i_k) & \text{if } a_k = \text{“admit”}, \end{cases}$$

where $f(i_k)$ is the probability that the buffer is full in state i_k.

Let $J(\pi)$ be the average cost per customer under control policy $\pi \in \Psi$:

$$J(\pi) = \limsup_{n\to\infty} \frac{1}{n} \mathbb{E}_\pi \left[\sum_{k=1}^{n} g(i_k, a_k) \right].$$

Let δ be the minimum average cost per incoming customer:

$$\delta \equiv \inf_{\pi\in\Psi} J(\pi).$$

Also, for $0 < \beta < 1$, let $J_\beta(i)$ be the optimal total discounted cost for initial state $i \in \mathcal{S}$:

$$J_\beta(i) = \inf_{\pi\in\Psi} \mathbb{E}_{i,\pi} \left[\sum_{k=1}^{\infty} \beta^k g(i_k, a_k) \right].$$

Let $V_\beta(i)$ be the relative discounted value function in state $i \in \mathcal{S}$, given by

$$V_\beta(i) = J_\beta(i) - \inf_{j\in\mathcal{S}} J_\beta(j).$$

In other words, $V_\beta(i)$ is equal to the difference in optimal total discounted cost between two systems: one that starts in state i and one that starts in the state for which the total discounted cost is minimum. This quantity is bounded by the sums of costs in the two systems, each *up to* the next time a rejection occurs, because from that point onward, the probabilistic evolutions of the two systems are the same.

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

Using this observation and the assumption that the expected time to a rejection is finite and because per-stage costs are bounded, we obtain that

$$\sup_{\beta<1} V_\beta(i) < \infty, \quad \forall i \in \mathcal{S}.$$

Therefore, the system satisfies the "boundedness" condition (B) of Schäl [17]. Moreover, since the set of actions available in each state is finite, the system also satisfies condition (S) of Schäl [17]. Hence, by Theorem 3.8 of Schäl, there exists an admissible stationary policy u^*, which is average optimal, in the sense that

$$J(u^*) = \delta.$$

Therefore, we limit the search for an optimal policy to the set of admissible stationary policies, denoted as Ψ_s. In what follows, u denotes an admissible stationary policy.

4. TWO REPRESENTATIONS OF THE STATE SPACE

As mentioned in Section 3, there are two possible representations for the state space. We examine each below.

4.1. State Defined as Information Since Last Rejection

Let time 0 correspond to the epoch of the last rejection. Let t denote the time since the last rejection. Let n be the number of admitted customers since the last rejection. Let t_j for $j = 1, \ldots, n$ be the admission epoch of the jth customer. A state is represented by the vector $(t, n, t_1, \ldots, t_n)$. The state space $\mathcal{S}$ is given by

$$\mathcal{S} = \{(t, n, t_1, \ldots, t_n) : n = 0,1,2,\ldots,0 < t_1 \leq \cdots \leq t_n \leq t\}. \quad \textbf{(1)}$$

The controller uses this vector to compute the probability distribution of the number of customers in the system. We propose the following method for computing the probability distribution. First, from $(t, n, t_1, \ldots, t_n)$ we compute the $(b + s + 2)$-dimensional vector of probabilities $\mathbf{p}(t, n, t_1, \ldots, t_n) = (p_x(t, n, t_1, \ldots, t_n); x = 0, \ldots, b + s + 1)$, where $p_x(t, n, t_1, \ldots, t_n)$ for $0 \leq x \leq b + s$ is the probability that the system has x customers at time t given that customers were *admitted* (but not necessarily accepted) at times $t_1, \ldots, t_n$. Also, $p_{b+s+1}(t, n, t_1, \ldots, t_n)$ is the probability that the system has incurred a rejection since time 0 given that customers were *admitted* at times $t_1, \ldots, t_n$. We refer to vector $\mathbf{p}$ as the vector of *unconditional* probabilities associated with state $(t, n, t_1, \ldots, t_n)$ because it does not factor in the condition that the arrivals were accepted. There is a nice algebraic method to determine $\mathbf{p}$, which we describe next. This vector will then be used to derive the vector of *conditional* probabilities $\mathbf{r}$, which factors in the condition that the arrivals were accepted.

Let $\mathbf{e}^{b+s}$ be a $(b + s + 2)$-row vector with 1 in the $(b + s)$st column and zero otherwise. At the time a customer is rejected, the controller sets the probability distribution of the number of customers in the system equal to $\mathbf{e}^{b+s}$ because he

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

knows that there are exactly $b + s$ customers in the system at this time. At each customer arrival epoch, the controller updates the distribution, using matrix $\mathbf{M}$ defined below, to account for the fact that customers may have left the system during the inter-arrival time. Finally every time he admits a customer he updates the distribution using matrix $\mathbf{Z}$ below. The vector of unconditional probabilities $\mathbf{p}$ associated with state $(t, n, t_1, \ldots, t_n)$ is computed as follows:

$$\mathbf{p}(t, n, t_1, \ldots, t_n) = \mathbf{e}^{b+s}\mathbf{M}(t_1)\mathbf{Z}\mathbf{M}(t_2 - t_1)\mathbf{Z} \ldots \mathbf{M}(t_n - t_{n-1})\mathbf{Z}\mathbf{M}(t - t_n) \quad (2)$$

for all $(t, n, t_1, \ldots, t_n) \in \mathcal{S}$.

In this expression, $\mathbf{M}(\tau)$ is the $(b + s + 2) \times (b + s + 2)$ transition probability matrix such that component $M_{x,y}(\tau)$ is the probability that the number of customers in the system changed from x to y during an interarrival time of τ units. It can be written as

$$M_{x,y}(\tau) = \begin{cases} C_y^x (e^{-\mu\tau})^y (1 - e^{-\mu\tau})^{x-y}, & 0 \le x \le s, 0 \le y \le x \\ \displaystyle\int_0^\tau \frac{(s\mu)^{(x-s)}}{(x-s-1)!} z^{x-s-1} e^{-s\mu z} C_y^s (e^{-\mu(\tau-z)})^y (1 - e^{-\mu(\tau-z)})^{s-y}\, dz, & \\ & s+1 \le x \le b+s, 0 \le y \le s \\ \dfrac{(\mu s\tau)^{x-y}}{(x-y)!} e^{-\mu s\tau}, & s+1 \le x \le b+s, s+1 \le y \le x \\ 1, & x = 0, y = 0 \\ 1, & x = b+s+1, y = b+s+1 \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

In the case of a single server, it simplifies to

$$M_{x,y}(\tau) = \begin{cases} 1 - \displaystyle\sum_{k=0}^{x-1} \mathbb{P}(D_\tau = k), & 1 \le x \le b+1, y = 0 \\ \mathbb{P}(D_\tau = x - y), & 1 \le y \le x, 1 \le x \le b+1 \\ 1, & x = 0, y = 0 \\ 1, & x = b+2, y = b+2 \\ 0 & \text{otherwise,} \end{cases}$$

where D_τ is a Poisson random variable with mean $\mu\tau$.

$\mathbf{Z}$ is the $(b + s + 2) \times (b + s + 2)$ transition probability matrix that represents the impact on the probability distribution due to the admission of a customer:

$$Z_{x,y} = \begin{cases} 1, & y = x + 1, x < b + s + 1, \\ 1, & y = x = b + s + 1 \\ 0 & \text{otherwise.} \end{cases}$$

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

If $x < b + s$, then admitting a customer increases the number of customers in the system by one. If $x = b + s$, then the admitted customer is rejected.

Let the $(b + s + 1)$-dimensional vector $\mathbf{r}(t, n, t_1, \ldots, t_n) = (r_x(t, n, t_1, \ldots, t_n); x = 0, \ldots, b + s)$, be the vector of *conditional* probabilities because it factors in the condition that the arrivals were accepted. $r_x(t, n, t_1, \ldots, t_n)$ is the probability that there are x customers in the system t units of time after the last rejection given that n customers were accepted at epochs $t_1, \ldots, t_n$.

From the vector $\mathbf{p}(t, n, t_1, \ldots, t_n)$ the controller obtains the vector $\mathbf{r}(t, n, t_1, \ldots, t_n)$ by dividing the first $b + s + 1$ components of $\mathbf{p}$ by the probability that the system has not incurred a rejection so far; that is, $1 - p_{b+s+1}(t, n, t_1, \ldots, t_n)$:

$$r_x(t, n, t_1, \ldots, t_n) = \frac{p_x(t, n, t_1, \ldots, t_n)}{1 - p_{b+s+1}(t, n, t_1, \ldots, t_n)} \qquad x = 0, \ldots, b + s. \tag{4}$$

One reason that we introduced the unconditional vector $\mathbf{p}$ is for ease of computation of $\mathbf{r}$. Another reason will become apparent in Section 6.

The state space $\mathcal{S}$ defined in (1) is infinite because each component of the state vectors is continuous and unbounded and the size of the vector itself is not bounded from above. For this reason, Lin and Ross [11] used the second definition of a state for solving the problem.

4.2. State Defined as Probability Distribution

Let a state be the vector of conditional probabilities $\mathbf{r}$, as defined in (4). The set of all possible values for this vector is continuous and infinite. In order to make this state space discrete (and finite), select a positive integer m, called the discretizing constant, such that the probabilities are rounded off to a multiple of $1/m$ (see details in Lin and Ross [11, p. 648]). Let $\mathcal{S}_m$ be the set of discretized vector of conditional probabilities obtained using the discretizing constant m.

As explained in Section 3, the cost of policy u in state $\mathbf{r} \in \mathcal{S}_m$ is given by

$$g(\mathbf{r}, u(r)) = \begin{cases} c & \text{if } u(\mathbf{r}) = \text{“block”} \\ f(\mathbf{r}) & \text{if } u(\mathbf{r}) = \text{“admit”}, \end{cases}$$

where $f(\mathbf{r})$ is the probability that the buffer is full in state $(\mathbf{r})$ and is given by

$$f(\mathbf{r}) = r_{b+s};$$

that is, $f(\mathbf{r})$ is given by the $(b + s)$th component of the state vector.

Let $P(\mathbf{r}, \tilde{\mathbf{r}}; u)$ be the transition probability, under control u, from state $\mathbf{r}$ to state $\tilde{\mathbf{r}}$, where $\mathbf{r}, \tilde{\mathbf{r}} \in \mathcal{S}_m$. The components of the matrix P are functions of $f(\mathbf{r})$ and λ.

Let $J_m(u)$ denote the average cost per customer, under stationary control policy u defined on state space $\mathcal{S}_m$. Let δ_m be the minimum average cost per customer when the state space is $\mathcal{S}_m$:

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

$$\delta_m = \inf_{u \in \Psi_s} J_m(u).$$

Let u_m^* be the control that achieves this minimum, such that $J_m(u_m^*) = \delta_m$. The Bellman equation associated with this discrete-time problem is given by

$$V_m(\mathbf{r}) + \delta_m = \min_{u \in \Psi_s} \left\{ g(\mathbf{r}, u) + \sum_{\tilde{\mathbf{r}} \in \mathcal{S}_m} P(\mathbf{r}, \tilde{\mathbf{r}}; u) V_m(\tilde{\mathbf{r}}) \right\}, \qquad \forall \mathbf{r} \in \mathcal{S}_m, \tag{5}$$

where $V_m(\mathbf{r})$ is the relative value function in state $\mathbf{r}$.

The minimum average cost δ_m and optimal control policy u_m^* on state space $\mathcal{S}_m$ can be obtained using value iteration. We assume that as $m \to \infty$, δ_m tends to δ and u_m^* tends to u^*. It follows that for large m, solving the problem with state space $\mathcal{S}_m$ gives a good approximation of u^*. However, for a given precision m, the state space becomes very large as either the buffer size b or the number of servers s increases (see Section 1). In particular, the number of states in $\mathcal{S}_m$ is equal to

$$\binom{m + b + s + 1}{b + s}.$$

For this reason we provide an alternative method to obtain a good control policy for which the size of the state space is independent of b and s. Our method consists of aggregating the state space $\mathcal{S}$ defined in (1).

5. POLICY BASED ON STATE AGGREGATION

If we use the state space $\mathcal{S}$ defined in (1), the controller has to remember all of the admission epochs since the last rejection. We suggest a method in which the controller keeps in memory a finite number of the most recent admission epochs. Formally, let l be the size of the controller's memory (i.e., the maximum number of past admission epochs he remembers). For example, $l = 1$ means that states are of the type (t, n, t_n) when $n \geq 1$ and they are called *aggregate* states with memory size 1. In general for $l < \infty$, aggregate states with memory size l are of the type $(t, n, t_{(n-l+1)\wedge 1}, \ldots, t_n)$ for $n \geq 1$ (where $a \wedge b \equiv \min(a, b)$) and $(t,0)$ otherwise. To simplify the notation, we denote a generic aggregate state with memory size l as $(t, n, t_{n-l+1}, \ldots, t_n)$ with the understanding that the size of each vector is $n + 2$. The states described in Section 4.1, $(t, n, t_1, \ldots, t_n)$, are such that $l = \infty$ and are referred to as *full memory states*.

The variables in this section are denoted with a superscript l to refer to the size of the controller's memory. Note that when $l = \infty$, the variables are equivalent to those defined in Section 4.1. Let $\mathcal{S}^l$ be the state space for memory size l:

$$\mathcal{S}^l = \{(t, n, t_{n-l+1}, \ldots, t_n) : n = 0,1,2,\ldots, 0 < t_{n-l+1} \leq \cdots \leq t_n \leq t\}. \tag{6}$$

Given the information that the system is currently in aggregate state $(t, n, t_{n-l+1}, \ldots, t_n) \in \mathcal{S}^l$ such that $n > l$, the controller does not have all of the

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

information necessary to compute the exact unconditional distribution of the number of customers in the system (i.e., $\mathbf{p}$). In particular, he does not remember when the first $n-l$ customers were admitted (i.e., $t_1, \ldots, t_{n-l}$). However, he can obtain an estimate of $\mathbf{p}$, denoted $\mathbf{p}^l$, by taking a weighted average of the vector of unconditional probabilities associated with all possible values of $t_1, \ldots, t_{n-l}$:

$$\mathbf{p}^l(t, n, t_{n-l+1}, \ldots, t_n) = \begin{cases} \displaystyle\sum_{t_1, \ldots, t_{n-l}: (t, n, t_1, \ldots, t_n) \in \mathcal{S}} w_{(t, n, t_1, \ldots, t_n)} \mathbf{p}(t, n, t_1, \ldots, t_n) & \text{if } n > l \\ \mathbf{p}(t, n, t_1, \ldots, t_n) & \text{otherwise,} \end{cases} \tag{7}$$

where $\mathbf{p}$ is defined in (2) and

$$\sum_{t_1, \ldots, t_{n-l}: (t, n, t_1, \ldots, t_n) \in \mathcal{S}} w_{(t, n, t_1, \ldots, t_n)} = 1.$$

The constants w should be interpreted as conditional probabilities or weights on each full memory state in which the values of $t_{n-l+1}, \ldots, t_n$ are equal to those in the aggregate space.

We define the probability that the buffer is full in the aggregate state $(t, n, t_{n-l+1}, \ldots, t_n)$ as

$$f^l(t, n, t_{n-l+1}, \ldots, t_n) = \frac{p^l_{b+s+1}(t, n+1, t_{n-l+2}, \ldots, t_n, t) - p^l_{b+s+1}(t, n, t_{n-l+1}, \ldots, t_n)}{1 - p^l_{b+s+1}(t, n, t_{n-l+1}, \ldots, t_n)}.$$

The numerator corresponds to the probability that the system has incurred a rejection when admitting the nth customer since time 0. By dividing this value by the probability that the system has not incurred any rejection before the arrival of the nth customer, we obtain the probability that the buffer is full at time t given that there has been no rejection since time 0.

The set $\mathcal{S}^l$ defined in (6) is infinite. To make it finite, we do two things. First, we discretize the distribution of the interarrival time F in steps of Δ; that is, the probability that the time until the next arrival is k for $k = \Delta, 2\Delta, \ldots$ is given by q_k, where

$$q_k = F(k) - F(k-1), \qquad k = \Delta, 2\Delta, \ldots.$$

Second, we bound the maximum number of admissions per cycle by T. We define the set of aggregate states $\mathcal{S}^l_{\Delta,T}$:

$$\mathcal{S}^l_{\Delta,T} = \{(t, n, t_{n-l+1}, \ldots, t_n) : t_{n-l+1}, \ldots, t_n, t \in \{\Delta, \ldots, \Delta T\}, n \in \{0, 1, \ldots, T-1\}, t_{n-l+1} + (l+1)\Delta \le \cdots \le t_n + \Delta \le t\}.$$

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

To avoid end effects due to the choice of T, we formulate the problem as a discrete-time semi-Markov decision process (SMDP). We describe the formulation briefly. Notice that aggregate states outside of $\mathcal{S}^l_{\Delta,T}$ can occur once time ΔT is exceeded. To deal with this, we assume that the controller admits every arriving customer in aggregate states outside $\mathcal{S}^l_{\Delta,T}$ until a rejection occurs. Let $\tau^l(t,n,t_{n-l+1},\ldots,t_n)$ be equal to 1 if the next arrival takes place within ΔT with probability 1 or 1 plus the expected number of transitions until the next rejection otherwise. The cost incurred in state $i=(t,n,t_{n-l+1},\ldots,t_n)\in\mathcal{S}^l_{\Delta,T}$ using control policy u is given by

$$g^l(i,u)=\begin{cases}\dfrac{1}{\tau^l(i)}\left(c+\displaystyle\sum_{k=T-t}^{\infty}q_k\right) & \text{if } u(i)=\text{“block”}\\[2ex] \dfrac{1}{\tau^l(i)}\left(f^l(i)+[1-f^l(i)]\displaystyle\sum_{k=T-t}^{\infty}q_k\right) & \text{if } u(i)=\text{“admit”}.\end{cases}$$

Also, let $P^l(i,\tilde{i};u)$ be the probability of going from aggregate state i to $\tilde{i}$, where $i,\tilde{i}\in\mathcal{S}^l_{\Delta,T}$, under control u.

Let $\delta^l_{\Delta,T}$ and $u^{l*}_{\Delta,T}$ respectively denote the minimum average cost per customer and the optimal stationary policy when $\mathcal{S}^l_{\Delta,T}$ is the state space and P^l is the transition probability matrix. The Bellman equation associated with the SMDP is the following (see Ross [14, p. 162]):

$$V^l(i)=\min_{u\in\Psi_s}\left\{g^l(i,u)\tau^l(i)-\delta^l_{\Delta,T}\tau^l(i)+\sum_{\tilde{i}\in\mathcal{S}^l_{\Delta,T}}P^l(i,\tilde{i};u)V^l(\tilde{i})\right\},\quad\forall i\in\mathcal{S}^l_{\Delta,T},$$

where $V^l(i)$ is the relative value function in aggregate state $i\in\mathcal{S}^l_{\Delta,T}$.

The problem can be transformed into an Markov decision process by allowing self-transitions. The corresponding Bellman equation is:

$$V^l(i)+\delta^l_{\Delta,T}=\min_{u\in\Psi_s}\left\{g^l(i,u)+\eta^l(i)P^l(i,i;u)V^l(i)+(1-\eta^l(i))\sum_{\substack{\tilde{i}\in\mathcal{S}^l_{\Delta,T}\\ \tilde{i}\neq i}}P^l(i,\tilde{i};u)V^l(\tilde{i})\right\},$$

$$\forall i\in\mathcal{S}^l_{\Delta,T},\qquad\textbf{(8)}$$

where

$$\eta^l(t,n,t_{n-l+1},\ldots,t_n)=1-\frac{1}{\tau^l(t,n,t_{n-l+1},\ldots,t_n)}$$

is the probability of staying in aggregate state $(t,n,t_{n-l+1},\ldots,t_n)$ and is such that the expected number of transitions in aggregate state $(t,n,t_{n-l+1},\ldots,t_n)$ is equal to $\tau^l(t,n,t_{n-l+1},\ldots,t_n)$.

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

The minimum average cost $\delta^l_{\Delta,T}$ and optimal control policy $u^{l*}_{\Delta,T}$ on state space $\mathcal{S}^l_{\Delta,T}$ can be obtained using value iteration. Also, as $T \to \infty$ and $\Delta \to 0$, $\delta^l_{\Delta,T}$ tends to δ^l, where

$$\delta^l = \inf_{u \in \Psi_s} J^l(u)$$

and $J^l(u)$ denotes the average cost per customer under stationary control policy $u \in \Psi_s$ defined on state space $\mathcal{S}^l$. Also, $u^{l*}_{\Delta,T}$ tends to u^{l*}, which is the control that minimizes the average cost for state space $\mathcal{S}^l$ [i.e., such that $J^l(u^{l*}) = \delta^l$].

The advantage of using this method is that the size of the state space does not depend on the size of the system (i.e., b and s). The disadvantage is the problem of defining the weights in (7). The conventional method for doing this is described below, but it does not have any guarantee of convergence or any result with regard to the performance of the resulting policy. An alternate method that has both of these properties is suggested in Section 7.

The conventional way to determine the weights is to use the following recursive procedure. First, fix a control u_k, where $k = 1$ at the first step. By simulation, estimate the steady-state probabilities $\nu^{u_k}_{(t,n,t_1,\ldots,t_n)}$ for every full memory state $(t, n, t_1, \ldots, t_n)$. Then compute the weights w^k at step k using

$$w^k_{(t,n,t_1,\ldots,t_n)} = \frac{\nu^{u_k}_{(t,n,t_1,\ldots,t_n)}}{\sum_{t_1,\ldots,t_{n-l}:(t,n,t_1,\ldots,t_n)\in\mathcal{S}} \nu^{u_k}_{(t,n,t_1,\ldots,t_n)}}.$$

Next, solve the DP in (8) to obtain the control u_{k+1} and repeat this procedure. It is not easy to provide a proof of convergence for this method (see Bertsekas [7, p. 44] for a discussion). Also, this method does not use knowledge of the system to determine the weights. Therefore, we propose an alternate method for selecting w that is inspired by the simulation algorithm given in the next section.

6. SIMULATION TO DETERMINE THE AVERAGE COST

In this section we describe an efficient simulation technique for estimating the average cost per incoming customer under policy u, for a given size of the controller's memory l. We refer to the time between two rejections as one *cycle*. Successive cycles length are i.i.d. because state transitions are Markovian and the control is a stationary policy. Also, cycles are assumed to have finite expected length.

In the algorithm below, we simulate the arrival process; that is, we generate random numbers according to the interarrival time distribution F. Note that we do not simulate the departure process. In other words, we do not determine whether each admitted arrival was accepted or not. However, we can compute the probability that a customer was accepted based on the knowledge of past admission epochs. The simulation of a cycle ends when the probability of no rejection in the cycle falls below a predetermined small value.

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

We keep track of two quantities in cycle k:

- A running estimate of the expected cycle length (in number of transitions): $\hat{C}_k$
- A running estimate of the expected number of blocked customers per cycle: $\hat{B}_k$

The two quantities are updated at every customer arrival until the simulation of the cycle ends. At this time, $\hat{C}_k$ is an estimate of the expected length of cycle k, where the expectation is taken with respect to the departure process, for a given sample path of the arrival process.

Simulation algorithm

For a given stationary policy $u \in \Psi_s$:

- Step 0: Set $k = 1$.
- Step 1: Simulate cycle k.
  Step 1.0: Initialization
  - Set $\hat{C}_k = 0$ and $\hat{B}_k = 0$.
  - Set $t = 0$ and $n = 0$.

  Step 1.1: Transition
  - Generate $X \sim F$.
  - Set $t := t + X$.
  - Given that the state is $(t, n, t_{n-l+1}, \ldots, t_n)$, compute the probability of no rejection so far:

$$P = 1 - p^l_{b+s+1}(t, n, t_{n-l+1}, \ldots, t_n).$$

  - Update the expected cycle length:

$$\hat{C}_k = \hat{C}_k + P.$$

  - Update the number of blocked customers and the state:
    If $u(t, n, t_{n-l+1}, \ldots, t_n) =$ "admit":

$$n = n + 1,$$

$$t_n = t.$$

    If $u(t, n, t_{n-l+1}, \ldots, t_n) =$ "block":

$$\hat{B}_k = \hat{B}_k + P.$$

  - If P $\geq \epsilon$, go back to Step 1.1, otherwise set $k := k + 1$ and go to Step 1.

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

- Step 2: Compute averages:

$$\bar{\hat{C}}(K,\epsilon) = \frac{1}{K}\sum_{k=1}^{K}\hat{C}_k,$$

$$\bar{\hat{B}}(K,\epsilon) = \frac{1}{K}\sum_{k=1}^{K}\hat{B}_k.$$

The algorithm requires the specification of K, the number of cycles generated, and ϵ, the small probability value that determines the end of a cycle.

LEMMA 1: *For a given control policy $u \in \Psi_s$, we have*

$$J^l(u) = \lim_{K\to\infty}\lim_{\epsilon\to 0}\frac{1 + c\bar{\hat{B}}(K,\epsilon)}{\bar{\hat{C}}(K,\epsilon)},$$

where $\bar{\hat{B}}(K,\epsilon)$ and $\bar{\hat{B}}(K,\epsilon)$ are obtained using the simulation algorithm.

PROOF: Let C_1 denote the length of the first cycle (i.e., the number of transitions until a rejection) given that the system was full at time 0. Let G_1 denote the total cost incurred in the first cycle:

$$G_1 = \sum_{k=1}^{C_1} g(i_k, u(i_k)).$$

Because of the assumption that the cycle length has finite expectation and given that per-stage costs are bounded, we get that $\mathbb{E}[G_1] < \infty$. Since the control is a stationary policy, total costs in each cycle are i.i.d. and, therefore, we have

$$J^l(u) = \frac{\mathbb{E}[G_1]}{\mathbb{E}[C_1]}.$$

Let C_k and G_k be the length and total cost in cycle k, respectively. Given that $\mathbb{E}[C_1] < \infty$ and $\mathbb{E}[G_1] < \infty$, the strong law of large numbers (see Chung [8, Thm. 5.4.2]) implies that

$$\lim_{K\to\infty}\frac{1}{K}\left[\sum_{k=1}^{K}C_k\right] = \mathbb{E}[C_1],$$

$$\lim_{K\to\infty}\frac{1}{K}\left[\sum_{k=1}^{K}G_k\right] = \mathbb{E}[G_1].$$

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

We now prove that

$$\lim_{K\to\infty}\lim_{\epsilon\to 0}\bar{\hat{C}}(K,\epsilon) = \mathbb{E}[C_1], \tag{9}$$

$$\lim_{K\to\infty}\lim_{\epsilon\to 0}[1 + c\bar{\hat{B}}(K,\epsilon)] = \mathbb{E}[G_1]. \tag{10}$$

For this, simulate one cycle with the simulation algorithm. Let ω denote the sample path of interarrival times generated in this cycle. Let $\Phi \subset \mathcal{S}^l$ be the set of states that are reached during the repetitions of Step 1.1 for this cycle, using policy u. Let $i_j \in \Phi$ denote the state reached in the jth transition (i.e., the jth repetition of Step 1.1). State i_j is visited if the system does not incur a rejection before the jth transition, which happens with probability $1 - p^l_{b+s+1}(i_j)$. It follows that the expected length of the cycle, given sample path ω, is greater than or equal to

$$\sum_{j:i_j\in\Phi}[1 - p^l_{b+s+1}(i_j)] = \hat{C}_1.$$

It follows that if $\epsilon \to 0$, the expected length of the cycle, given ω, tends to $\hat{C}_1$. Using the strong law of large numbers, we obtain (9).

The expected cost of blocking an arrival in state $i_j \in \Phi$ is given by

$$c[1 - p^l_{b+s+1}(i_j)],$$

which is the cost of blocking multiplied by the probability that i_j is visited.

The expected cost of accepting an arrival in state $i_j \in \Phi$ is equal to the probability that the system is full multiplied by the probability of visiting the state; that is,

$$\begin{aligned} f^l(i_j)[1 - p^l_{b+s+1}(i_j)] &= \frac{p^l_{b+s+1}(i_{j+1}) - p^l_{b+s+1}(i_j)}{1 - p^l_{b+s+1}(i_j)}[1 - p^l_{b+s+1}(i_j)] \\ &= p^l_{b+s+1}(i_{j+1}) - p^l_{b+s+1}(i_j). \end{aligned}$$

The expected total cost incurred in the first cycle, given a sample path ω, is greater than or equal to

$$\sum_{\substack{j:i_j\in\Phi,\\ u(i_j)=\text{“}block\text{”}}} c[1 - p^l_{b+s+1}(i_j)] + \sum_{\substack{j:(i_j)\in\Phi,\\ u(i_j)=\text{“}admit\text{”}}} [p^l_{b+s+1}(i_{j+1}) - p^l_{b+s+1}(i_j)]$$

$$= c\hat{B}_1 + 1 - \epsilon.$$

It follows that if $\epsilon \to 0$, the expected total cost incurred in the first cycle, given ω, tends to $1 + c\hat{B}_1$. Finally, using the strong law of large numbers, we obtain (10). ■

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

It follows that for sufficiently high K and small ϵ, we have

$$J^l(u) \approx \frac{1 + c\bar{\hat{B}}(K,\epsilon)}{\bar{\hat{C}}(K,\epsilon)};$$

that is, the average cost per incoming customer under policy u can be approximated using the simulation algorithm. Note that this simulation technique has a lower variance of the estimators $\bar{\hat{B}}(K,\epsilon)$ and $\bar{\hat{C}}(K,\epsilon)$ because we avoid introducing the additional variance due to the service process (see similar arguments in Ross and Seshadri [16] and Ross [15]).

7. DEFINING THE WEIGHTS TO OBTAIN A LOWER BOUND

Based on the proposed simulation method, we suggest choosing the weights $w_{(t,n,t_1,\ldots,t_n)}$ in (7) in the following way:

$$\bar{w}_{(t,n,t_1,\ldots,t_n)} = \begin{cases} 1 & \text{if } (t_1,\ldots,t_{n-l}) = (\bar{t}_1,\ldots,\bar{t}_{n-l}) \\ 0 & \text{otherwise,} \end{cases} \tag{11}$$

where

$$(t,n,\bar{t}_1,\ldots,\bar{t}_{n-l},t_{n-l+1},\ldots,t_n) = \underset{t_1,\ldots,t_{n-l}:(t,n,t_1,\ldots,t_n)\in\mathcal{S}}{\operatorname{argmin}} p_{b+s+1}(t,n,t_1,\ldots,t_{n-l},t_{n-l+1},\ldots,t_n) \tag{12}$$

and $\mathbf{p}$ is given by (2). In the event that more than one set of values achieves the minimum, pick one of them randomly.

It follows that the vector of unconditional probabilities associated with aggregate state $(t,n,t_{n-l+1},t_{n-l},\ldots,t_n)$ is the vector of unconditional probabilities associated with the full memory state $(t,n,\bar{t}_1,\ldots,\bar{t}_{n-l},t_{n-l+1},\ldots,t_n)$, where $\bar{t}_1,\ldots,\bar{t}_{n-l}$ are chosen such that they minimize the *unconditional* probability that the system has incurred a rejection since time 0. In other words, we have

$$\mathbf{p}^l(t,n,t_{n-l+1},\ldots,t_n) = \mathbf{p}(t,n,\bar{t}_1,\ldots,\bar{t}_{n-l},t_{n-l+1},\ldots,t_n),$$

and, therefore,

$$\begin{aligned} f^l(t,n,t_{n-l+1},\ldots,t_n) &= f(t,n,\bar{t}_1,\ldots,\bar{t}_{n-l},t_{n-l+1},\ldots,t_n) \\ &= r_{b+s}(t,n,\bar{t}_1,\ldots,\bar{t}_{n-l},t_{n-l+1},\ldots,t_n), \end{aligned}$$

with $\mathbf{r}$ defined in (4).

The reason for the choice of $(t,n,\bar{t}_1,\ldots\bar{t}_{n-l},t_{n-l+1},\ldots,t_n)$ is established in the following proposition.

PROPOSITION 1: *With w given by (11), we have* $\delta^{l+1} \geq \delta^l$ *for* $l > 0$.

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

PROOF: By (12), we have, for every $(t, n, t_1, \ldots, t_n) \in \mathcal{S}$,

$$p^{l+1}_{b+s+1}(t, n, t_{n-l}, \ldots, t_n) \geq p^{l}_{b+s+1}(t, n, t_{n-l+1}, \ldots, t_n). \tag{13}$$

Fix a policy u and use the simulation algorithm of the previous section to estimate $J^{l+1}(u)$ and $J^l(u)$. We use the subscripts $l+1$ and l for the simulation of $J^{l+1}(u)$ and $J^l(u)$, respectively.

Using the same sample path of arrivals for each simulation, we get from (13) that at any given transition of cycle k,

$$P^{l+1} \leq P^l. \tag{14}$$

It follows that

$$\hat{C}^{l+1}_k \leq \hat{C}^l_k,$$

and, therefore,

$$\bar{\hat{C}}^{l+1}(K, \epsilon) \leq \bar{\hat{C}}^l(K, \epsilon). \tag{15}$$

Now consider control policy u^{l+1*}, which is optimal when the memory size is $l+1$. Again, use the simulation algorithm of Section 6 in two different systems: one with memory size l and one with $l+1$. We refer to these two cases as systems l and $l+1$, respectively. Let t^{l+1}_ϵ be the time at which the simulation of system $l+1$ stops. By (14), the simulation of system l has not ended by then. After t^{l+1}_ϵ, start admitting all customers in system l. The two simulations result in the same number of blocked and rejected customers. However, (15) implies that the cycle length is longer for system l. By Lemma 1, this implies that the cost of this policy for system l is less than that for system $l+1$; that is,

$$J^l(u^{l+1*}) \leq J^{l+1}(u^{l+1*}).$$

Finally, by the optimality of u^{l*},

$$\delta^l = J^l(u^{l*}) \leq J^{l+1}(u^{l+1*}) = \delta^{l+1}.$$

■

By applying Proposition 1 to the case of full memory ($l = \infty$), we get the following corollary.

COROLLARY 1: *$\delta^l \leq \delta$ for $l < \infty$.*

We have obtained lower bounds on the optimum average cost δ, which do not require the computation of the optimal control u^*. The quality of the lower bound improves as l increases however, so does the complexity of the DP needed to obtain the control u^{*l}. The numerical results of Section 9 show that the policies obtained with weights (11) perform quite well also. The question of how to find the original state $(t, n, \bar{t}_1, \ldots, \bar{t}_{n-l}, t_{n-l+1}, \ldots, t_n)$ for each aggregate state $(t, n, t_{n-l+1}, \ldots, t_n)$ is considered in the following section.

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

8. DETERMINING THE STATE $(t, n, \bar{t}_1, \ldots, \bar{t}_{n-l}, t_{n-l+1}, \ldots, t_n)$

In this section we derive properties that can be used to determine $\bar{t}_1, \ldots, \bar{t}_{n-l}$ given $(t, n, t_{n-l+1}, \ldots, t_n)$ with $n > l$. We refer to $\bar{t}_1, \ldots, \bar{t}_{n-l}$ as the *variable* admission epochs (versus $t_{n-l+1}, \ldots, t_n$ that are fixed). Also, let $\bar{t}_k^-$ denote the instant just before the admission of the kth customer.

LEMMA 2: *$p_{b+s+1}(t, n, \bar{t}_1, \ldots, \bar{t}_{n-l}, t_{n-l+1}, \ldots, t_n)$ is convex in $\bar{t}_k$ for $k = 1, \ldots, n-l$.*

PROOF: Choose $1 \le k \le n-l$ (assume that $\bar{t}_{k-1} = 0$ if $k = 1$ and $\bar{t}_{k+1} = t_{n-l+1}$ if $k = n-l$). From (2), we get that the probability that there is a rejection before time t can be written as

$$\mathbf{p}_{b+s+1}(t, n, \bar{t}_1, \ldots, \bar{t}_{n-l}, t_{n-l+1}, \ldots, t_n) = \tilde{\mathbf{p}} M(\bar{t}_k - \bar{t}_{k-1}) Z M(\bar{t}_{k+1} - \bar{t}_k) Y_{\cdot, b+s+1}, \tag{16}$$

where

$$\tilde{\mathbf{p}} = e^{b+s+1} M(\bar{t}_1) Z \ldots M(\bar{t}_{k-1} - \bar{t}_{k-2}) Z,$$

$$\mathbf{Y} = Z M(\bar{t}_{k+2} - \bar{t}_{k+1}) \ldots Z M(t - t_n).$$

In words, $\tilde{\mathbf{p}}$ is the vector of unconditional probabilities vector estimated at time $\bar{t}_{k-1}$ and $Y_{y, b+s+1}$ with $y \le b+s$ is the probability that the system experiences a rejection in the interval $[\bar{t}_{k+1}, t]$, given there are y in the system at time $\bar{t}_{k+1}^-$. From the definition of $Y_{y, b+s+1}$, we get

$$Y_{0, b+s+1} \le Y_{1, b+s+1} \le \cdots \le Y_{b+s, b+s+1} = 1 = Y_{b+s+1, b+s+1}. \tag{17}$$

Let x denote the number of customers that are in the system at time $\bar{t}_{k-1}$; we refer to them as *original* customers. Let $i \le x$ denote the number of original customers that are still in the system at time $\bar{t}_{k+1}$. We consider six cases. For each one, we show that the probability of rejection before time t is either independent of $\bar{t}_k$ or is a convex function of $\bar{t}_k$. Let customer k be the customer that is admitted at time $\bar{t}_k$.

Case 1: $x < s$. The probability of a rejection before t is given by

$$\tilde{\mathbf{p}}_x M_{x,i}(\bar{t}_{k+1} - \bar{t}_{k-1}) [e^{-\mu(\bar{t}_{k+1} - \bar{t}_k)} Y_{i+1, b+s+1} + (1 - e^{-\mu(\bar{t}_{k+1} - \bar{t}_k)}) Y_{i, b+s+1}].$$

In this expression, $\tilde{p}_x$ is the probability that there are x customers in the system at time $\bar{t}_{k-1}$ and $M_{x,i}(\bar{t}_{k+1} - \bar{t}_{k-1})$ is the probability that out of these x customers, i are still in the system at time $\bar{t}_{k+1}$. Because the number of original customers is less than the number of servers, there is at least one free server at time $\bar{t}_k$ so that customer k starts service right away. With probability $e^{-\mu(\bar{t}_{k+1} - \bar{t}_{k-1})}$, customer k is still being served at time $\bar{t}_{k+1}^-$ and there is a total of $i+1$ customers in the system at that time, so that the probability of a rejection after $\bar{t}_{k+1}^-$ is $Y_{i+1, b+s+1}$. With probability $(1 - e^{-\mu(\bar{t}_{k+1} - \bar{t}_{k-1})})$, customer k has left the system and there are i customers at $\bar{t}_{k+1}^-$,

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

so that the probability of a rejection after $\bar{t}_{k+1}^-$ is $Y_{i,b+s+1}$. By (17), this expression is convex in $\bar{t}_k$.

Case 2: $s \le x < b + s$ and $s \le i \le x$. At time $\bar{t}_{k+1}^-$, some original customers are still in the queue, waiting to be served. It follows that customer k does not start service before time $\bar{t}_{k+1}$ and, hence, there are $i + 1$ customers in the system at time $\bar{t}_{k+1}$. By a logic similar to that in Case 1, the probability of a rejection before t is given by

$$\tilde{\mathbf{p}}_x M_{x,i}(\bar{t}_{k+1} - \bar{t}_{k-1}) Y_{i+1,b+s+1},$$

which does not depend on $\bar{t}_k$.

Case 3: $s \le x < b + s$ and $i < s$. At time $\bar{t}_{k+1}^-$, there is at least one free server. Therefore, customer k starts service in the interval $[\bar{t}_k, \bar{t}_{k+1})$. Let $\tau \in [\bar{t}_{k-1}, \bar{t}_{k+1})$ denote the time when the number of original customers becomes $s - 1$. Customer k starts service at time $\max\{\bar{t}_k, \tau\} \equiv \bar{t}_k \vee \tau$. The probability of a rejection before t in Case 3 is given by

$$\tilde{\mathbf{p}}_x M_{x,s-1}(\tau - \bar{t}_{k-1}) M_{s-1,i}(\bar{t}_{k+1} - \tau) \times [e^{-\mu(\bar{t}_{k+1} - (\bar{t}_k \vee \tau))} Y_{i+1,b+s+1} + (1 - e^{-\mu(\bar{t}_{k+1} - (\bar{t}_k \vee \tau))}) Y_{i,b+s+1}].$$

This is a convex function of $\bar{t}_k$.

Case 4: $x = b + s$ and $i = b + s$. The system is full at time $\bar{t}_{k-1}$ and all of the original customers are still in the system at time $\bar{t}_{k+1}^-$. Therefore, customer k is rejected. The probability of a rejection before t is given by

$$\tilde{\mathbf{p}}_{b+s} M_{b+s,b+s}(\bar{t}_{k+1} - \bar{t}_{k-1}),$$

which does not depend on $\bar{t}_k$.

Case 5: $x = b + s$ and $s \le i < b + s$. The system is full at time $\bar{t}_{k-1}$ and some original customers are still in the queue at time $\bar{t}_{k+1}^-$. If at least one original customer leaves the system before time $\bar{t}_k$, then this case is equivalent to Case 2. If none of the original customers leaves the system by time $\bar{t}_k$, which happens with probability $M_{b+s,b+s}(\bar{t}_k - \bar{t}_{k-1}) M_{b+s,i}(\bar{t}_{k+1} - \bar{t}_k)$, then customer k is rejected. The probability of a rejection before t is the sum of these:

$$\tilde{\mathbf{p}}_{b+s} M_{b+s,i}(\bar{t}_{k+1} - \bar{t}_{k-1}) Y_{i+1,b+s+1} + \tilde{\mathbf{p}}_{b+s} M_{b+s,b+s}(\bar{t}_k - \bar{t}_{k-1}) \times M_{b+s,i}(\bar{t}_{k+1} - \bar{t}_k)[1 - Y_{i+1,b+s+1}].$$

It can be shown using (3) that this is a convex function of $\bar{t}_k$.

Case 6: $x = b + s$ and $i < s$. The system is full at time $\bar{t}_{k-1}$ and there is at least one free server at time $\bar{t}_{k+1}^-$. Again, let $\tau \in [\bar{t}_{k-1}, \bar{t}_{k+1})$ denote the time when the number of original customers becomes $s - 1$. If at least one original customers left before time $\bar{t}_k$, then this case is similar to Case 3. However, if no original

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

customer leaves the system before time $\bar{t}_k$, which happens with probability $M_{b+s,b+s}(\bar{t}_k - \bar{t}_{k-1})M_{b+s,s-1}(\tau - \bar{t}_k)M_{s-1,i}(\bar{t}_{k+1} - \tau)$, then customer k is rejected. The probability of a rejection before t is the sum of these:

$$\begin{aligned}
&\tilde{\mathbf{p}}_{b+s}M_{b+s,s-1}(\tau - \bar{t}_{k-1})M_{s-1,i}(\bar{t}_{k+1} - \tau)\\
&\quad\times [e^{-\mu(\bar{t}_{k+1}-(\bar{t}_k\vee\tau))}Y_{i+1,b+s+1} + (1 - e^{-\mu(\bar{t}_{k+1}-(\bar{t}_k\vee\tau))})Y_{i,b+s+1}]\\
&\quad+ \tilde{\mathbf{p}}_{b+s}M_{b+s,b+s}(\bar{t}_k - \bar{t}_{k-1})M_{b+s,s-1}(\tau - \bar{t}_k)M_{s-1,i}(\bar{t}_{k+1} - \tau)\\
&\quad\times [e^{-\mu(\bar{t}_{k+1}-\tau)}(1 - Y_{i+1,b+s+1}) + (1 - e^{-\mu(\bar{t}_{k+1}-\tau)})(1 - Y_{i,b+s+1})].
\end{aligned}$$

It can be shown using (3) that this is a convex function of $\bar{t}_k$. ■

It follows from Lemma 2 that a local optimum in the search for the minimum value of $p_{b+s}(t, n, t_1, \ldots, t_n)$ can be found by a line search. We also observe from the proof that the optimal value of $\bar{t}_k$ is the result of a trade-off between the risk of losing that customer because the system was full at the time he was admitted and the benefit of serving that customer as soon as possible in order to free the servers for future admissions.

The following corollary establishes that it is optimal to set the most recent $b - l$ variable admission epochs (i.e., $\bar{t}_{(n-b)\wedge 1}, \ldots, \bar{t}_{n-l}$) equal to the earliest admission epoch the controller remembers (i.e., t_{n-l+1}).

COROLLARY 2: *The state* $(t, n, \bar{t}_1, \ldots, \bar{t}_{n-l}, t_{n-l+1}, t_n)$ *is such that for* $n \geq 2$,

$$\bar{t}_{(n-b)\wedge 1} = \cdots = \bar{t}_{n-l} = t_{n-l+1}.$$

PROOF: The proof is by induction. Let $(n - b) \wedge 1 \leq k \leq n - l$ and assume that $\bar{t}_j = t_{n-l+1}$ for $j = k + 1, \ldots, n - l$; that is, $n - l - k + 1$ customers are admitted at time $t_{n-l+1} = \bar{t}_{k+1}$. After that, $l - 1$ customers are admitted at times $t_{n-l+2}, \ldots, t_n$. The probability of a rejection before time t is given by (16). In this expression, $Y_{y,b+s+1}$ is the probability that the system incurs a rejection in the interval $[\bar{t}_{k+1}, t]$, given that there are y in the system at time $\bar{t}_{k+1}^-$. We claim that

$$Y_{b+s-(n-l-k),b+s+1} = \cdots = Y_{b+s,b+s+1} = 1, \tag{18}$$

$$0 < Y_{b+s-(n-k)+1,b+s+1} \leq \cdots \leq Y_{b+s-(n-l-k+1),b+s+1} \leq 1, \tag{19}$$

$$Y_{0,b+s+1} = \cdots = Y_{b+s-(n-k),b+s+1} = 0. \tag{20}$$

Since there are $n - l - k + 1$ admissions at time $\bar{t}_{k+1}$, there is a rejection at this time if and only if there are more than $b + s - (n - l - k + 1)$ customers at time $\bar{t}_{k+1}^-$, which gives (18). The $l - 1$ subsequent customers are rejected if they find the system full at the time of their arrival, which can only happen if there are more than $b + s - (l - 1) - (n - l - k + 1) = b + s - (n - k)$ at time $\bar{t}_{k+1}^-$; therefore, we get (19) and (20).

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

As shown in the proof of Lemma 2, the probability of a rejection before time t can be analyzed in six cases. In Cases 2 and 4, the probability does not depend on $\bar{t}_k$. We can eliminate Cases 1 and 3 and the first term of Case 6, as $i < s \le b + s - (n - k)$ and (20) implies $Y_{i,b+s+1} = Y_{i+1,b+s+1} = 0$ in these expressions. However, the second term of Case 6 is a function of $\bar{t}_k$. Summing up all of those terms for $i = 0, \ldots, s-1$, we get

$$\tilde{\mathbf{p}}_{b+s} M_{b+s,b+s}(\bar{t}_k - \bar{t}_{k-1}) \sum_{i=0}^{s-1} M_{b+s,i}(\bar{t}_{k+1} - \bar{t}_k).$$

In Case 5, the first term does not depend on $\bar{t}_k$. As for the second term, it is positive only if $Y_{i+1,b+s+1} < 1$, which from (19) is true for $s \le i \le b + s - (n - k - l) - 2$. Therefore, only the second term depends on $\bar{t}_k$ if $s \le i \le b + s - (n - k - l) - 2$. Summing up all of those terms, we get

$$\tilde{\mathbf{p}}_{b+s} M_{b+s,b+s}(\bar{t}_k - \bar{t}_{k-1}) \sum_{i=s}^{b+s-(n-l-k+2)} M_{b+s,i}(\bar{t}_{k+1} - \bar{t}_k)[1 - Y_{i+1,b+s+1}].$$

Adding up the two expressions, we get

$$\tilde{\mathbf{p}}_{b+s} M_{b+s,b+s}(\bar{t}_k - \bar{t}_{k-1}) \sum_{i=0}^{b+s-(n-l-k+2)} M_{b+s,i}(\bar{t}_{k+1} - \bar{t}_k)[1 - Y_{i+1,b+s+1}].$$

The first term is the probability that the system remains full in the interval $[\bar{t}_{k-1}, \bar{t}_k]$. This probability decreases as $\bar{t}_k$ increases since more customers have time to finish service. The second term is the probability that there is no rejection in the interval $[\bar{t}_{k+1}, t]$, given that the system has $b + s$ customers at time $\bar{t}_k$. The second term can be made equal to zero when $\bar{t}_k = \bar{t}_{k+1}$ and, therefore, it is optimal to do so. ■

The intuition behind this result is the following. There can be a rejection either at time $\bar{t}_k$ or in the interval $[\bar{t}_{k+1}, t]$. The former happens if the system is full at time $\bar{t}_k$ and the probability of this happening decreases with $\bar{t}_k$. The latter happens if there are more than $b + s - (n - k)$ customers at $\bar{t}_{k+1}^-$, which is more than the number of servers because $k \ge n - b$. The probability of this happening is independent of $\bar{t}_k$ because customer k does not start service in this case and therefore has no chance of leaving the system before time $\bar{t}_{k+1}$. It follows that it is optimal to set $\bar{t}_k = \bar{t}_{k-1}$ to minimize the probability of a rejection at time $\bar{t}_k$.

There is a remarkable special case in which we can establish a symmetry property of the variable admission epochs. The result is given in the following lemma.

LEMMA 3: *When $s = 1$, the state $(t, n, \bar{t}_1, \ldots, \bar{t}_{n-1}, t_n)$ corresponding to aggregate state (t, n, t_n) is such that for $n \ge b + 1$,*

$$\bar{t}_1 = t_n - \bar{t}_{n-b-1},$$

$$\bar{t}_k - \bar{t}_{k-1} = \bar{t}_{n-b-k+1} - \bar{t}_{n-b-k}, \qquad k = 2, \ldots, \left\lfloor \frac{n-b-1}{2} \right\rfloor. \tag{21}$$

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

PROOF: Let us assume for ease of presentation that $n - b > 1$ (the proof when $n - b = 1$ is similar). By Corollary 2, we have $\bar{t}_{n-b} = \cdots = t_n$; that is, $(b + 1)$ customers are accepted at time t_n. This will result in a rejection unless the system is empty at time t_n^-. So we wish to determine $\bar{t}_1, \ldots, \bar{t}_{n-b-1}$ so as to maximize the probability that the system is empty at time t_n^-. Let $t_0 = 0$ denote the epoch of the last rejection.

Let s_k denote the number of departures in the interval $(t_k, t_{k+1}]$ for $k = 0, \ldots, n - b - 1$. Given that the buffer is full at time 0, the number of departures should satisfy the following conditions in order to have $(b + 1)$ customers at time t_n and no rejection in the interval $[0, t_n]$:

$$\sum_{i=0}^{k} s_i \geq k + 1, \qquad k = 0, \ldots, n - b - 1, \tag{22}$$

$$\sum_{i=0}^{k} s_{n-b-1-i} \geq k + 1, \qquad k = 0, \ldots, n - b - 1, \tag{23}$$

$$\sum_{i=0}^{n-b-1} s_i = n. \tag{24}$$

The first condition guarantees that the system is not full at each admission epoch. The second condition makes sure that the customers that have been admitted after time 0 have left the system before time t_n. The third condition states that all of the customers that were ever in the system in the time interval $[0, t_n^-]$ should have left the system before time t_n^-. This includes the $b + 1$ customers that were in the system at time 0 plus the $n - b - 1$ customers who arrived in $(0, t_n)$.

Now, suppose that we let time run backward so that we start with a full system at time t_n, and have customers come at time $\bar{t}_{n-b-1}, \ldots, \bar{t}_1$. Denote this as the reversed system. Due to the symmetry of (22) and (23), the same conditions would apply for the reversed system to be empty at time 0 without any rejection in the "interval" $[t_n, 0]$.

It follows that every realization of the forward system in which there is no loss corresponds to a realization of the reversed system in which there is no loss. Similarly, every realization of the reversed system in which there is no loss corresponds to a realization of the forward system where there is no loss. Thus, the sample paths over which there is no loss are in 1–1 correspondence in the two systems.

If the epochs $\bar{t}_1, \ldots, \bar{t}_{n-b-1}$ maximize the probability that the system is empty in the forward system, then the reversed epochs maximize the same probability in the reversed system. Therefore, we conclude that the arrival epochs $\bar{t}_1, \ldots, \bar{t}_{n-b-1}$ must be symmetric in $[0, t_n]$. ■

9. NUMERICAL RESULTS

In this section we compute the optimal average cost, the lower bounds, and the performance of the policies on the aggregate state space. We use the MDP in (5) to

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

Table 1. Parameters Used in Experiments

Scenario	s	b	μ	ρ
1	1	1	0.5	1.18
2	1	1	0.2	2.96
3	3	3	0.1	1.97

obtain the optimal policy and we use the MDP in (8) with weights defined by (11) to obtain the policies on the aggregate state spaces S^l with $l = 1,2$. Average costs are estimated using the simulation algorithm described in Section 6.

We choose the following discretizing parameters: $m = 50, \Delta = 1, T = 75$. Also, we set $\lambda = 0.9$ and take interarrival time X to be distributed according to

$$P[X = k] = e^{-(k-1)\lambda} - e^{-k\lambda}, \qquad k = 1,2,\ldots,$$

such that the mean interarrival time is $(1 - e^{-\lambda})^{-1}$.

Table 1 shows the different sets of parameter values we use in each scenario, including $\rho = (1 - e^{-\lambda})(s\mu)^{-1}$, the load offered to the system.

For each set of parameters, we vary the cost of blocking c in steps of 0.05 for a range of values within which it is optimal to block and also to admit some customers.

Figure 1 shows that computing the bound with $l = 2$ (i.e., δ^2) significantly improves the bound obtained with $l = 1$ (i.e., δ^1). δ^1 is, on average, 11.14% lower

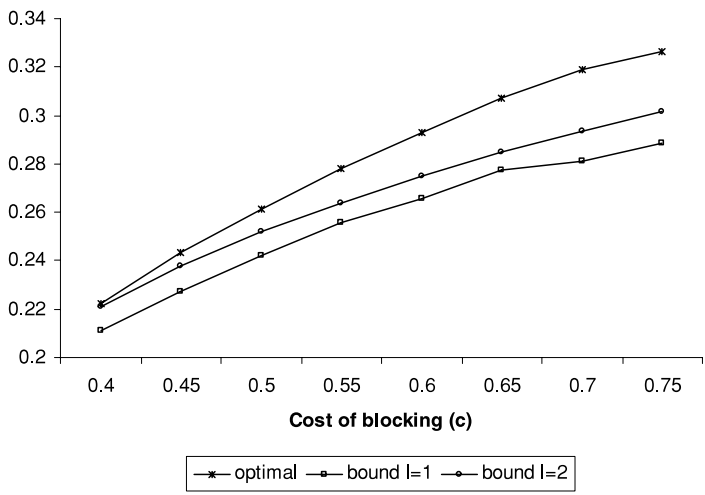

Figure 1. Scenario 1: bounds.

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

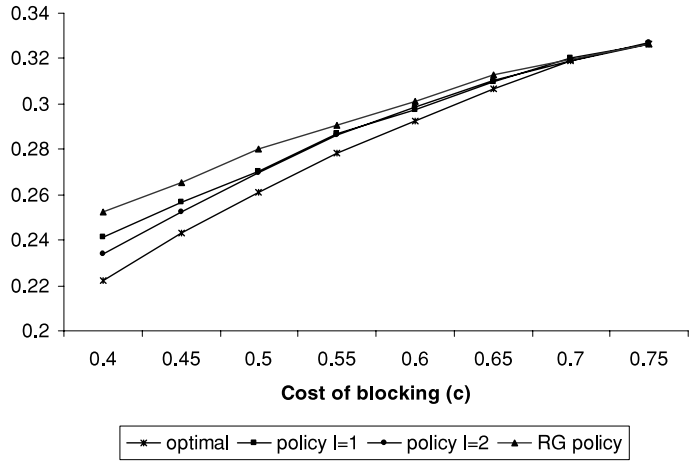

FIGURE 2. Scenario 1: performance of policies.

than the optimal average cost, whereas δ^2 is, on average, 8.03% lower. The quality of the two bounds increases as c decreases. In contrast, Figure 2 shows that the policies u^{1*} and u^{2*} improve as c increases. They both perform better than the Reject–Gapping policy of Lin and Ross (depicted as RG) with an average optimality gap of 2.30% and 1.72%, respectively, for $l = 1$ and $l = 2$ versus 3.15% for the Reject–Gapping policy.

Figure 3 shows that the performance of the policy with $l = 1$ and the quality of the bound improve as the offered load increases. The bound is, on average, 1.1%

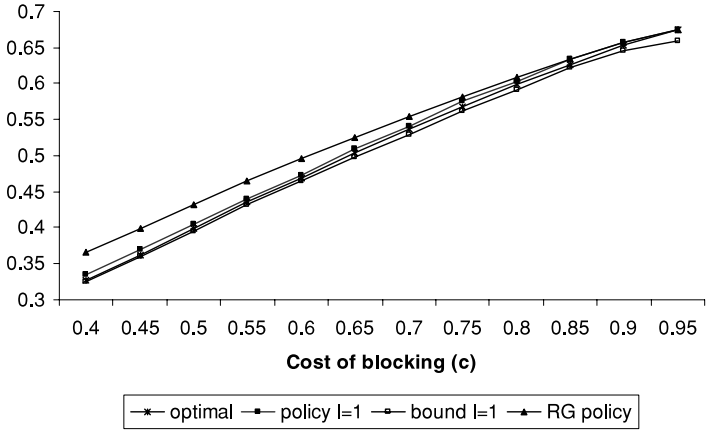

FIGURE 3. Scenario 2: performance and bound.

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

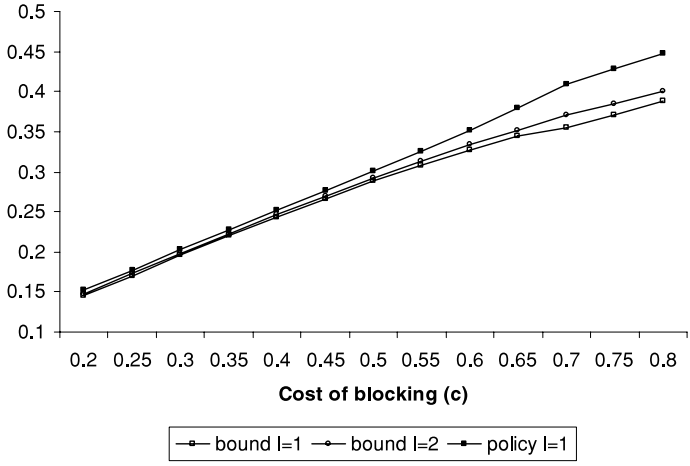

FIGURE 4. Scenario 3: bounds.

lower than the optimal policy and the policy u^{1*} performs, on average, only 1.01% worse than the optimal policy, as opposed to 4.66% for the Reject–Gapping policy.

In scenario 3, we cannot obtain the optimal policy because of the large state space, however, we can compute δ^1 and δ^2. Figure 4 shows that the difference between these bounds increases with c and is, on average, equal to 2.01%. Figure 5 shows that the performance of the policies are very similar. However, our policies tend to perform better than the RG policy for small values of c. The average per-

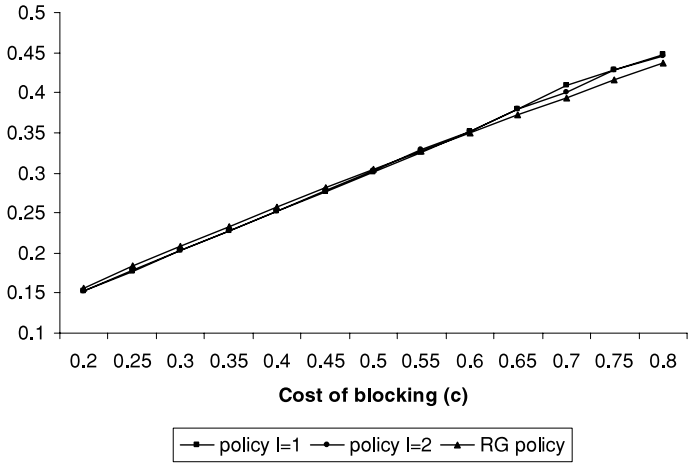

FIGURE 5. Scenario 3: performance of policies.

https://doi.org/10.1017/S0269964807070027 Published online by Cambridge University Press

centage difference between the cost given by our policy, $J(u^{2*})$, and δ^2 is 5.03%, so that even without knowing the optimal policy, we have an estimate of the optimality gap.

A good feature of our lower bound is that is is tight when the cost of blocking is low, because, in this case, the optimality gap is likely to be larger given that it is optimal for the controller to frequently block arriving customers. In contrast, when c is high, all policies tend to converge quickly to one that admits most customers and, therefore, the case when c is high is not as interesting.

10. CONCLUSION

We have provided a method for finding a control policy based on aggregation of the state space by discarding some information about the admissions to the system. By remembering only a finite number l of the most recent admission epochs along with the time and number of customers admitted since the last rejection, the controller can construct a control policy on a state space that does not grow with either the size of the buffer or the number of servers.

For each aggregate state, we set the unconditional probability of a rejection equal to the minimum of the rejection probability over all of the full memory states with the same values of the l most recent admission epochs. The full memory state that realizes this minimum is given a weight of one, whereas every other state is given a weight of zero. Our approach uses knowledge about the system to define and efficiently identify this state. It is interesting to note that the weights we obtain differ from the conditional probabilities induced by the control. The conventional approach of state aggregation uses these conditional probabilities but has no guarantee of convergence. In contrast, our method for defining the weights allows us to obtain a lower bound on the performance of the optimal policy. This is particularly useful when the buffer size and the number of servers are large and computing the optimal policy is very computationally intensive. We show numerically that the bound is tight in the important case when the cost of blocking is low and the load offered to the system is high. Moreover, the policies that we obtain perform generally better than the Reject–Gapping policy of Lin and Ross [11] and their performance naturally improves with the number of admission epochs the controller remembers. Finally, they perform best when the cost of blocking is high and the offered load increases. An opportunity for future research is to determine a technique for policy improvement that at the same time maintains the lower bound.

Acknowledgments

The authors wish to thank participants in the seminar at New York University and the 2004 Informs meetings in Denver for their comments and suggestions.

References

1. Altman, E. & Basar, T. (1995). Optimal rate control for high speed telecommunication networks: The case of delayed information. In *First Workshop on ATM Traffic Management*, *WATM'95*, *IFIP*, *WG.6.2 Broadband Communication*, Paris.

2. Altman, E., Kofman, D., & Yechiali, U. (1995). Discrete time queues with delayed information. *Queueing Systems* 19: 361–376.
3. Altman, E. & Koole, G. (1995). Control of a random walk with noisy delayed information. *Systems and Control Letters* 24: 207–213.
4. Altman, E., Marquez, R., & Yechiali, U. (2003). Admission and routing control with partial information and limited buffers. *International Journal of Systems Science* 34(10–11): 615–626.
5. Altman, E., & Nain, P. (1995). Closed-loop control with delayed information. In *Proceedings of the ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems*, pp. 193–204.
6. Altman, E. & Stidham, S. (1995). Optimality of monotonic policies for two-action Markovian decision processes, with applications to control of queues with delayed information. *QUESTA* 21: 267–291.
7. Bertsekas, D. (1995). *Dynamic programming and optimal control*, Vol. 2. Belmont, MA: Athena Scientific.
8. Chung, K. (1974). *A course in probability theory*, 2nd ed. New York: Academic Press.
9. Kuri, J. & Kumar, A. (1995). Optimal control of arrivals to queues with delayed queue length information. *IEEE Transactions on Automatic Control* 40(8): 1444–1450.
10. Kuri, J. & Kumar, A. (1997). On the optimal control of arrivals to a single queue with arbitrary feedback delay. *Queueing Systems* 27(1–2): 1–16.
11. Lin, K. & Ross, S. (2003). Admission control with incomplete information of a queueing system. *Operations Research* 51(4): 645–654.
12. Lin, K. & Ross, S. (2004). Optimal admission control for a single-server loss queue. *Journal of Applied Probability* 41: 535–546.
13. Litvak, N. & Yiechiali, U. (2003). Routing in queues with delayed information. *Queueing Systems* 43: 147–165.
14. Ross, S. (1970). *Applied probability models with optimization applications*. New York: Dover.
15. Ross, S. (1997). *Simulation*, 2nd ed. New York: Academic Press.
16. Ross, S. & Seshadri, S. (2002). Hitting times in an Erlang loss system. *Probability in the Engineering and Informational Sciences* 16(2): 167–184.
17. Schäl, M. (1993). Average optimality in dynamic programming with general state space. *Mathematics of Operations Research* 18(1): 163–172.