Adv. Appl. Prob. **37,** 915–937 (2005)
Printed in Northern Ireland

DYNAMIC ADMISSION CONTROL FOR LOSS SYSTEMS WITH BATCH ARRIVALS

E. L. ÖRMECİ,* *Koç University*
A. BURNETAS,** *University of Athens*

Abstract

We consider the problem of dynamic admission control in a Markovian loss system with two classes. Jobs arrive at the system in batches; each admitted job requires different service rates and brings different revenues depending on its class. We introduce the definition of a 'preferred class' for systems receiving mixed and single-class batches separately, and derive sufficient conditions for each system to have a preferred class. We also establish a monotonicity property of the optimal value functions, which reduces the number of possibly optimal actions.

Keywords: Admission control; loss system; batch arrival

2000 Mathematics Subject Classification: Primary 93E20
Secondary 90B99

1. Introduction

In this paper, we consider a two-class loss system with c identical parallel servers and no waiting room. Arrivals occur according to a Poisson process with rate λ. Each arrival consists of a random batch of jobs from one or both classes. Specifically, the probability that an arriving batch consists of j_1 jobs of class 1 and j_2 jobs of class 2 is equal to $p_{j_1 j_2}$. Whenever a class-i job is admitted to the system, it brings a reward of $r_i > 0$ upon its arrival and requires a service time exponentially distributed with rate μ_i. We are interested in dynamic admission policies that maximize both the total expected discounted reward, with a continuous discount rate β over an infinite horizon, and the long-run average net profit. The system may employ batch acceptance, meaning that it can either accept or reject the entire batch, or partial acceptance, meaning that some of the jobs in a batch can be admitted and the remaining ones rejected. For the case of batch acceptance, Örmeci and Burnetas [13] considered a system with equal service rates and equal rewards for all jobs, so that the only difference between the jobs is the size of the batches to which they belong. They showed that this system does not possess any monotonicity property. Hence, our paper concentrates only on systems following a partial acceptance policy.

Systems of this kind may arise in various applications, such as production systems and rental businesses. Control of the workload in a production system via dynamic policies of admission has received increasing attention in recent years. The types of item produced may differ in their service requirements and the profit they generate. Although the system owner may charge

Received 19 January 2004; revision received 23 November 2004.
* Postal address: Department of Industrial Engineering, Koç University, Rumeli Feneri Yolu, 34450 Sarıyer, İstanbul, Turkey. Email address: lormeci@ku.edu.tr
** Postal address: Department of Mathematics, University of Athens, Panepistimiopolis, 15784 Athens, Greece. Email address: aburnetas@math.uoa.gr
Supported by the Greek Ministry of Education (30%) and the European Union (70%) under the research program 'EPEAEK II: Pythagoras'.

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

a fixed admission fee for jobs of a specific type, the actual cost of serving each job may vary due to the changing use of the system resources. In such situations, the system can benefit from basing its acceptance decisions on the current state of the system. This calls for dynamic rather than static admission control rules. Moreover, in most production systems, orders arrive in batches and the system owner generally has the option of accepting some of the jobs in a batch while rejecting the rest. All these features are represented in the system described above.

Our model can also be applied to revenue management problems in rental businesses. Capacity control in revenue management addresses the optimal allocation of a fixed amount of resources to different demand segments. The underlying assumption is the perishability of the resources at a certain time. The effect of revenue management on the performance of car rental firms was discussed by Carrol and Grimes [4] and Geraghty and Johnson [5]. However, for many rental businesses, in particular car rental, the perishability assumption is not appropriate, since the resources are rented for a random amount of time, after which they are available for future customers. Savin *et al.* [21] were the first to formulate such systems as multiple-server loss models with uncertain customer arrivals and uncertain rental durations. Geraghty and Johnson [5] indicated that the major car rental companies depend largely on corporate customers, who demand more than one resource at a time. The system under consideration models the uncertain demand size generated by these customers, using random batches.

We first develop sets of sufficient conditions which ensure that a job class is 'preferred', in the sense that its jobs are always accepted whenever there are free servers, regardless of the system's congestion level. We also show that the characterization of a preferred class when all batches consist of jobs from the same class is analogous to that in a system with single-job arrivals. In both cases, accepting a job (either single or from the batch) depends on the trade-off between the burden this job will bring to the system if accepted and the profit lost if rejected. On the other hand, when a batch includes jobs from both classes, in order for a job of one class to be accepted, its net benefit must not only be positive, but also higher than the benefit of a job from the other class. Therefore, the characterization of preferred classes in this case is more complicated. We note that the conditions developed in this paper are sufficient but not necessary, since there are combinations of parameter values for which they do not ensure the existence of a preferred class. We analyze such cases numerically and make several observations and conjectures.

The existence of a preferred class constitutes a static property of the optimal admission policy, in the sense that if class i is preferred, jobs of this class are accepted whenever there is an available server, regardless of the current state of the system. However, the optimal policy is generally state dependent. Regarding this general issue, we prove the submodularity of the value function, a desirable property that guarantees the existence of optimal thresholds in single-arrival loss systems (see [2]). However, in systems with batch arrivals, submodularity can characterize the structure of the optimal policy only partially, although its proof is mathematically challenging and interesting due to the boundary effects, multiple servers, and batch arrivals.

Markov decision processes are widely used in the control of queueing systems: Altman [1] surveyed the theoretical tools developed to model and solve such problems, as well as their application areas in communications networks. In particular, admission control problems in stochastic knapsacks have been studied by many authors; see Chapter 4 of [19] for a comprehensive review. A stochastic knapsack is defined as a system consisting of c identical parallel servers, no waiting room, and K job classes. Each class is distinguished by its size b_i, its arrival rate, and its mean service time. If a class-i job is accepted into the system, it siezes b_i servers and occupies all of them till the end of its service time, at which point it releases all

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

b_i servers simultaneously. This is significantly different from the system with batch arrivals, for various reasons. In a system with batch arrivals, each admitted job of an arriving batch behaves independently, i.e. each job has its own service time and occupies and releases only one server. Moreover, these systems, unlike stochastic knapsacks, can receive mixed batches. For further comparison of stochastic knapsacks and systems with batch arrivals, we refer the reader to [13], which includes examples in which the structures of the corresponding optimal policies are compared.

There have been earlier investigations of the structural properties of optimal admission policies for certain stochastic knapsacks. Miller [11] has shown that there exists an optimal trunk reservation policy for a stochastic knapsack with K different job classes, each of which has the same mean service time and a unit size, meaning that $b_i = 1$. Lippman and Ross [8] analyzed the optimal admission rule for a system with one server and no waiting room that receives offers from jobs according to a joint service time and reward probability distribution. They also considered such a system when it receives batch arrivals. Several authors have studied the structural properties of optimal dynamic admission policies in Markovian stochastic knapsacks with two classes of job. Altman *et al.* [2] showed that these policies are of threshold type, whereas Örmeci *et al.* [15] established the monotonicity of thresholds under certain conditions while assuming that $b_i = 1$. Moreover, Örmeci *et al.* [15] and Savin *et al.* [21] analyzed the issue of preferred classes in these systems. Örmeci *et al.* [14] addressed the issues of optimal thresholds and preferred jobs when the rewards are random. Carrizosa *et al.* [3] analyzed an optimal static control policy for a stochastic knapsack with K classes of job, where the service times follow a general distribution. A loss system with batch arrivals was considered by Puhalskii and Reiman [16], who restricted the domain of admission policies to the set of trunk reservation policies. However, optimal policies are not necessarily trunk reservation policies, as illustrated in Section 4. Örmeci and Burnetas [13] also considered the structure of optimal policies in loss systems with batch arrivals, concentrating on systems with K classes of jobs that have different rewards but the same service rate.

The paper is organized as follows. In the next section, we present the Markov decision process model of the system described above. In Section 3, we present the conditions under which a preferred class exists for systems with single-class and mixed batches. In Section 4, we prove the submodularity of the optimal value functions. We conclude and point out possible extensions of this work in Section 5. The paper closes with a technical appendix.

2. Model description

2.1. Discrete-time equivalent

In this section, we construct a discrete-time Markov decision process for systems employing a partial acceptance policy, with the objective of maximizing the total expected discounted returns over a finite time horizon with β as the discount rate. We first define the state as $\boldsymbol{x} = (x_1, x_2)$, where x_i is the number of class-i jobs. We interpret discounting as exponential failures, i.e. the system closes down in an exponentially distributed time with rate β. (For the equivalence of the process with discounting and the process without discounting, but with an exponential deadline, see, e.g. [22].) We also assume, without loss of generality, that $\mu_1 \leq \mu_2$. Arrivals occur according to a Poisson process with rate λ. The maximum possible rate out of any state is then $\lambda + c\mu_2 + \beta$. Since the maximum rate of transitions is finite, we can use uniformization (introduced by Lippman [7]) and normalization to construct a discrete-time equivalent of the original system. Specifically, we assume that the time between two successive transitions is exponentially distributed with rate $\lambda + c\mu_2 + \beta$, and, using the appropriate time scale, that $\lambda + c\mu_2 + \beta = 1$. At each transition epoch, there is an arrival of a batch of jobs with probability λ; a service completion of a class-i job with probability $x_i\mu_i$; a fictitious service completion with probability $c\mu_2 - x_1\mu_1 - x_2\mu_2$, due to uniformization; or a transition to the terminal state with probability β, due to discounting.

Furthermore, we will refer to the instantaneous states at the arrival epochs as $(\boldsymbol{x}; \boldsymbol{j}) = (x_1, x_2; j_1, j_2)$, indicating that a batch of (j_1, j_2) jobs has arrived at the system to find x_i, $i = 1, 2$, class-i jobs already present. Note that admission and rejection decisions are assumed to be made upon an arrival, meaning that these states are observed only at arrival epochs. Immediately after those epochs, the system moves to another state according to the decision made.

A batch, (j_1, j_2), consisting of two classes, i.e. $j_1 > 0$ and $j_2 > 0$, is called a mixed batch, whereas batches with $j_1 = 0$ or $j_2 = 0$ contain only one class of job and are referred to as single-class batches. We will consider two kinds of system: systems with mixed batches receive at least one mixed batch with a positive probability, and systems with single-class batches never receive mixed batches.

2.2. Markov decision model for a finite horizon

We denote the maximal expected β-discounted n-horizon net benefit of systems starting in states $\boldsymbol{x}$ and $(\boldsymbol{x}; \boldsymbol{j})$ by $u^n(\boldsymbol{x})$ and $v^n(\boldsymbol{x}; \boldsymbol{j})$, respectively. Let $\mathcal{S}$ be the state space: $\mathcal{S} = \{\boldsymbol{x} : x_1 + x_2 \le c\}$. Arrivals occur according to a Poisson process with rate λ and, at each arrival epoch, the batch of arriving jobs consists of j_i class-i jobs, $i = 1, 2$, with probability $p_{j_1 j_2}$, where $\sum_{j_1=0}^{c} \sum_{j_2=0}^{c} p_{j_1 j_2} = 1$ and $p_{00} = 0$. We occasionally denote a batch (j_1, j_2) by $\boldsymbol{j}$ and the corresponding probability by $p_{\boldsymbol{j}}$ for brevity. We define $\boldsymbol{y}^n(\boldsymbol{x}; \boldsymbol{j}) = (y_1^n(\boldsymbol{x}; \boldsymbol{j}), y_2^n(\boldsymbol{x}; \boldsymbol{j}))$ as the optimal state to move into when a batch $\boldsymbol{j}$ arrives at the system in state $\boldsymbol{x}$ with n transitions remaining. Then, let $S(\boldsymbol{x}; \boldsymbol{j})$ be the action space for state $(\boldsymbol{x}; \boldsymbol{j})$, i.e. the set of states that can be reached from state $\boldsymbol{x}$ when a batch $\boldsymbol{j}$ arrives:

$$S(\boldsymbol{x}; \boldsymbol{j}) = \{\boldsymbol{y} \in \mathcal{S} : x_i \le y_i \le x_i + j_i,\ i = 1, 2\}.$$

Note that $S(\boldsymbol{x}; \boldsymbol{j}) = \{\boldsymbol{x}\}$ for any state $(\boldsymbol{x}; \boldsymbol{j})$ such that $x_1 + x_2 = c$.

We now present the optimality equations. Let $\boldsymbol{e}_i$ be the ith unit vector. For $x_1 + x_2 \le c$, we have

$$v^n(\boldsymbol{x}; \boldsymbol{j}) = \max\Biggl\{\sum_{i=1,2} r_i(y_i - x_i) + u^n(\boldsymbol{y}) : \boldsymbol{y} \in S(\boldsymbol{x}; \boldsymbol{j})\Biggr\}, \tag{1}$$

$$u^{n+1}(\boldsymbol{x}) = \lambda \sum_{\boldsymbol{j}} p_{\boldsymbol{j}} v^n(\boldsymbol{x}; \boldsymbol{j}) + x_1\mu_1 u^n(\boldsymbol{x} - \boldsymbol{e}_1) + x_2\mu_2 u^n(\boldsymbol{x} - \boldsymbol{e}_2) + (c\mu_2 - x_1\mu_1 - x_2\mu_2) u^n(\boldsymbol{x}), \tag{2}$$

where we define $u^n(-1, x_2) = u^n(0, x_2)$ and $u^n(x_1, -1) = u^n(x_1, 0)$. Let $\boldsymbol{y}^* = \boldsymbol{y}^n(\boldsymbol{x}; \boldsymbol{j})$ be the optimal decision if a batch $\boldsymbol{j}$ arrives in state $\boldsymbol{x}$ when there are n transitions remaining. If the nth event is an arrival of batch $\boldsymbol{j}$, then $y_i^* - x_i$ of j_i class-i jobs are accepted, meaning that the system moves to state $\boldsymbol{y}^*$ with total reward equal to $\sum_{i=1,2} r_i(y_i^* - x_i)$. If a class-$i$ job completes service with probability $x_i\mu_i$, the system state changes to state $\boldsymbol{x} - \boldsymbol{e}_i$. The 'fictitious' service completions, which occur with probability $c\mu_2 - x_1\mu_1 - x_2\mu_2$, affect neither the state nor the total reward. Finally, if the terminal state is reached (with probability β), no further reward is received.

Remark 1. In this model, we assume class-dependent rewards, r_i, which are collected at the beginning of service, immediately upon the admission of a class-i job. Indeed, these rewards can be used to model more general reward/cost functions, as our results do not change if holding or rejection costs are incurred: a system with holding cost h_i, rejection cost b_i, and reward r_i' is equivalent to a system which collects a reward of $r_i = r_i' + b_i - h_i/(\mu_i + \beta)$. Similarly, a system collecting a reward of R_i at the end of a service is equivalent to a system that collects a reward of $r_i = R_i \mu_i/(\mu_i + \beta)$ upon an admission.

2.3. Infinite-horizon models

Our main aim in this paper is to describe the structure of optimal dynamic admission policies for systems that operate over an infinite horizon. For this purpose, we first prove our results with the objective of maximizing the total expected β-discounted reward for a finite number of transitions, n, including the 'fictitious' transitions due to the 'fictitious' service completions (the final term in (2)). Thus, 'finite'-horizon problems are in fact pseudo-finite, which allows us to use the powerful tool of induction to prove our results for all finite n. In this subsection, we show that the results for finite n extend to infinite-horizon problems. To start the induction, we specify the initial function u^0 as $u^0(\boldsymbol{x}) = 0$ for all $\boldsymbol{x} \in \mathcal{S}$ in all our results. Of course, this makes no difference to the optimal policy for infinite-horizon problems.

We first analyze the model when the total expected β-discounted reward over an infinite horizon is maximized. A maximizing policy is sought in the set of all history-dependent policies. However, for this problem there is always an optimal deterministic stationary policy, due to the finiteness of the state space and the action spaces of all states and the bounded rewards (see Theorem 6.2.10 of [17]). Moreover, this policy can be computed using the value iteration algorithm. All of our results for finite-horizon problems then extend to infinite-horizon problems with discounting. Specifically, let $v(\boldsymbol{x}; \boldsymbol{j})$ and $u(\boldsymbol{x})$ denote the value functions for the infinite-horizon expected discounted reward. Then, for $\beta > 0$,

$$v(\boldsymbol{x}; \boldsymbol{j}) = \lim_{n\to\infty} v^n(\boldsymbol{x}; \boldsymbol{j}), \qquad u(\boldsymbol{x}) = \lim_{n\to\infty} u^n(\boldsymbol{x}),$$

and $y(\boldsymbol{x}; \boldsymbol{j}) = (y_1(\boldsymbol{x}; \boldsymbol{j}), y_2(\boldsymbol{x}; \boldsymbol{j}))$ is an optimal action in state $(\boldsymbol{x}; \boldsymbol{j})$, meaning that it is optimal to accept $y_i(\boldsymbol{x}; \boldsymbol{j}) - x_i$ class-i jobs when a batch of $\boldsymbol{j} = (j_1, j_2)$ arrives at a system with x_i class-i jobs.

Next we consider the criterion of maximizing the expected long-run average reward. In this case, we must define the relative value functions, $v'(\boldsymbol{x}; \boldsymbol{j})$ and $u'(\boldsymbol{x})$, and the gain, g, in the usual Markov decision process formulation. Specifically, we let

$$v'(\boldsymbol{x}; \boldsymbol{j}) = \max\left\{ \sum_{i=1,2} r_i(y_i' - x_i) + u'(\boldsymbol{y}') \colon \boldsymbol{y}' \in S(\boldsymbol{x}; \boldsymbol{j}) \right\},$$

$$\begin{aligned} g + u'(\boldsymbol{x}) = \lambda \sum_{j} p_{j} v'(\boldsymbol{x}; \boldsymbol{j}) + x_1\mu_1 u'(\boldsymbol{x} - \boldsymbol{e}_1) + x_2\mu_2 u'(\boldsymbol{x} - \boldsymbol{e}_2) \\ + (c\mu_2 - x_1\mu_1 - x_2\mu_2)u'(\boldsymbol{x}), \end{aligned}$$

where we define $\boldsymbol{y}'(\boldsymbol{x}; \boldsymbol{j}) = (y_1'(\boldsymbol{x}; \boldsymbol{j}), y_2'(\boldsymbol{x}; \boldsymbol{j}))$ as an optimal action in state $(\boldsymbol{x}; \boldsymbol{j})$. We first observe that the resulting Markov decision process is unichain, since, under all possible policies, state (0, 0) is reachable. Moreover, it is also aperiodic, due to the fictitious service completions. Theorem 8.4.5 of [17] then guarantees both the existence of an optimal deterministic stationary policy in the set of all history-dependent policies for the long-run average criterion, and the validity of the value iteration algorithm as a means of finding this policy. Moreover, as $\beta \to 0$

in the infinite-horizon problem, we obtain the long-run average problem (see Theorems 6.18 and 6.19 of [20]). Hence, all of our results for the infinite-horizon problem with discounting hold in the long-run average case.

2.4. Coupling and policy perturbations

In our proofs, we mostly use induction and coupling together. Coupling is a widely used method in Markov decision models, especially in showing the stochastic optimality of a specified policy; see [10] for some examples in the control of queueing systems and [18] for some examples in stochastic scheduling. In the context of our model, a question that often arises is that of the relative value of one state versus another. To investigate this, we define a number of systems that have the same cost structure and dynamics as the model described in Section 2.2, and start in different states. Furthermore, they are coupled in the realizations of arrivals and service completions, and follow policies related in a way that will be specified below. We describe the coupling process and the policies for two systems A and B only; the extension to more than two systems is straightforward.

First, we explicitly couple all the random variables for systems A and B as in an algorithm for simulation, or construction, of realizations. Specifically, both systems will have the same arrival stream. Moreover, the service times of jobs are coupled as follows. If the coupled jobs are of the same class then they depart at the same time; otherwise, we use the assumption that $\mu_1 \leq \mu_2$ (from which it follows that class-1 jobs are 'slow'). We let ξ be a uniformly distributed random variable in (0, 1), and we generate the service time of the class-1 job, say job d_1, and the class-2 job, say job d_2, using the same ξ, meaning that job d_2 has a shorter service time than job d_1 with probability 1. In terms of discrete time, this translates to the following: both jobs leave the system with probability μ_1, and a class-2 job departs, leaving the coupled class-1 job in the system, with probability $\mu_2 - \mu_1$. Thus, coupling does not allow a coupled class-1 job to leave the system while the coupled class-2 job is still there. The introduction of coupling brings certain restrictions on the dynamics of the two systems, implying relationships between the generated profits that facilitate the comparison of the different states.

In terms of the policies followed, we let π be an optimal policy obtained as a solution to (1) and (2). Hence, $\pi = (\pi^n)_{n\geq 1}$ specifies a rule to follow in each state $(\boldsymbol{x}; \boldsymbol{j})$ for every period n. Moreover, we can compute π^n for any given n using dynamic programming. We now let system A always follow the optimal policy π, whereas system B follows π except during a period m, when it follows a feasible rule, π_{B}^m, that is a modification of π^m. In order to ensure its feasibility, this rule depends on the initial state, $\boldsymbol{x}_{\mathrm{B}}$, of system B. In other words, in period m system B follows a rule that depends only on $\boldsymbol{x}_{\mathrm{B}}$ and π^m. In particular, π_{B}^m depends on $\pi^m(\boldsymbol{x}_{\mathrm{A}})$, where $\boldsymbol{x}_{\mathrm{A}}$ is the initial state of system A. However, there is no dependence of π_{B}^m on the sample path of system A, or on any realizations of the coupling process ξ. In this framework, the policy that system B follows belongs to the class of history-dependent policies for system B. Therefore, if $u_{\mathrm{B}}^n(\boldsymbol{x})$ is the value function of system B, we have $u_{\mathrm{B}}^n(\boldsymbol{x}) \leq u^n(\boldsymbol{x})$ for all $\boldsymbol{x}$ and all n, an inequality we use in our proofs.

2.5. Effect of an additional job

In our analysis below, the effect of admitting one class-i job is important. To see this, we let $D^n(ki)(\boldsymbol{x}) = u^n(\boldsymbol{x} + \boldsymbol{e}_k) - u^n(\boldsymbol{x} + \boldsymbol{e}_i)$ denote the difference between the total expected discounted rewards of two systems A and B, where system A starts in state $\boldsymbol{x}$ 'plus' one class-k job and system B starts in state $\boldsymbol{x}$ plus a class-i job. Note that $k = 0$ means system A is in state $\boldsymbol{x}$, i.e. there is no additional job. We occasionally omit the arguments $\boldsymbol{x}$ and n when there is no danger of confusion.

The four functions of interest are $D^n(01)$, $D^n(02)$, $D^n(21)$, and $D^n(12)$. We can interpret the difference $D^n(0i)(\boldsymbol{x})$ as the loss in future rewards because of the increased load caused by the acceptance of a class-i job, or, equivalently, the expected burden that an additional class-i job brings to the system in state $\boldsymbol{x}$ when there are n remaining transitions. Similarly, the difference $D^n(21)(\boldsymbol{x})$ represents the expected additional burden, when in state $\boldsymbol{x} + \boldsymbol{e}_2$, of changing a class-2 job that is already in the system to a class-1 job.

If the arrivals were single, the admission policy would compare the effects of two actions in state $\boldsymbol{x}$ upon a class-i arrival, namely accepting a class-i job and moving to state $\boldsymbol{x} + \boldsymbol{e}_i$ or rejecting it and remaining in state $\boldsymbol{x}$. Thus, it would be better to accept a class-i job if and only if we had

$$u^n(\boldsymbol{x}) \leq r_i + u^n(\boldsymbol{x} + \boldsymbol{e}_i)$$

or, equivalently,

$$D^n(0i)(\boldsymbol{x}) \leq r_i. \tag{3}$$

When (3) holds for all $\boldsymbol{x} \in \mathcal{S}$ for some class i, it specifies class i as a preferred class in a system with single-class batches, since in such a system it is optimal to accept as many jobs as possible from a class-i batch. To see this, assume, by contradiction, that there exists an optimal policy under which the system moves to state $\boldsymbol{y}$ with $y_i < x_i + j_i$ and $y_1 + y_2 < c$, meaning that this optimal policy rejects at least one job from the arriving class-i batch while the system has at least one idle server. Then, from the optimality equation (1), we have $u^n(\boldsymbol{y}) > r_i + u^n(\boldsymbol{y} + \boldsymbol{e}_i)$, which contradicts our assumption that (3) holds for all $\boldsymbol{x} \in \mathcal{S}$ for class i.

On the other hand, when mixed-class batches are possible, (3) is not sufficient to characterize a class as being preferred. Indeed, assume that only one server is empty and a mixed batch has arrived. Then only one job can be accepted, which calls for a comparison of the actions of accepting a class-i or a class-k job. It is better to accept a job of class i rather than of class k in state $\boldsymbol{x}$ if

$$r_k + u^n(\boldsymbol{x} + \boldsymbol{e}_k) \leq r_i + u^n(\boldsymbol{x} + \boldsymbol{e}_i)$$

or, in terms of $D^n(ki)(\boldsymbol{x})$,

$$D^n(ki)(\boldsymbol{x}) \leq r_i - r_k. \tag{4}$$

Therefore, a sufficient condition for class i to be preferred is that both (3) and (4) hold for i for all $\boldsymbol{x}$. When class i is preferred, free servers are filled with class-i jobs first, and only then is the admission of jobs of the other class considered.

3. Existence of a preferred class

We define a preferred class as the class whose jobs are always admitted to the system whenever there are available servers. As discussed earlier, this definition leads to different characterizations of preferred class(es) with single-class and mixed batches, which are considered separately in this section. In fact, the results for single-class batches, as well as their proofs, are very similar to those for systems with single arrivals (see [15]). Hence, we will only state the results, and our corresponding intuitions, for this case.

In determining preferred class(es), two different criteria can be considered: lump rewards and average rewards. We would expect that whenever the corresponding reward of class i is higher than that of the other class, class i is preferred. We will discuss the validity of both criteria in conjunction with the related results.

Before separately analyzing the systems with mixed and single-class batches, we prove the following result, which applies to both systems.

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

Lemma 1. *For all* $\boldsymbol{x} \in \mathcal{S}$ *and all* $n \geq 0$,

(i) $D^n(0i)(\boldsymbol{x}) \geq 0$ *for* $i = 1, 2$, *and*

(ii) $D^n(21)(\boldsymbol{x}) = -D^n(12)(\boldsymbol{x}) \geq 0$.

Proof. We prove the statements by a sample path analysis.

(i) Assume that system A is in state $\boldsymbol{x}$ and system B in state $\boldsymbol{x} + \boldsymbol{e}_i$ during period n. We let system B move to state $\boldsymbol{y}^* = \boldsymbol{y}^n(\boldsymbol{x} + \boldsymbol{e}_i; \boldsymbol{j})$, following the optimal policy, and let system A move to state $\boldsymbol{y}^* - \boldsymbol{e}_i$. In other words, systems A and B always accept the same numbers of jobs from each class. We couple the two systems via the service and interarrival times, so that, except for the additional job in system B, all the departure and arrival times are the same in both systems. Then all future rewards of systems A and B are the same:

$$D^n(0i)(\boldsymbol{x}) = u^n(\boldsymbol{x}) - u^n(\boldsymbol{x} + \boldsymbol{e}_i) \geq u^n_{\mathrm{A}}(\boldsymbol{y}^* - \boldsymbol{e}_i) - u^n(\boldsymbol{y}^*) = 0,$$

where u^n_{A} is the expected discounted return of system A.

(ii) Assume that system A starts in state $\boldsymbol{x} + \boldsymbol{e}_2$ and system B in state $\boldsymbol{x} + \boldsymbol{e}_1$, where we now couple the additional class-2 job, say job d_2, in system A with the additional class-1 job, say job d_1, in system B (as well as all other service and interarrival times) such that, as discussed earlier, if d_1 leaves the system then d_2 also leaves. Then we can let system B follow the optimal policy, moving to state $\boldsymbol{y}^* = \boldsymbol{y}^n(\boldsymbol{x} + \boldsymbol{e}_1; \boldsymbol{j})$, and let system A accept exactly the same jobs, so that it moves to state $\boldsymbol{y}^* - \boldsymbol{e}_1 + \boldsymbol{e}_2$. Now, as before, all future rewards of the systems are equal:

$$D^n(21)(\boldsymbol{x}) = u^n(\boldsymbol{x} + \boldsymbol{e}_2) - u^n(\boldsymbol{x} + \boldsymbol{e}_1) \geq u^n_{\mathrm{A}}(\boldsymbol{y}^* - \boldsymbol{e}_1 + \boldsymbol{e}_2) - u^n(\boldsymbol{y}^*) = 0.$$

This lemma shows that it is always preferable to be in a state where there are fewer or faster jobs. Recall that $u^n(\boldsymbol{x})$ is equal to the expected discounted total reward of the system under an optimal policy when there are n transitions remaining. Since rewards are collected at the beginning of service, jobs that are already in the system do not contribute to $u^n(\boldsymbol{x})$. Therefore, the jobs initially in the system bring only more burden, by blocking acceptance of future jobs.

3.1. Mixed batches

In this section, we assume that there exists a batch $\boldsymbol{j} = (j_1, j_2)$ with $j_1 > 0$, $j_2 > 0$, and $p_{j_1 j_2} > 0$, meaning that we have batches consisting of both classes. Whenever a mixed batch arrives at the system, i.e. when a batch $\boldsymbol{j}$, $j_1 > 0$, $j_2 > 0$, arrives, we must compare at least three actions: having an empty server or having a server work on a class-1 or a class-2 job. As discussed earlier, in Section 2.5, there can be at most one preferred class for these systems. Moreover, class i is preferred if $D^n(ki)(\boldsymbol{x}) \leq r_i - r_k$ and $D^n(0i)(\boldsymbol{x}) \leq r_i$, $i \neq k$, for all x and for all n. Therefore, the effect of changing a class-i job to class k, as measured by $D^n(ki)(\boldsymbol{x})$, is as important as the effect of an additional class-i job, as measured by $D^n(0i)(\boldsymbol{x})$, in determining the preferred class.

We first present the sufficient conditions for class 2 to be preferred.

Proposition 1. *If* $r_2 \geq r_1$ *then class 2 is the single preferred class.*

Proof. Assume that $r_2 \geq r_1$. It suffices to show that $D^n(12)(\boldsymbol{x}) \leq r_2 - r_1$ and $D^n(02)(\boldsymbol{x}) \leq r_2$ for all $\boldsymbol{x}$ and all n.

To prove the former statement, we use Lemma 1 and the assumption that $r_2 \geq r_1$ to immediately obtain

$$D^n(12)(\boldsymbol{x}) \leq 0 \leq r_2 - r_1 \quad \text{for all } x \in \mathcal{S} \text{ and all } n.$$

Hence, we need only show that $D^n(02)(\boldsymbol{x}) \leq r_2$ for all $\boldsymbol{x}$ and all n. Now, $D^0(02)(\boldsymbol{x}) \leq r_2$ if $u^0(\boldsymbol{x}) = 0$ for all $\boldsymbol{x} \in \mathcal{S}$. Thus, assume that $D^n(02)(\boldsymbol{x}) \leq r_2$ for all x and some n, and consider $D^{n+1}(02)(\boldsymbol{x})$. Let system A be in state $\boldsymbol{x}$ and system B be in state $\boldsymbol{x} + \boldsymbol{e}_2$ during period $n+1$. We let system A take optimal actions $\boldsymbol{y}^* = \boldsymbol{y}^{n+1}(\boldsymbol{x}; \boldsymbol{j})$, and let system B imitate these actions whenever possible. If system A accepts at least one class-2 job, meaning that $y_2^* > x_2$, then both systems end up in the same state with a difference of r_2 in rewards, since system B rejects one of the class-2 jobs that system A accepts. If $y_2^* = x_2$ and $y_1^* + y_2^* < c$ then system B moves to state $\boldsymbol{y}^* + \boldsymbol{e}_2$, meaning that system B accepts exactly the same jobs as system A, resulting in the same amount of reward. If $y_2^* = x_2$ and $y_1^* + y_2^* = c$, meaning that $y_1^* > x_1$, then system B moves to state $\boldsymbol{y}^* + \boldsymbol{e}_2 - \boldsymbol{e}_1$, accepting one fewer class-1 job than system A, so the difference in the rewards of the two systems is r_1. If the extra job in system B leaves, which happens with probability μ_2, then the two systems couple with no difference in rewards. If there is any other service completion then the difference between the two systems remains the same, due to the extra class-2 job in system B. If $u_{\mathrm{B}}^{n+1}(\boldsymbol{x} + \boldsymbol{e}_2)$ is the expected discounted return of system B, then

$$\begin{aligned}
D^{n+1}(02)(\boldsymbol{x}) = u^{n+1}(\boldsymbol{x}) - u^{n+1}(\boldsymbol{x} + \boldsymbol{e}_2) &\leq u^{n+1}(\boldsymbol{x}) - u_{\mathrm{B}}^{n+1}(\boldsymbol{x} + \boldsymbol{e}_2) \\
&\leq \lambda \max_{z \in \mathcal{S}} \{r_2, D^n(02)(z), r_1 + D^n(12)(z)\} \\
&\quad + (c-1)\mu_2 \max_{z \in \mathcal{S}} \{D^n(02)(z)\} \\
&\leq \lambda \max\{r_2, r_2, r_1 + r_2 - r_1\} + (c-1)\mu_2 r_2 \\
&< r_2,
\end{aligned}$$

where the first inequality holds because the policy followed by system B is not necessarily optimal, the second is due to the coupling described above, and the third follows by the induction hypothesis $D^n(02)(\boldsymbol{x}) \leq r_2$ and the fact that $D^n(12)(\boldsymbol{x}) \leq r_2 - r_1$ under the assumption that $r_2 \geq r_1$. Hence,

$$D^n(02)(\boldsymbol{x}) \leq r_2 \quad \text{and} \quad D^n(12)(\boldsymbol{x}) \leq r_2 - r_1 \quad \text{for all } x \text{and all } n,$$

implying that class-2 jobs are preferred.

We can state Proposition 1 in terms of the lump reward criterion to determine preferred class(es): if the lump reward of class 2 (the faster class) is higher, then class 2 is preferred. Obviously, class 1 (the slower class) is not necessarily preferred if $r_1 > r_2$, as we may have $\mu_1 \ll \mu_2$. Hence, the criterion of lump rewards favors class 2 by emphasizing its advantage of being 'fast'. Indeed, the condition given in Proposition 1 is a strong requirement, as we could expect class 2 still to be preferred even if r_2 were slightly lower than r_1. In fact, we will find a weaker condition on the r_i for class 2 to be preferred, when the system receives single-class batches (see Proposition 3, below).

We now derive a sufficient condition for class 1 to be preferred. However, this requires more effort, since we have to consider the differences $D^n(01)(\boldsymbol{x})$ and $D^n(21)(\boldsymbol{x})$ simultaneously.

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

Lemma 2. *If*

$$(\lambda + \mu_1 + \beta) r_2 (\mu_2 + \beta) \leq (\lambda + \mu_2 + \beta) r_1 (\mu_1 + \beta)$$

then, for all $\boldsymbol{x} \in \mathcal{S}$ *and all* $n \geq 0$,

(i) $D^n(01)(\boldsymbol{x}) \leq \dfrac{\lambda r_1}{\lambda + \mu_1 + \beta}$ *and*

(ii) $D^n(21)(\boldsymbol{x}) \leq r_1 - r_2$.

Proof. We first note that, under the assumption of the lemma, $r_1 \geq r_2$. We use induction on the number of transitions, n. Both statements are satisfied if $u^0(\boldsymbol{x}) = 0$ for all $\boldsymbol{x} \in \mathcal{S}$. Assume that both are true for a general n. Then, note that $D^n(01)(\boldsymbol{x}) \leq r_1$ for all $\boldsymbol{x} \in \mathcal{S}$, by part (i). We now have to consider two pairs of systems, one for $D^{n+1}(01)(\boldsymbol{x})$ and the other for $D^{n+1}(21)(\boldsymbol{x})$.

(i) Consider the first pair. Assume that system A is in state $\boldsymbol{x}$ and system B in state $\boldsymbol{x} + \boldsymbol{e}_1$ during period $n+1$. We let system A follow an optimal policy and set $\boldsymbol{y}^* = \boldsymbol{y}^n(\boldsymbol{x}; \boldsymbol{j})$. System B takes the same action as system A, whenever it is possible. If $y_1^* > x_1$ then system B also moves to state $\boldsymbol{y}^*$, rejecting one of the class-1 jobs system A accepts, meaning that the two systems couple with a difference in rewards of r_1. If $y_1^* = x_1$ and $y_1^* + y_2^* < c$ then system B moves to state $\boldsymbol{y}^* + \boldsymbol{e}_1$. In this case, system B accepts exactly the same jobs as system A, resulting in the same amount of reward. If $y_1^* = x_1$ and $y_1^* + y_2^* = c$, meaning that $y_2^* > x_2$, then system B moves to state $\boldsymbol{y}^* + \boldsymbol{e}_1 - \boldsymbol{e}_2$, accepting one fewer class-2 job than system A, so the difference in the rewards of the two systems is r_2. With the departure of the additional class-1 job in system B, both systems enter the same state with no difference in reward, whereas all other service completions keep the difference between the two systems the same, due to the extra class-1 job. Thus,

$$\begin{aligned}
D^{n+1}(01)(\boldsymbol{x}) &\leq u^{n+1}(\boldsymbol{x}) - u_{\mathrm{B}}^{n+1}(\boldsymbol{x} + \boldsymbol{e}_1) \\
&\leq \lambda \max_{\boldsymbol{y} \in \mathcal{S}} \{r_1, D^n(01)(\boldsymbol{y}), r_2 + D^n(21)(\boldsymbol{y})\} \\
&\quad + (c\mu_2 - \mu_1) \max_{\boldsymbol{y} \in \mathcal{S}} \{D^n(01)(\boldsymbol{y})\} \\
&\leq \lambda \max\{r_1, r_2 + r_1 - r_2\} + (c\mu_2 - \mu_1) \frac{\lambda r_1}{\lambda + \mu_1 + \beta} \\
&= \lambda r_1 + (1 - \lambda - \mu_1 - \beta) \frac{\lambda r_1}{\lambda + \mu_1 + \beta} \\
&= \frac{\lambda r_1}{\lambda + \mu_1 + \beta},
\end{aligned}$$

where the first and second inequalities are due to the policy followed by system B, the third is due to the induction hypotheses $D^n(01)(\boldsymbol{x}) = \lambda r_1/(\lambda + \mu_1 + \beta) < r_1$ and $D^n(21)(\boldsymbol{x}) \leq r_1 - r_2$, and the equalities are due to uniformization, i.e. $\lambda + c\mu_2 + \beta = 1$, and some algebra. Thus, the first statement holds for all $\boldsymbol{x} \in \mathcal{S}$ and all $n \geq 0$.

(ii) Now consider the second pair of systems. Let system A′ be in state $\boldsymbol{x} + \boldsymbol{e}_2$ and system B′ be in state $\boldsymbol{x} + \boldsymbol{e}_1$ during period $n+1$. System A′ takes the optimal actions, where we set $\boldsymbol{y}^{*\prime} = \boldsymbol{y}^n(\boldsymbol{x} + \boldsymbol{e}_2; \boldsymbol{j})$, and we let system B′ accept exactly the same jobs as system A′, meaning that system B′ moves to state $\boldsymbol{y}^{*\prime} + \boldsymbol{e}_1 - \boldsymbol{e}_2$ during this period, producing no difference in the rewards of the two systems. As in Lemma 1, we couple the additional class-2 job, say job d_2, in

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

system B′ with the additional class-1 job, say job d_1, as well as all other service and interarrival times. Then, if d_1 leaves the system, which happens with probability μ_1, then d_2 also leaves. The departure of d_1 leads the two systems to couple with no difference in rewards; the departure of d_2 alone, which happens with probability $\mu_2 - \mu_1$, takes the systems to two different states, $\boldsymbol{x} + \boldsymbol{e}_1$ and $\boldsymbol{x}$, again with no difference in rewards; and, whenever there is any other transition, both systems retain their additional jobs, meaning that the difference between the two systems is due only to changing a class-1 job to class 2. Thus,

$$\begin{aligned} D^{n+1}(21)(\boldsymbol{x}) &\le \lambda D^n(21)(\boldsymbol{y}^{*\prime} - \boldsymbol{e}_2) + (\mu_2 - \mu_1) D^n(01)(\boldsymbol{x}) \\ &\quad + (c-1)\mu_2 \max_{\boldsymbol{y}\in\mathcal{S}}\{D^n(21)(\boldsymbol{y})\} \\ &\le (\lambda + (c-1)\mu_2) \max_{\boldsymbol{y}\in\mathcal{S}}\{D^n(21)(\boldsymbol{y})\} + (\mu_2 - \mu_1) \max_{\boldsymbol{y}\in\mathcal{S}}\{D^n(01)(\boldsymbol{y})\} \\ &\le (1 - \mu_2 - \beta)(r_1 - r_2) + (\mu_2 - \mu_1)\frac{\lambda r_1}{\lambda + \mu_1 + \beta} \\ &= r_1 - r_2 + r_2(\mu_2 + \beta) - r_1(\mu_1 + \beta)\frac{\lambda + \mu_2 + \beta}{\lambda + \mu_1 + \beta} \\ &\le r_1 - r_2, \end{aligned}$$

where the first inequality is due to the coupling, the third follows by uniformization and the induction hypotheses for $D^n(01)(\boldsymbol{x})$ and $D^n(21)(\boldsymbol{x})$, and the last one is due to the assumption that

$$(\lambda + \mu_1 + \beta) r_2(\mu_2 + \beta) \le (\lambda + \mu_2 + \beta) r_1(\mu_1 + \beta).$$

This proves the second part of the lemma.

We can now conclude that class-1 jobs are preferred under the following condition.

Proposition 2. *If*

$$(\lambda + \mu_1 + \beta) r_2(\mu_2 + \beta) \le (\lambda + \mu_2 + \beta) r_1(\mu_1 + \beta),$$

then class 1 is the single preferred class.

The condition specified in this proposition compares the average rewards of the two classes. A class-i job stays in the system for an exponential amount of time with rate $\mu_i + \beta$, since it either finishes its service with rate μ_i or the system closes down with rate β. Thus, a class-i job with a lump reward of r_i brings an average reward of $r_i(\mu_i + \beta)$. Proposition 2 shows that, when the average reward of class 1 (the slower class) is higher than that of class 2, class 1 is preferred. In a queueing environment with two classes of job, this criterion would determine a preferred class, as shown in [6]; however, in a loss system it favors slower jobs: if both classes bring the same average reward per unit time, then the system will prefer the one whose jobs occupy a server for a longer time, as this server will then generate a reward for a longer period. The difference between loss and queueing systems is due to the fact that a queueing system is not concerned with the 'occupation' time of a server, since all jobs can wait, i.e. there is no loss of work.

On the other hand, class 2 (the faster class) is not necessarily preferred if $r_1(\mu_1 + \beta) < r_2(\mu_2 + \beta)$. Consider a firm that must choose one from two jobs, one of which brings a \$1000 profit each month for 12 months and the other a \$1200 profit per month for only 3 months.

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

The possibility that the firm will have no job after 3 months works in favor of the job of longer duration. Hence, class-2 jobs with higher average rewards can be rejected if $\mu_1 \ll \mu_2$. We finally note that the average reward criterion favors class 1 more than the lump reward criterion favors class 2: class 1 can still be preferred even if its average reward is slightly lower than that of class 2, whereas our results cannot guarantee that class 2 is preferred when the lump reward of class 2 is slightly lower than that of class 1; compare Proposition 2 with Proposition 1.

As we expected, the condition for class 1 to be preferred excludes the condition for class 2 to be preferred, since at most one class can be preferred when the system receives mixed batches. Moreover, there is a range of parameters over which we do not know if there exists a preferred class.

Corollary 1. *If the parameter values are such that*

$$\frac{\lambda + \mu_2 + \beta}{\lambda + \mu_1 + \beta} < \frac{r_2(\mu_2 + \beta)}{r_1(\mu_1 + \beta)} < \frac{\mu_2 + \beta}{\mu_1 + \beta}, \tag{5}$$

then our results are inconclusive regarding the existence of a preferred class.

In Subsection 3.3, we will numerically analyze systems satisfying condition (5). Note that, for all λ, μ_1, and μ_2, the range given by (5) is nonempty, meaning that, for all possible arrival and service rates, there are rewards, r_i, for which our results cannot guarantee the existence of a preferred class. As we will see in the next subsection, this is not the case for systems with single-class batches.

3.2. Single-class batches

In this section, we assume that $\sum_{j_1=1}^{c} p_{j_1 0} + \sum_{j_2=1}^{c} p_{0 j_2} = 1$, meaning that the batches can have jobs of only one class. In this system, we must consider the trade-off between having an empty server and a server occupied by a class-i job upon the arrival of a batch with j_i class-i jobs. In this case, if $D^n(0i)(\boldsymbol{x}) \le r_i$ for all $\boldsymbol{x}$ and all n, then class i is preferred. Since we do not have to compare the actions of having a server work on a class-1 job or a class-2 job, the system can have 0, 1, or 2 preferred classes.

For preferred classes in systems receiving single-class batches, we present the results without proof, since the corresponding proofs are very similar to those in systems with single arrivals (see [15]). The following proposition summarizes them.

Proposition 3. *Let λ_i be the arrival rate of batches with class-i jobs, meaning that $\lambda_1 = \lambda \sum_{j_1=0}^{c} p_{j_1 0}$ and $\lambda_2 = \lambda \sum_{j_2=0}^{c} p_{0 j_2}$. The following statements hold.*

(i) *If $\lambda_1 r_1 \le (\lambda_1 + \mu_2 + \beta) r_2$ then $D^n(02)(\boldsymbol{x}) \le r_2$ for all $\boldsymbol{x} \in \mathcal{S}$ and all n; hence, class 2 is a preferred class.*

(ii) *If $r_2(\mu_2 + \beta)\lambda_2 \le r_1(\mu_1 + \beta)(\lambda_2 + \mu_2 + \beta)$ then, for all $\boldsymbol{x} \in \mathcal{S}$ and all n,*

$$D^n(21)(\boldsymbol{x}) \le \frac{(\mu_2 - \mu_1) r_1}{\mu_2 + \beta} \quad \text{and} \quad D^n(01)(\boldsymbol{x}) \le r_1;$$

hence, class 1 is a preferred class.

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

The conditions for both classes to be preferred are relaxed in systems with single-class batches: in mixed batches, the condition $r_2 \geq r_1$ guarantees class 2 to be preferred, whereas now class 2 is still preferred even when r_2 is slightly lower than r_1. The condition on the average rewards for class 1 to be preferred is also relaxed in single-class batch systems; compare Propositions 2 and 3.

As mentioned earlier, the results of Proposition 3 are the same as those for systems receiving single class-i jobs according to a Poisson process with rate λ_i (see [15] for results on systems with single arrivals). This is a natural consequence of their common acceptance criterion for a class-i job, i.e. $D^n(0i)(\boldsymbol{x}) \leq r_i$, which translates to the same trade-off for both systems at each arrival epoch. Moreover, 'being preferred' is a global property: if class i is preferred then jobs of this class are accepted whenever there is an available server, regardless of the current state of the system. Therefore, the effect of batch arrivals vanishes when the issue is determining the preferred class(es).

3.3. A discussion of preferred classes

We have discussed the meaning of 'being preferred', and derived sufficient conditions for a class to be preferred in systems with single-class batches and systems with mixed batches. This subsection mainly discusses the existence of a preferred class through numerical examples; thus, the meaning of 'no preferred class' also becomes important. In the most general terms, a system with 'no preferred class' can be regarded as a system that changes its preference of class with respect to the state of the system. Such a dependence on the state is surprising since we expect one of the classes to have 'globally better' qualities in terms of its service rates, μ_i, and rewards, r_i. However, we have observed systems with no preferred class, all of which receive mixed batches, meaning that the change in their preferences is rather 'mild', as we will discuss below.

Now consider the meaning of 'no preferred class' in systems receiving single-class batches and systems receiving mixed batches. Recall that in the former systems, class i is preferred if $D(0i)(\boldsymbol{x}) \leq r_i$ for all $\boldsymbol{x}$, whereas the latter systems require $D(0i)(\boldsymbol{x}) \leq r_i$, $i \neq k$, *and* $D(ki)(\boldsymbol{x}) \leq r_i - r_k$ for all $\boldsymbol{x}$ for class i to be preferred. Thus, a system has no preferred class, regardless of whether the system receives mixed or single-class batches, if there exist two states $\boldsymbol{x}$ and $\boldsymbol{x}'$ such that $D(0i)(\boldsymbol{x}) > r_i$, $D(0k)(\boldsymbol{x}) \leq r_k$, $D(0i)(\boldsymbol{x}') \leq r_i$, and $D(0k)(\boldsymbol{x}') > r_k$. This indicates a major change in the preference of the system with respect to the state, as an idle server is preferred over each class in different states, a situation that we never observe in our examples. In addition, a system receiving mixed batches and satisfying $D(0i)(\boldsymbol{x}) \leq r_i$ for all $\boldsymbol{x}$ does not have a preferred class if it contains two states $\boldsymbol{x}$ and $\boldsymbol{x}'$ with $D(ik)(\boldsymbol{x}) > r_k - r_i$ and $D(ik)(\boldsymbol{x}') \leq r_k - r_i$. Such a system accepts as many class-i jobs as possible when the incoming batch consists only of such jobs, while it rejects at least one class-i job in the presence of jobs of the other class. In fact, all the systems we have observed as having no preferred class have satisfied $D(0i)(\boldsymbol{x}) \leq r_i$ for all $\boldsymbol{x}$ for *both* $i = 1$ and $i = 2$, while violating the condition $D(ki)(\boldsymbol{x}) \leq r_i - r_k$ for some state(s) $\boldsymbol{x}$. In other words, these systems accept as many jobs as possible from all single-class batches, but prefer different classes in different states when the batches consist of jobs of both classes. This suggests that systems with no preferred class have a 'very mild' change in their preference of class(es) with respect to the state of the system.

In this subsection, we present examples of systems receiving mixed batches, and numerically explore the optimal strategy in a large set of systems for which our results are inconclusive, i.e. when the parameters satisfy (5). However, as the above discussion suggests, they also give insight into systems with single-class batches.

Example 1. We consider a system with five servers over an infinite horizon, and set $\beta = 1$, $\mu_1 = 1$, $r_2 = 1$, $p_{10} = \frac{1}{6}$, $p_{02} = \frac{29}{60}$, $p_{13} = \frac{1}{3}$, $p_{55} = \frac{1}{60}$, and let r_1 vary between 1.1 and 6.1, λ vary between 1 and 31, and μ_2 vary between 1.1 and 6.1, all in increments of 0.1. This way, a total of 750 000 examples is created, 129 686 of which satisfy (5). In all of these 129 686 examples, when a batch $\boldsymbol{j} = (0, 2)$ arrives, the optimal policy is to accept as many class-2 jobs as possible. In 79 031 of them, a class-1 job is rejected in at least one state. However, this does not mean that class 2 is preferred in all these systems, since there are two different kinds of optimal policy that give higher or 'equal' priority to class 1. For the first kind, which were observed 6094 times, class 1 is the preferred class, i.e. when a mixed batch arrives, the system first accepts as many class-1 jobs as possible, and only then, depending on the availability of the servers, accepts class-2 jobs. Policies of the second kind provide examples with no preferred class: in 11 215 examples, an empty system receiving a batch (5, 5) chooses to move to states (4, 1), (3, 2), (2, 3), or (1, 4), rather than (5, 0) or (0, 5), which obviously violates $D(ki)(\boldsymbol{x}) \leq r_i - r_k$ for all $\boldsymbol{x}$ for both $i = 1$ and $i = 2$. Note that all of these systems satisfy $D(0i)(\boldsymbol{x}) \leq r_i$ for all $\boldsymbol{x}$ for both $i = 1$ and $i = 2$. Hence, when the arriving batch consists of only one class, specifically when $\boldsymbol{j} = (1, 0)$ or $\boldsymbol{j} = (0, 2)$, optimal policies accept as many jobs as possible in all states.

In light of this example, the optimal policies of the 'inconclusive' parameter region can be grouped into three categories: one with class 1 as the preferred class, one with no preferred class, and one with class 2 as the preferred class. Let L_1 be the left-most and L_2 be the right-most expressions in (5). The optimal policies of the systems considered in Example 1 lead to the following crude observations. Class 1 is preferred when the ratio $r_2(\mu_2+\beta)/r_1(\mu_1+\beta)$ is close to L_1. As this ratio increases, we first observe no preferred class, and then class 2 becomes preferred. When the ratio becomes large, class-1 jobs can be rejected in some states. However, the cut-off points for the changes in the optimal policies cannot be pinpointed. In our next example, we show the relation between the optimal policies and the ratio $r_2(\mu_2+\beta)/r_1(\mu_1+\beta)$ more precisely in a two-server system.

Example 2. Consider a system with two servers over an infinite horizon, and set $\beta = 1$, $\lambda = 30$, $\mu_1 = 2$, $\mu_2 = 3$, $r_1 = 1$, $p_{10} = \frac{1}{6}$, $p_{01} = \frac{1}{3}$, and $p_{22} = \frac{1}{2}$. With L_1 and L_2 as given above, we then have $L_1 = \frac{34}{33}$ and $L_2 = \frac{4}{3}$, meaning that if r_2 varies between $\frac{17}{22}$ and 1, the whole inconclusive region given in Corollary 1 will be explored. Figure 1 shows the optimal policies as r_2 changes in increments of $\frac{1}{220}$. There are several exceptional increments, corresponding to the changes in the structures of the optimal policies.

We plot whether or not it is optimal to admit an incoming class-i job in all states, which corresponds to the optimal number of jobs to be accepted when a batch of (1, 0) or (0, 1) arrives. Actions indicated as 'accept none' refer to the rejection of all jobs due to the system being full. The plots also show the optimal action for state (0, 0) when a batch (2, 2) arrives, which can be interpreted as the 'best' state for the empty system to move into when it has the option. Figure 1 does not describe the optimal actions in states (1, 0) and (0, 1) when a batch (2, 2) arrives; these are as follows: all of the actions try to reach a state that is closer to the 'best' state, meaning that in case (a) it is optimal to accept one of the class-1 jobs from batch (2, 2) in both states (1, 0) and (0, 1), in cases (c), (d), (e), and (f) it is optimal to accept one of the class-2 jobs in these states, whereas in case (b) it is optimal to accept a class-1 job in state (0, 1) and a class-2 job in state (1, 0).

For $r_2 = 17$, class 1 is known to be preferred, as shown in the figure. Class 1 is still preferred when $r_2 = \frac{1702}{2200}$. However, the system with $r_2 = \frac{1703}{2200}$ does not have a preferred class, due to

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

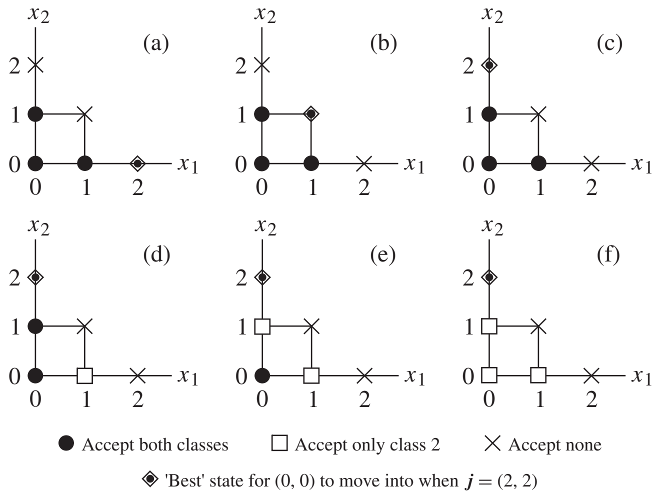

FIGURE 1: Illustration of optimal policies for the system in Example 2 with (a) $r_2 = \frac{17}{22}$ and $r_2 = \frac{1702}{2200}$, (b) $r_2 = \frac{1703}{2200}$, (c) $r_2 = \frac{1704}{2200}, r_2 = \frac{1710}{2200}, \dots, \frac{1920}{2200}$, and $r_2 = \frac{1921}{2200}$, (d) $r_2 = \frac{1922}{2200}, \dots, \frac{1924}{2200}$, (e) $r_2 = \frac{1925}{2200}$ and $r_2 = \frac{1930}{2200}, \dots, \frac{20}{22}$, and (f) $r_2 = \frac{2010}{2200}, \dots, \frac{22}{22} = 1$.

the optimal actions in all states when a batch $\boldsymbol{j} = (2, 2)$ arrives. For example, it is optimal to accept one job of each class while rejecting one of each in state $(0, 0; 2, 2)$. Class 2 then becomes preferred after an increment in r_2 of only $\frac{1}{2200} = 4.55 \times 10^{-4}$, since the system with $r_2 = \frac{1704}{2200}$ prefers class 2. Notice that jobs of both classes are always accepted in cases (a), (b), and (c) when the batches consist of only one class. Class 2 is preferred in all other cases, and the rejection region for class 1 grows as r_2 increases.

All examples of systems with no preferred class admit all jobs in single-class batches of either class. We have discussed how there are advantages and disadvantages to both classes and how, apparently, under certain conditions a combination of the two classes creates a good mix, yielding higher revenues in the long run. In other words, for systems with no preferred class, neither class has strict priority over the other; rather, their overall qualities, determined by their service rates, μ_i, and rewards, r_i, are compatible. This also provides strong evidence for expecting that systems receiving mixed batches with no preferred class would have two preferred classes if the system were receiving only single-class batches. Therefore, we conjecture that systems receiving single-class batches always have at least one preferred class. In fact, systems with single-class batches are very similar to single-arrival systems in terms of the existence of a preferred class, so for further details we refer the reader to [15]. We also observe that the structure of the optimal policy changes under a small increment in the parameters, which to some extent justifies the gap in our results regarding the existence of preferred class(es).

Finally, we have three remarks on the relation between the number of servers and the arrival and service rates of jobs with preferred class(es). These remarks apply to both single-class and mixed batch systems.

Remark 2. In all results in this section, the inequalities and the conditions on the parameters are independent of the number of servers. At first sight, this is surprising. However, it is not really so remarkable, as all these results provide sufficient conditions for always accepting jobs

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

of some class; the main issue is then with what decisions are to be made when almost all of the servers, or, more precisely, all but one of the servers, are busy, regardless of how many there are.

Remark 3. Whether or not a class is preferred also depends on the arrival stream of jobs: in [15] an example was presented of a system with single arrivals, in which classes are preferred on the basis of differing sets of arrival rates, even though the service rates, μ_i, and the rewards, r_i, are kept fixed.

Remark 4. The two criteria we have considered, i.e. lump rewards and average rewards, fail to determine a preferred class correctly when $\mu_1 \ll \mu_2$. In fact, for the other extreme, $\mu_1 = \mu_2$, Örmeci and Burnetas [13] completely characterized an optimal policy: jobs of the class bringing the highest reward are preferred, and the optimal admission policy for jobs of the other classes is of sequential threshold type, where the thresholds are monotone.

4. Monotonicity of the optimal value functions

Intuitively, we expect that it should be less profitable to accept jobs when there are many jobs already in the system, or, equivalently, that the benefit of additional jobs should decrease as the number of jobs already in the system increases. This conjecture leads to optimal policies with monotone thresholds in single-arrival systems. Altman *et al.* [2] showed the submodularity of the value functions, which guarantees the existence of an optimal threshold policy for a stochastic knapsack with two classes of job. For these knapsacks, the monotonicity of thresholds cannot be guaranteed, as the concavity of the value functions has not been proven for all possible parameter values. In the examples we have constructed, concavity is never violated but, due to the boundary conditions on the loss system, and having multiple servers and batch arrivals, we could not prove it analytically. In fact, even for systems with single arrivals, which correspond to stochastic knapsacks of unit size, $b_i = 1$, Örmeci *et al.* [15] were able to establish concavity only under very restrictive conditions on the parameters, which imply certain upper bounds on $D^n(ik)$. However, this method fails for systems receiving batch arrivals.

Our next lemma proves the submodularity of the optimal value functions for systems with batch arrivals, a mathematically challenging result due to the boundary effects, multiple servers, and batch arrivals.

Lemma 3. *For all* $\boldsymbol{x} + \boldsymbol{e}_1 + \boldsymbol{e}_2 \in \mathcal{S}$, *we have*

$$u^n(\boldsymbol{x}) - u^n(\boldsymbol{x} + \boldsymbol{e}_2) - u^n(\boldsymbol{x} + \boldsymbol{e}_1) + u^n(\boldsymbol{x} + \boldsymbol{e}_1 + \boldsymbol{e}_2) \leq 0 \quad \text{for all } n \geq 1, \tag{6}$$

whenever the inequality holds for $n = 0$.

The proof of the lemma is complicated, and so it is given in Appendix A along with several definitions (which are not used in any of our other results) and another lemma.

We now discuss the implications of Lemma 3. Unfortunately, submodularity cannot guarantee an optimal threshold policy when batch arrivals are allowed. However, it shows that optimal policies are more reluctant to accept class-i jobs when the number of class-k jobs, $k \neq i$, increases in the system, which then reduces the number of potentially optimal actions in certain states. To see this, we first rewrite (6) in terms of $D^n(0i)(\boldsymbol{x}) = u^n(\boldsymbol{x}) - u^n(\boldsymbol{x} + \boldsymbol{e}_i)$ for $i = 1, 2$:

$$\begin{aligned} D^n(02)(\boldsymbol{x}) &\leq D^n(02)(\boldsymbol{x} + \boldsymbol{e}_1), \\ D^n(01)(\boldsymbol{x}) &\leq D^n(01)(\boldsymbol{x} + \boldsymbol{e}_2). \end{aligned} \tag{7}$$

We have interpreted the difference $D^n(0i)(\boldsymbol{x})$ as the expected burden that an additional class-i job brings to the system in state $\boldsymbol{x}$. Hence, these inequalities imply that the burden of the system due to an additional class-i job is increasing in the number of class-k jobs, $k \neq i$. This is essentially the implication we stated above: when the number of class-k jobs increases, the system is more reluctant to accept class-i jobs. In addition, we would like to show that the optimal policy tends to reject class-k jobs as the number of class-k jobs increases. This would typically require us to establish concavity of the value functions in x_k for a fixed x_i. However, as mentioned above, concavity has not been proven generally even for systems with single arrivals.

To investigate the implications of Lemma 3 on the structure of the optimal policy, we now observe that (6) implies a seemingly stronger inequality almost immediately. We first iterate on $\boldsymbol{x} + \boldsymbol{e}_1$ using (7), to obtain

$$u^n(\boldsymbol{x}) - u^n(\boldsymbol{x} + \boldsymbol{e}_2) \leq u^n(\boldsymbol{x} + b_1\boldsymbol{e}_1) - u^n(\boldsymbol{x} + b_1\boldsymbol{e}_1 + \boldsymbol{e}_2)$$

for all $b_1 \geq 0$. This can be rewritten as

$$u^n(\boldsymbol{x}) - u^n(\boldsymbol{x} + b_1\boldsymbol{e}_1) \leq u^n(\boldsymbol{x} + \boldsymbol{e}_2) - u^n(\boldsymbol{x} + b_1\boldsymbol{e}_1 + \boldsymbol{e}_2). \tag{8}$$

We now iterate on $\boldsymbol{x} + \boldsymbol{e}_2$ using (8), to obtain

$$u^n(\boldsymbol{x}) - u^n(\boldsymbol{x} + b_1\boldsymbol{e}_1) \leq u^n(\boldsymbol{x} + b_2\boldsymbol{e}_2) - u^n(\boldsymbol{x} + b_1\boldsymbol{e}_1 + b_2\boldsymbol{e}_2) \tag{9}$$

for all $b_2 \geq 0$. This inequality looks more general than (6), although they are obviously equivalent. It will be more convenient to use (9) in the subsequent results of this section as well as in the proof of Lemma 3 in the appendix. Now we can deduce the following property of an optimal policy.

Theorem 1. *Let $\boldsymbol{y}^* = \boldsymbol{y}^n(\boldsymbol{x}; \boldsymbol{j})$. Then, for $i, k = 1, 2$ with $k \neq i$, either*

$$y_k^* = \min\{c - y_i^*, x_k + j_k\}$$

or it is never optimal to accept b_k class-k jobs in states $\boldsymbol{y}^* + b_i\boldsymbol{e}_i$, where $b_i \geq 0$ and

$$b_k = 1, \dots, \min\{c - y_i^* - b_i, x_k + j_k\} - y_k^*.$$

Proof. Let $\boldsymbol{y}^*$ be such that y_i^* is the optimal number of class-i jobs to have in the system immediately after state $(\boldsymbol{x}; \boldsymbol{j})$, when there are n more observation points remaining in the horizon. Then it is clear that having $y_k^* \leq \min\{c - y_i^*, x_k + j_k\}$ is feasible. The optimal policy may accept as many class-k jobs as possible, in which case $y_k^* = \min\{c - y_i^*, x_k + j_k\}$, one of the possible cases stated in the theorem. Thus, assume that this is not the case, i.e. $y_k^* < \min\{c - y_i^*, x_k + j_k\}$. Hence, the optimal policy could accept more class-k jobs – since a strictly positive number, $x_k + j_k - y_k^*$, of class-k jobs are rejected while there is a strictly positive number, $c - y_i^* - y_k^*$, of free servers – but chooses to reject them. Then

$$b_k r_k + u^n(\boldsymbol{y}^* + b_k\boldsymbol{e}_k) < u^n(\boldsymbol{y}^*) \quad \text{for } b_k = 1, \dots, \min\{c - y_i^*, x_k + j_k\} - y_k^*,$$

from which we have

$$0 < u^n(\boldsymbol{y}^*) - u^n(\boldsymbol{y}^* + b_k\boldsymbol{e}_k) - b_k r_k \leq u^n(\boldsymbol{y}^* + b_i\boldsymbol{e}_i) - u^n(\boldsymbol{y}^* + b_i\boldsymbol{e}_i + b_k\boldsymbol{e}_k) - b_k r_k$$

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

TABLE 1: Optimal admission policy for Example 3.

	x_1				
x_2	0	1	2	3	4
0	30, 01, 13, 05	30, 11, 13, 14	30, 21, 23, 23	30, 31, 32, 32	40, 41, 41, 41
1	21, 02, 04, 05	21, 12, 14, 14	21, 22, 23, 23	31, 32, 32, 32	
2	12, 03, 05, 05	12, 13, 14, 14	22, 23, 23, 23		
3	13, 04, 05, 05	13, 14, 14, 14			
4	04, 05, 05, 05				

for $b_k = 1, \dots, \min\{c - y_i^*, x_k + j_k\} - y_k^*$ and all $b_i \geq 0$ with $\boldsymbol{y}^* + b_i\boldsymbol{e}_i + b_k\boldsymbol{e}_k \in \mathcal{S}$, where the second inequality follows from (9) with $\boldsymbol{x} = \boldsymbol{y}^*$. Hence, when $y_k^* < \min\{c - y_i^*, x_k + j_k\}$ it is not optimal to accept b_k class-k jobs in states $\boldsymbol{y}^* + b_i\boldsymbol{e}_i$, where $b_i \geq 0$ and

$$b_k = 1, \dots, \min\{c - y_i^* - b_i, x_k + j_k\} - y_k^*.$$

Note that we must have $b_k \leq \min\{c - y_i^* - b_i, x_k + j_k\} - y_k^*$, to ensure that $\boldsymbol{y}^* + b_i\boldsymbol{e}_i + b_k\boldsymbol{e}_k \in \mathcal{S}$, and $b_k \geq 1$ for $b_i = 0$.

This theorem specifies a relation between y_1^* and y_2^*, rather than specifying y_1^* or y_2^* or both. In two cases, we already know one component of $\boldsymbol{y}^*$: when batches consist of only class i we have $y_k^* = x_k$, and when class i is known to be a preferred class we have $y_i^* = \min\{c - x_k, x_i + j_i\}$. However, the remaining component of $\boldsymbol{y}^*$ still cannot be determined. Hence, Theorem 1 can be used only to restrict the space for the optimal actions of *other* states. We illustrate this in the following example.

Example 3. We consider a system with five servers over an infinite horizon, and set $\mu_1 = 2$, $\mu_2 = 3$, $r_1 = 1$, $r_2 = \frac{204}{220}$, $\beta = 1$, $\lambda = 30$, $p_{50} = \frac{1}{6}$, $p_{01} = \frac{1}{3}$, $p_{23} = \frac{29}{60}$, and $p_{55} = \frac{1}{60}$. In Table 1, we present the optimal policy for this example. Each entry, which corresponds to a state $\boldsymbol{x}$, is divided into four, where the first part is $\boldsymbol{y}(\boldsymbol{x}; (5, 0))$, the second $\boldsymbol{y}(\boldsymbol{x}; (0, 1))$, the third $\boldsymbol{y}(\boldsymbol{x}; (2, 3))$, and the fourth $\boldsymbol{y}(\boldsymbol{x}; (5, 5))$. For example, in state $(1, 2)$ it is optimal to reject all class-1 jobs and to accept as many class-2 jobs as possible from all kinds of batches.

We first observe several characteristics of the optimal policy. Class 2 is preferred, although the existence of a preferred class is not guaranteed as the parameters of the system satisfy (5). Moreover, the optimal policy is not a trunk reservation policy, i.e. a policy determined by trunk reservation parameters, t_1 and t_2, such that an additional class-i job is rejected if $x_1 + x_2 + 1 > c - t_i$ and accepted otherwise. In this example, an additional class-1 job is accepted in state $(0, 3)$ since $\boldsymbol{y}((0, 3); (5, 0)) = (1, 3)$, and rejected in state $(3, 0)$ as $\boldsymbol{y}((3, 0); (5, 0)) = (3, 0)$.

To illustrate the use of Theorem 1, consider the state $\boldsymbol{x} = (1, 1)$, and let $\boldsymbol{y}^* = \boldsymbol{y}(\boldsymbol{x}; \boldsymbol{j})$. For $\boldsymbol{j} = (5, 0)$ we have $\boldsymbol{y}^* = (2, 1)$, meaning that $y_1^* = 2 < \min\{c - y_2^*, x_1 + j_1\} = 4$. Then, we can deduce that it is neither optimal to accept one or two class-1 jobs in $\boldsymbol{x} = (2, 1)$, nor to accept one class-1 job in $\boldsymbol{x} = (2, 2)$, in agreement with Table 1. For $\boldsymbol{j} = (2, 3)$ we have $\boldsymbol{y}^* = (1, 4)$. Therefore, $y_k^* = \min\{c - y_i^*, x_k + j_k\}$ for $k = 1, 2$, which gives no further information about other states.

As this example shows, Theorem 1 does not yield a precise description of the structure of the optimal policy. For stochastic knapsacks, submodularity alone guarantees the existence of an optimal threshold policy. In systems with batch arrivals, on the other hand, submodularity *and* concavity of the value functions together can establish the existence of an optimal threshold

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

policy. However, as discussed above, concavity has not been proven for all values of the parameters even in unit-sized stochastic knapsacks.

5. Conclusions and possible extensions

In this paper, we have addressed the problem of dynamic admission control in a two-class Markovian loss system receiving batch arrivals, where classes have different service rates. We assumed that batches can be partially accepted, as Örmeci and Burnetas [13] have shown that the value functions corresponding to batch acceptance do not satisfy any type of monotonicity. Each batch arriving at the system may consist of either a single class of job or several classes. We have differentiated between these two systems in all the issues regarding the structure of optimal policies. The structure of optimal admission policies was examined with respect to two characteristics: the existence of a preferred class and the submodularity of the value functions and its implications on the optimal policy. We derived two sets of conditions sufficient for there to be a preferred class, one for single-class batches and one for mixed batches. The submodularity of the value functions will be established in the following appendix. In batch systems, submodularity can only decrease the number of states that can be optimal, rather than specify an optimal threshold policy.

As a result, various aspects of dynamic admission control of systems with batch arrivals have been analyzed. Natural extensions of this work include the cases of general (rather than exponential) interarrival times, and random (instead of fixed) rewards. The admission of jobs to the system can also be controlled via pricing. Thus, the controller of the system can propose a price for which the system is willing to serve the incoming jobs. This kind of control has been considered in the context of social optimization for different queueing systems; see, e.g. [12], [9], and [23]. Note that either dynamic or static pricing may be considered, respectively depending on whether or not the prices depend on the state of the system.

Appendix A.

To prove Lemma 3, we must consider all possible actions in four different states: $\boldsymbol{x}, \boldsymbol{x}+\boldsymbol{e}_1$, $\boldsymbol{x}+\boldsymbol{e}_2$, and $\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2$. Thus, it is essential to specify the sets of states reachable from each of these states, and the relations within these sets, when a batch $\boldsymbol{j}$ arrives at the system. Let $\boldsymbol{x}^1=\boldsymbol{x}+\boldsymbol{e}_1, \boldsymbol{x}^2=\boldsymbol{x}+\boldsymbol{e}_2$, and $\bar{\boldsymbol{x}}=\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2$. Then we define the following sets for a given $(\boldsymbol{x};\boldsymbol{j})$ with $x_1+x_2+2\le c$:

$$
\begin{aligned}
S^0 &= S(\boldsymbol{x};\boldsymbol{j}) \cap S(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2;\boldsymbol{j}),\\
S^x_\bullet &= \{\boldsymbol{x}\},\\
S^x_{x^1} &= \{(y_1,x_2)\colon x_1<y_1\le x_1+j_1\}\cap\mathcal{S},\\
S^x_{x^2} &= \{(x_1,y_2)\colon x_2<y_2\le x_2+j_2\}\cap\mathcal{S},\\
S^x &= S^x_\bullet \cup S^x_{x^1} \cup S^x_{x^2},\\
S^{\bar{x}}_\bullet &= \{(x_1+j_1+1,x_2+j_2+1)\}\cap\mathcal{S},\\
S^{\bar{x}}_{x^1} &= \{(x_1+j_1+1,y_2): x_2+1\le y_2<x_2+j_2+1\}\cap\mathcal{S},\\
S^{\bar{x}}_{x^2} &= \{(y_1,x_2+j_2+1): x_1+1\le y_1<x_1+j_1+1\}\cap\mathcal{S},\\
S^{\bar{x}} &= S^{\bar{x}}_\bullet \cup S^{\bar{x}}_{x^1} \cup S^{\bar{x}}_{x^2}.
\end{aligned}
$$

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

In words, S^0 is the set of states reachable from both $\boldsymbol{x}$ and $\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2$, and, so, from $\boldsymbol{x}+\boldsymbol{e}_1$ and $\boldsymbol{x}+\boldsymbol{e}_2$ also. S^x is the set of states reachable from $\boldsymbol{x}$ but not from $\bar{\boldsymbol{x}}$, whereas $S^{\bar{x}}$ is the set of states reachable from $\bar{\boldsymbol{x}}$ but not from $\boldsymbol{x}$. $S^x_\bullet$ is a singleton and it is reachable only from $\boldsymbol{x}$, whereas $S^{\bar{x}}_\bullet$ is either a singleton or the empty set, and is reachable only from $\bar{\boldsymbol{x}}$. The sets $S^x_{x^i}$, $i=1,2$, are reachable from $\boldsymbol{x}$ and $\boldsymbol{x}^i$, but not from $\bar{\boldsymbol{x}}$ or $\boldsymbol{x}^k$, $k\neq i$; similarly, the sets $S^{\bar{x}}_{x^i}$, $i=1,2$, are reachable from $\bar{\boldsymbol{x}}$ and $\boldsymbol{x}^i$, but not from $\boldsymbol{x}$ or $\boldsymbol{x}^k$, $k\neq i$. Also, note that the set $S^{\bar{x}}_{x^i}$ is empty if $x_1+x_2+2+j_i>c$. Lemma 4 summarizes all useful (and also obvious) relations among these sets, and is presented without proof since all the relations are very easy to verify.

Lemma 4. *The following relations hold among the sets defined above.*

(i) $S^0=S(\boldsymbol{x}+\boldsymbol{e}_1;\boldsymbol{j})\cap S(\boldsymbol{x}+\boldsymbol{e}_2;\boldsymbol{j})$.

(ii) $S^x=S(\boldsymbol{x};\boldsymbol{j})\setminus S(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2;\boldsymbol{j})$.

(iii) $S^{\bar{x}}=S(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2;\boldsymbol{j})\setminus S(\boldsymbol{x};\boldsymbol{j})$.

(iv) $S^x_{x^1}\subseteq S(\boldsymbol{x}+\boldsymbol{e}_1;\boldsymbol{j})\cap S(\boldsymbol{x};\boldsymbol{j})$.

(v) $S^x_{x^1}\cap S(\boldsymbol{x}+\boldsymbol{e}_2;\boldsymbol{j})=S^x_{x^1}\cap S(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2;\boldsymbol{j})=\varnothing$.

(vi) $S^x_{x^2}\subseteq S(\boldsymbol{x}+\boldsymbol{e}_2;\boldsymbol{j})\cap S(\boldsymbol{x};\boldsymbol{j})$.

(vii) $S^x_{x^2}\cap S(\boldsymbol{x}+\boldsymbol{e}_1;\boldsymbol{j})=S^x_{x^2}\cap S(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2;\boldsymbol{j})=\varnothing$.

(viii) $S^{\bar{x}}_{x^1}\subseteq S(\boldsymbol{x}+\boldsymbol{e}_1;\boldsymbol{j})\cap S(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2;\boldsymbol{j})$.

(ix) $S^{\bar{x}}_{x^1}\cap S(\boldsymbol{x}+\boldsymbol{e}_2;\boldsymbol{j})=S^{\bar{x}}_{x^1}\cap S(\boldsymbol{x};\boldsymbol{j})=\varnothing$.

(x) $S^{\bar{x}}_{x^2}\subseteq S(\boldsymbol{x}+\boldsymbol{e}_2;\boldsymbol{j})\cap S(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2;\boldsymbol{j})$.

(xi) $S^{\bar{x}}_{x^2}\cap S(\boldsymbol{x}+\boldsymbol{e}_1;\boldsymbol{j})=S^{\bar{x}}_{x^2}\cap S(\boldsymbol{x};\boldsymbol{j})=\varnothing$.

We can now present the proof of Lemma 3.

Proof of Lemma 3. Setting $u^0(\boldsymbol{x})=0$ for all $\boldsymbol{x}$ satisfies (6). Assume that the statement is true for a general n. Then (9) holds, as shown in Section 4. We first show that the v^n satisfy (6). To do so, we define δ^n as

$$\delta^n=v^n(\boldsymbol{x};\boldsymbol{j})-v^n(\boldsymbol{x}+\boldsymbol{e}_2;\boldsymbol{j})-v^n(\boldsymbol{x}+\boldsymbol{e}_1;\boldsymbol{j})+v^n(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2;\boldsymbol{j}),$$

and will show that $\delta^n\leq 0$ for all possible actions.

Let $\boldsymbol{y}^*=(y_1^*,y_2^*)=\boldsymbol{y}^n(\boldsymbol{x};\boldsymbol{j})$ and

$$\boldsymbol{y}^{*\prime}=(y_1^{*\prime},y_2^{*\prime})=\boldsymbol{y}^n(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2;\boldsymbol{j}),$$

meaning that $\boldsymbol{y}^*$ and $\boldsymbol{y}^{*\prime}$ are the optimal states to go to from states $(\boldsymbol{x};\boldsymbol{j})$ and $(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2;\boldsymbol{j})$, respectively. Now we distinguish four cases according to the actions taken.

Case I: $\boldsymbol{y}^*\in S^0$ and $\boldsymbol{y}^{*\prime}\in S^0$. In this case, both $\boldsymbol{y}^*$ and $\boldsymbol{y}^{*\prime}$ are reachable from states $\boldsymbol{x}+\boldsymbol{e}_1$ and $\boldsymbol{x}+\boldsymbol{e}_2$, since $S^0=S(\boldsymbol{x}+\boldsymbol{e}_1;\boldsymbol{j})\cap S(\boldsymbol{x}+\boldsymbol{e}_2;\boldsymbol{j})$ by Lemma 4(i). Thus, we let one of the latter states go to state $\boldsymbol{y}^*$ and the other one to $\boldsymbol{y}^{*\prime}$. Then the difference of immediate rewards in δ_n is 0, and we have

$$\delta^n\leq u^n(\boldsymbol{y}^*)-u^n(\boldsymbol{y}^*)-u^n(\boldsymbol{y}^{*\prime})+u^n(\boldsymbol{y}^{*\prime})=0,$$

since the v^n are optimal.

https://doi.org/10.1239/aap/1134587747 Published online by Cambridge University Press

Case II: $\boldsymbol{y}^* \in S^0$ and $\boldsymbol{y}^{*\prime} \in S^{\bar{\boldsymbol{x}}}$.

Case II.1: $\boldsymbol{y}^{*\prime} \in S^{\bar{\boldsymbol{x}}}_{x1}$. Since $S^0 = S(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j}) \cap S(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j})$ by Lemma 4(i), $\boldsymbol{y}^*$ is a feasible action in state $(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j})$ and, because $S^{\bar{\boldsymbol{x}}}_{x1} \subseteq S(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j})$ by Lemma 4(viii), $\boldsymbol{y}^{*\prime}$ is reachable from state $(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j})$. Thus,

$$\delta^n \le u^n(\boldsymbol{y}^*) - u^n(\boldsymbol{y}^*) - u^n(\boldsymbol{y}^{*\prime}) + u^n(\boldsymbol{y}^{*\prime}) = 0.$$

Case II.2: $\boldsymbol{y}^{*\prime} \in S^{\bar{\boldsymbol{x}}}_{x2}$. This is very similar to case II.1. We only need to observe that $\boldsymbol{y}^{*\prime}$ is feasible for state $(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j})$, since $S^{\bar{\boldsymbol{x}}}_{x2} \subseteq S(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j})$ by Lemma 4(x), and that $\boldsymbol{y}^*$ is reachable from $(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j})$, since $S^0 \subseteq S(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j})$ by Lemma 4(i). It then follows that $\delta^n \le 0$.

Case II.3: $\boldsymbol{y}^{*\prime} \in S^2_{\bullet}$. Since $\boldsymbol{y}^{*\prime} = (x_1 + j_1 + 1, x_2 + j_2 + 1)$, it cannot be reached from either $(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j})$ or $(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j})$. Therefore, we cannot use the same technique as above. Instead, we first observe that the state $(y_1^*, y_2^{*\prime})$ is reachable from the state $(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j})$ and that $(y_1^{*\prime}, y_2^*)$ is reachable from $(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j})$. Also notice that both $(y_1^*, y_2^{*\prime}) \in \mathcal{S}$ and $(y_1^{*\prime}, y_2^*) \in \mathcal{S}$, since $y_1^* < y_1^{*\prime}$ and $y_2^* < y_2^{*\prime}$ due to the assumptions in this case. Thus,

$$\delta^n \le u^n(y_1^*, y_2^*) - u^n(y_1^{*\prime}, y_2^*) - u^n(y_1^*, y_2^{*\prime}) + u^n(y_1^{*\prime}, y_2^{*\prime}) \le 0,$$

where the first inequality is due to the optimality of the v^n and the feasibility of the actions $(y_1^*, y_2^{*\prime})$ and $(y_1^{*\prime}, y_2^*)$, and the second is due to (9) with $\boldsymbol{x} = \boldsymbol{y}^*$, $b_1 = y_1^{*\prime} - y_1^*$, and $b_2 = y_2^{*\prime} - y_2^*$.

Case III: $\boldsymbol{y}^* \in S^x$ and $\boldsymbol{y}^{*\prime} \in S^0$. Here we must consider three subcases, $\boldsymbol{y}^* \in S^x_{x1}$, $\boldsymbol{y}^* \in S^x_{x2}$, and $\boldsymbol{y}^* = \boldsymbol{x}$, each of which is similar to the corresponding subcase of case II; hence, the details are omitted.

Case IV: $\boldsymbol{y}^* \in S^x$ and $\boldsymbol{y}^{*\prime} \in S^{\bar{\boldsymbol{x}}}$. In case, we have nine subcases, most of which are proven similarly. We consider two cases in detail and only mention the similarities of the others.

Let $\boldsymbol{y}^* \in S^x_{x1}$ and $\boldsymbol{y}^{*\prime} \in S^{\bar{\boldsymbol{x}}}_{x1}$. Then $\boldsymbol{y}^*$ and $\boldsymbol{y}^{*\prime}$ are both reachable from $(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j})$ but not from $(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j})$. Now observe that $(y_1^*, y_2^{*\prime})$ is reachable for $(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j})$ and $(y_1^{*\prime}, y_2^*)$ is reachable for $(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j})$. Moreover, $(y_1^*, y_2^{*\prime}) \in \mathcal{S}$ and $(y_1^{*\prime}, y_2^*) \in \mathcal{S}$: since $\boldsymbol{y}^* \in S^x_{x1}$, we have $x_1 < y_1^* \le \min\{x_1 + j_1, c - x_1\}$ and $y_2^* = x_2$, and, because $\boldsymbol{y}^{*\prime} \in S^{\bar{\boldsymbol{x}}}_{x1}$, we have $y_1^{*\prime} = x_1 + j_1 + 1$ and $x_2 + 1 \le y_2^{*\prime} < x_2 + j_2 + 1$, meaning that $y_k^* \le y_k^{*\prime}$ for $k = 1, 2$. Thus,

$$\delta^n \le u^n(y_1^*, y_2^*) - u^n(y_1^{*\prime}, y_2^*) - u^n(y_1^*, y_2^{*\prime}) + u^n(y_1^{*\prime}, y_2^{*\prime}) \le 0,$$

by (9) with $\boldsymbol{x} = \boldsymbol{y}^*$, $b_1 = y_1^{*\prime} - y_1^*$, and $b_2 = y_2^{*\prime} - y_2^*$, as in case II.3.

In the following cases, we let $(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j})$ move to $(y_1^*, y_2^{*\prime})$ and $(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j})$ to $(y_1^{*\prime}, y_2^*)$:

(i) $\boldsymbol{y}^* \in S^x_{x1}$ and $\boldsymbol{y}^{*\prime} \in S^{\bar{\boldsymbol{x}}}_{\bullet}$,

(ii) $\boldsymbol{y}^* \in S^x_{x2}$ and $\boldsymbol{y}^{*\prime} \in S^{\bar{\boldsymbol{x}}}_{x2}$,

(iii) $\boldsymbol{y}^* \in S^x_{x2}$ and $\boldsymbol{y}^{*\prime} \in S^{\bar{\boldsymbol{x}}}_{\bullet}$,

(iv) $\boldsymbol{y}^* \in S^x_{\bullet}$ and $\boldsymbol{y}^{*\prime} \in S^{\bar{\boldsymbol{x}}}_{x1}$,

(v) $\boldsymbol{y}^* \in S^x_{\bullet}$ and $\boldsymbol{y}^{*\prime} \in S^{\bar{\boldsymbol{x}}}_{x2}$,

(vi) $\boldsymbol{y}^* \in S^x_{\bullet}$ and $\boldsymbol{y}^{*\prime} \in S^{\bar{\boldsymbol{x}}}_{\bullet}$.

Notice that in all these cases either $\boldsymbol{y}^*$ or $\boldsymbol{y}^{*\prime}$ is not reachable from either $(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j})$ or $(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j})$, or both states can be reached from only one of them. Thus, we cannot cancel out the terms; instead, we have to let one of these states move to $(y_1^*, y_2^{*\prime})$ and the other one to $(y_1^{*\prime}, y_2^*)$. Moreover, it can be easily verified that $y_k^* \le y_k^{*\prime}$ for $k = 1, 2$ in all these cases, meaning that $(y_1^*, y_2^{*\prime}) \in \mathcal{S}$ and $(y_1^{*\prime}, y_2^*) \in \mathcal{S}$.

If $\boldsymbol{y}^* \in S_{x^1}^{x}$ and $\boldsymbol{y}^{*\prime} \in S_{x^2}^{\bar{x}}$, then $\boldsymbol{y}^*$ is reachable from $(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j})$ and $\boldsymbol{y}^{*\prime}$ is reachable from $(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j})$. Then by the optimality of the v^n,

$$\delta^n \le u^n(y_1^*, y_2^*) - u^n(y_1^*, y_2^*) - u^n(y_1^{*\prime}, y_2^{*\prime}) + u^n(y_1^{*\prime}, y_2^{*\prime}) = 0.$$

Similarly, when $\boldsymbol{y}^* \in S_{x^2}^{x}$ and $\boldsymbol{y}^{*\prime} \in S_{x^1}^{\bar{x}}$, $\boldsymbol{y}^*$ is reachable from $(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j})$ and $\boldsymbol{y}^{*\prime}$ is reachable from $(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j})$.

Thus, for all possible cases we have $\delta^n \le 0$.

We now consider u^{n+1}. There are four systems starting from four different states: $\boldsymbol{x}, \boldsymbol{x}+\boldsymbol{e}_2$, $\boldsymbol{x}+\boldsymbol{e}_1$, and $\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2$. We couple all these systems so that, except for the additional customers, they all behave in the same way. Moreover, we couple the class-1 customers in systems starting in states $\boldsymbol{x}+\boldsymbol{e}_1$ and $\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2$, and we couple the class-2 customers in systems starting in states $\boldsymbol{x}+\boldsymbol{e}_2$ and $\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2$. Then, if the additional class-i job departs from the system starting in state $\boldsymbol{x}+\boldsymbol{e}_i$, the additional class-$i$ job in $\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2$ also departs. Thus,

$$\begin{aligned}
&u^{n+1}(\boldsymbol{x}) - u^{n+1}(\boldsymbol{x}+\boldsymbol{e}_2) - u^{n+1}(\boldsymbol{x}+\boldsymbol{e}_1) + u^{n+1}(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2) \\
&\quad = \lambda \sum_{j_1, j_2} p_{j_1, j_2}[v^n(\boldsymbol{x}; \boldsymbol{j}) - v^n(\boldsymbol{x}+\boldsymbol{e}_2; \boldsymbol{j}) - v^n(\boldsymbol{x}+\boldsymbol{e}_1; \boldsymbol{j}) + v^n(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2; \boldsymbol{j})] \\
&\qquad + x_1\mu_1[u^n(\boldsymbol{x}-\boldsymbol{e}_1) - u^n(\boldsymbol{x}-\boldsymbol{e}_1+\boldsymbol{e}_2) - u^n(\boldsymbol{x}) + u^n(\boldsymbol{x}+\boldsymbol{e}_2)] \\
&\qquad + \mu_1[u^n(\boldsymbol{x}) - u^n(\boldsymbol{x}+\boldsymbol{e}_2) - u^n(\boldsymbol{x}) + u^n(\boldsymbol{x}+\boldsymbol{e}_2)] \\
&\qquad + x_2\mu_2[u^n(\boldsymbol{x}-\boldsymbol{e}_2) - u^n(\boldsymbol{x}) - u^n(\boldsymbol{x}+\boldsymbol{e}_1-\boldsymbol{e}_2) + u^n(\boldsymbol{x}+\boldsymbol{e}_1)] \\
&\qquad + \mu_2[u^n(\boldsymbol{x}) - u^n(\boldsymbol{x}) - u^n(\boldsymbol{x}+\boldsymbol{e}_1) + u^n(\boldsymbol{x}+\boldsymbol{e}_1)] \\
&\qquad + \alpha[u^n(\boldsymbol{x}) - u^n(\boldsymbol{x}+\boldsymbol{e}_2) - u^n(\boldsymbol{x}+\boldsymbol{e}_1) + u^n(\boldsymbol{x}+\boldsymbol{e}_1+\boldsymbol{e}_2)] \\
&\quad \le 0,
\end{aligned}$$

where $\alpha = c\mu_2 - (x_1+1)\mu_1 - (x_2+1)\mu_2$. The terms in the summation are less than or equal to 0 since the v^n are shown to satisfy the inequality, the μ_1 and μ_2 terms are 0, and all other terms are nonpositive by the induction hypothesis.

Thus, the value functions u^n satisfy (6) for all n whenever u^0 does. This completes the proof.

References

[1] Altman, E. (2002). Applications of Markov decision processes in communication networks. In *Handbook of Markov Decision Processes*, eds E. A. Feinberg and A. Shwartz, Kluwer, Boston, MA, pp. 489–536.

[2] Altman, E., Jiménez, T. and Koole, G. M. (2001). On optimal call admission control in a resource-sharing system. *IEEE Trans. Commun* **49,** 1659–1668.

[3] Carrizosa, E., Conde, E. and Munoz-Marquez, M. (1998). Admission policies in loss queueing models with heterogeneous arrivals. *Manag. Sci.* **44,** 311–320.

[4] Carrol, W. and Grimes, R. (1995). Evolutionary change in product management. Experiences in the car rental industry. *Interfaces* **25,** 84–104.

[5] Geraghty, M. K. and Johnson, E. (1997). Revenue management saves national car rental. *Interfaces* **27,** 107–127.

[6] Harrison, J. M. (1975). Dynamic scheduling of a multiclass queue. Discount optimality. *Operat. Res.* **23,** 270–282.

[7] Lippman, S. A. (1975). Applying a new device in the optimization of exponential queueing systems. *Operat. Res.* **23,** 687–710.
[8] Lippman, S. A. and Ross, S. M. (1971). The streetwalker's dilemma. A job shop model. *SIAM J. Appl. Math.* **20,** 336–342.
[9] Lippman, S. A. and Stidham, S. M. (1977). Individual versus social optimization in exponential congestion systems. *Operat. Res.* **25,** 233–247.
[10] Liu, Z., Nain, P. and Towsley, D. (1995). Sample path methods in the control of queues. *Queueing Systems* **21,** 293–336.
[11] Miller, B. (1971). A queuing reward system with several customer classes. *Manag. Sci.* **16,** 234–245.
[12] Naor, P. (1969). On the regulation of queue size by levying tolls. *Econometrica* **37,** 15–24.
[13] Örmeci, E. L. and Burnetas, A. N. (2004). Admission control with batch arrivals. *Operat. Res. Lett.* **32,** 448–454.
[14] Örmeci, E. L., Burnetas, A. N. and Emmons, H. (2002). Dynamic policies of admission to a two-class system based on customer offers. *IIE Trans.* **34,** 813–822.
[15] Örmeci, E. L., Burnetas, A. N. and van der Wal, J. (2001). Admission policies for a two class loss system. *Stoch. Models* **17,** 513–540.
[16] Puhalskii, A. A. and Reiman, M. I. (1998). A critically loaded multirate link with trunk reservation. *Queueing Systems* **28,** 157–190.
[17] Puterman, M. (1994). *Markov Decision Processes*. John Wiley, New York.
[18] Righter, R. (1994). Scheduling. In *Stochastic Orders and Their Applications*, eds M. Shaked and J. G. Shanthikumar, Academic Press, Boston, MA, pp. 381–432.
[19] Ross, K. W. (1995). *Multiservice Loss Models for Broadband Telecommunication Networks*. Springer, London.
[20] Ross, S. M. (1970). *Applied Probability Models with Optimization Applications*. Dover, New York.
[21] Savin, S., Cohen, M., Gans, N. and Katalan, Z. (2000). Capacity management in rental businesses with heterogeneous customer bases. *Operat. Res.* **53,** 617–631.
[22] Walrand, J. (1988). *Introduction to Queueing Networks*. Prentice Hall, Englewood Cliffs, NJ.
[23] Xu, S. H. and Shanthikumar, J. G. (1993). Optimal expulsion control—a dual approach to admission control of an ordered-entry system. *Operat. Res.* **41,** 1137–1152.