Optimal control of a head-of-line processor sharing model with regular and opportunity customers

A.C.C. van Wijk*

Eindhoven University of Technology, Eindhoven, The Netherlands

June 17, 2011

Abstract

Motivated by a workload control setting, we study a model where two types of customers are served by a single server according to the head-of-line processor sharing discipline. Regular customers and opportunity customers are arriving to the system according to two independent Poisson processes, each requiring an exponentially distributed service time. The regular customers will queue, incurring some holding costs. On contrary, an opportunity customer has to be taken into service directly, or is lost otherwise. There can be at most one opportunity customer in the system. The server can work on both one regular and one opportunity customer at the same time, where one can decide on how the server speed is split out. Moreover, one can decide whether to accept or reject an opportunity customer, incurring penalty costs for the latter. In this way, one has partial control about the workload in the system. We formulate the model as a Markov decision problem. We prove that the optimal policy, minimizing the expected discounted long-run cost, has a monotone structure in the number of regular customers. That is, the optimal policy for accepting an opportunity customer is described be a threshold, and the fraction of the server's attention devoted to the opportunity customer is a monotone decreasing function. Further, we generalize our model to the case where opportunity customers will queue as well, and to the case where also regular customers can be rejected.

Keywords:head-of-line processor sharing; optimal control; dynamic programming; multimodularity; threshold type policy.

* *Corresponding author:* P.O. Box 513, 5600MB Eindhoven, The Netherlands, a.c.c.v.wijk@tue.nl

1 Introduction

Motived by a workload control setting, we study a model where two types of customers are served by a single server. These two types are *regular* customers and *opportunity* customers. The regular customers are willing to wait before taken into service, whereas the opportunity customers have to be taken into service directly, or are lost otherwise. We allow at most one such a customer in the system. When, for example, the workload of the regular customers is low, an opportunity customer provides the option for some extra revenue. Hence, we have control over the workload by deciding whether to take such a customer into service.

We model this problem as a single server queueing model servicing the two types of customers. The regular customers form a queue upon arrival to the system. The server can work on both one regular and one opportunity customer at the same time. The (total) service rate of the server is given and fixed, but one can adjust how it is split out among these two customers. This is known as the head-of-line processor sharing discipline, with adjustable service rate. Moreover, one can decide whether to accept or reject an opportunity customer, incurring penalty costs for the latter. Hence, we have to balance the service speed and acceptance decision. If the service rate of the opportunity customer is set too low, we face the risk of the next opportunity customer already arriving before service completion (having to reject it). On the other hand, a higher service speed will let all regular customers queue for a longer time, incurring longer and higher holding costs.

We formulate the model as Markov decision problem (see Puterman [8]). Based on the number of regular customers in the system, a decision has to be taken whether to accept or reject an arriving opportunity customer. Moreover, one has to decide the service speed for an opportunity customer. We use event-based dynamic programming (cf. Koole [4, 6]) to write the n-period minimal cost function (the *value function*) in so-called event operators, where the possible events are customer arrivals and service completions. We prove that the event operators preserve certain structural properties, such as convexity and supermodularity, and hence, by induction, the value function satisfies these properties. From that, the structure of the optimal policy follows (i.e. threshold policy for admitting an opportunity customer, and a monotonicity result for the server rate assigned to the opportunity customer).

The optimal control of a head-of-line processor sharing is to the best of our knowledge an open problem. We mention the following articles which are in some way related to our research.

Konheim et al. [3] give a complete analysis of a system with two parallel queueing lines, served by a single server, but assume that each is served with half of the service rate. Fayolle et al. [1] study a more general framework, where the fraction of the attention to each queue is flexible, but they assume that from a given number of customers on, the service rates are independent. Wasserman and Bambos [12] study the dynamic allocation of a single server to parallel queues with finite-capacity buffers, characterizing the allocation policy that stochastically minimizes the number of customers lost due to buffer overflows. A similar problem is studied in Towsley et al. [10]. Stidham [9] focuses on the optimal control of admission to queueing systems, and uses dynamic-programming to show that an optimal control is monotonic or characterized by one or more critical numbers. Weber and Stidham [13] study the optimal control of service rates in a network of queues. The optimal control of limited processor sharing is studied in Van der Weij et al. [11]. They dynamically adjust the number of servers in a queue with processor sharing, where every customer in service receives a proportional fraction of the processing time. They use the same kind of techniques and derive the same kind of results as we do, namely monotonicity properties and optimal dynamic policies using dynamic programming. We contribute to the literature by deriving monotonicity properties for the optimal control of a two queue head-of-line processor sharing model. Hence, we obtain optimal dynamic policies. The outline is as follows. We start by introducing the model and notation used in Section 2. We describe the problem in more detail and describe the dynamic programming formulation. Then, in Section 3 we introduce the structural properties needed, show that the event operators preserve them, and hence show that the value function satisfies them. This leads to the optimal policy structure. Furthermore, we show some examples. In Section 4 we consider two model generalizations, which we show to fit in the same framework. Finally, we conclude in Section 5. All proofs are in the Appendix.

2 Model and Notation

In this section we described the model in more detail and introduce the notation used. We then formulate the model as a *Markov decision problem* (MDP). For this we introduce the *value function*, which can be recursively expressed using so-called *event operators.*

2.1 Problem Description

We consider the following queuing model, with two types of customers served by a single server. Firstly, regular customers arrive according to a Poisson process with rate λ_{reg}, and form a queue. Secondly, opportunity customers arrive according to a Poisson process with rate λ_{opp}. An opportunity customers has to be taken into service directly, or is lost otherwise, at penalty cost C_{opp}. Holding costs are charged for both regular and opportunity customers in the system: $h_{reg}(\cdot)$ and $h_{opp}(\cdot)$ respectively, which we assume both to be increasing and convex. The holding costs for an opportunity customer in service prevents the model from choosing a very low service rate.

Both queues are served by a single server, which applies the head-of-line processor sharing strategy with adjustable weights. That is, the server can simultaneously serve an opportunity customer and the first in line regular customer. The total service rate of the server is fixed, say $\bar{\mu}$, but it can be decided how this is split out between both customer types: with rate $0 < \mu \leq \bar{\mu}$ the opportunity customer is served, leaving rate $\bar{\mu} - \mu$ for the regular customer. Here μ is a decision variable, where we assume $\mu = 0$ when $x = 0$. For generality, we charge costs $c(\mu)$ when rate μ is chosen, assuming $c(0) = \min_{\mu \in [0,1]} c(\mu)$. The service times of both opportunity and one regular customers are exponentially distributed with mean 1. We assume all processes to be mutually independent. Furthermore, we assume that the system is stable: $\lambda_{reg}/\bar{\mu} < 1$.

2.2 Dynamic Programming Formulation

Denote by $x \in \{0,1\}$ the number of opportunity customers in the system, and by $y \in \mathbb{N} \cup \{0\}$ the number of regular customers in the system. Then the state is given by $(x, y) \in \mathcal{S} = \{(x, y) \mid x \in \{0,1\},\ y \in \mathbb{N} \cup \{0\}\}$, denoting by $\mathcal{S}$ the state space.

As the interarrival times of demands as well as the replenishment times are independent exponentially distributed random variables, we can apply uniformization (cf. [7]) to convert the semi-Markov decision problem into an equivalent Markov decision problem (MDP). The existence of a stationary optimal policy is guaranteed by Theorem 11.5.3 of [8].

When facing a decision, we should take into account the direct costs for a decision as well as the future expected costs this decision brings along. For the expected costs from a state, we introduce the *value function* (see [8]) $V_n : \mathcal{S} \mapsto \mathbb{R}^+$. $V_n(x, y)$ is the minimum expected total costs when there are n events (customer arrivals or service completions) left starting in state $(x, y) \in \mathcal{S}$. This V_n can be recursively expressed:

$$\begin{aligned} V_{n+1}(x,y) = h_{opp}(x) + h_{reg}(y) + \frac{1}{\lambda_{opp} + \lambda_{reg} + \bar{\mu} + \alpha} \quad & \Big(\lambda_{reg} V_n(x, y+1) \\ + \lambda_{opp} \left\{ \begin{array}{ll} \min\{V_n(x+1,y), V_n(x,y) + C_{opp}\} & \text{if } x = 0 \\ V_n(x,y) + C_{opp} & \text{if } x = 1 \end{array} \right\} & \\ + \min_{\mu \in [0,\bar{\mu}]} \Big\{ c'(\mu) + \mu V_n((x-1)^+, y) + (\bar{\mu} - \mu) V_n(x, (y-1)^+) \Big\} \Big), & \end{aligned}$$

starting with $V_0 \equiv 0$, where $\alpha \in (0, \infty)$ is the discounting factor.

We now use the event operators, introduced in Koole [5] (see also Koole [6]), to rewrite the value function. The operator $T_{CA(1)}$ models the (controlled) arrivals of opportunity customers, and is defined by

$$T_{CA(1)} f(x,y) = \min\{V_n(x+1,y), V_n(x,y) + C_{opp}\}.$$

Here, one has the decision to either accept or reject the arriving customer. We also have the restriction $x \leq 1$, which can be achieved by setting

$$h_{opp}(x) = \begin{cases} h_{opp}(x) & \text{if } x \leq 1; \\ K\,x & \text{if } x > 1, \end{cases} \tag{1}$$

with K a very large constant (cf. [6, p.57]). Hence, when $x = 1$, the minimum will always be attained for $V_n(x,y) + C_{opp}$. Note that this $h_{opp}(\cdot)$ is still increasing and convex. The operator $T_{A(2)}$ models the (uncontrolled) arrivals of regular customers, and is defined by

$$T_{A(2)} f(x,y) = V_n(x, y+1).$$

The operator $\tilde{T}_{CTD(1)}$ models the service completions, and is defined by

$$\tilde{T}_{CTD(1)} f(x,y) = \min_{\mu \in [0,1]} \Big\{ c(\mu) + \mu V_n((x-1)^+, y) + (1-\mu) V_n(x, (y-1)^+) \Big\}, \tag{2}$$

where we use a tilde to distinguish it from $T_{CTD(1)}$ of [6], which is almost equal. Note that μ

is a decision variable here. Moreover, we have the costs operator T_{costs} defined by

$$T_{\text{costs}} f(x,y) = h_{opp}(x) + h_{reg}(y)$$

and the uniformization operator T_{unif} for this problem defined by

$$T_{\text{unif}}(f_1, f_2, f_3)(x,y) = \frac{1}{\lambda_{opp} + \lambda_{reg} + \bar{\mu} + \alpha}\Big(\lambda_{opp} f_1(x,y) + \lambda_{reg} f_2(x,y) + \bar{\mu} f_3(x,y)\Big).$$

Now, we can write the value function using event-operators as:

$$V_{n+1}(x,y) = T_{\text{costs}}\Big(T_{\text{unif}}\big(T_{CA(1)}V_n(x,y),\ T_{A(2)}V_n(x,y),\ \tilde{T}_{CTD(1)}V_n(x,y)\big)\Big).$$

3 Structural Results

In this section we prove our main result: the structure of the optimal policy. For this, we first introduce the structural property multi-modularity, and show which properties it is composed of. We then prove that the value function V_n satisfies multi-modularity by showing that each of the operators in V_n preserve this property. From this we derive the structure of the optimal policy, which is a threshold policy for accepting an opportunity customer, and a monotone deceasing function for the optimal server speed dedicated to the opportunity customer. We illustrate the policy by examples. All proofs are given in the appendix.

3.1 Properties of Operators and Value Function

Consider the following properties of a function f, defined for all x such that the states appearing in the right-hand and left-hand side of the inequalities exist in $\mathcal{S}$:

$$\text{Conv}(x): \quad f(x,y) + f(x+2,y) \geq 2f(x+1,y), \tag{3}$$

$$\text{Conv}(y): \quad f(x,y) + f(x,y+2) \geq 2f(x,y+1), \tag{4}$$

$$\text{Supermod}: \quad f(x,y) + f(x+1,y+1) \geq f(x+1,y) + f(x,y+1), \tag{5}$$

$$\text{SuperC}(x,y): \quad f(x+2,y) + f(x,y+1) \geq f(x+1,y) + f(x+1,y+1), \tag{6}$$

$$\text{SuperC}(y,x): \quad f(x,y+2) + f(x+1,y) \geq f(x,y+1) + f(x+1,y+1). \tag{7}$$

Conv(x) stands for convexity of f in x, that is, the difference $f(x, y) - f(x+1, y)$ is decreasing in x. Analogously, Conv(y) is convexity of f in y. Supermod is supermodularity, the definition of which is symmetric in x and y. SuperC(x, y) and SuperC(y, x) stands for superconvexity, adopting the terminology of [6]. Note that it is not symmetric in it arguments. It is a straightforward result that Supermod and SuperC(x, y) imply Conv(x), and Supermod and SuperC(y, x) imply Conv(y).

Multimodularity (MM) (introduced by Hajek [2]) is, for the case of a two-dimensional domain, equal to the combination of Supermod and both SuperC's:

$$\text{MM} = \text{Supermod} \cap \text{SuperC}(x, y) \cap \text{SuperC}(y, x). \tag{8}$$

Lemma 3.1. *All operators $T_{CA(1)}$, $T_{A(2)}$, $\tilde{T}_{CTD(1)}$, T_{unif}, and T_{costs} preserve MM.*

That is, if some function f is MM, then Tf is MM as well, where T is one of the mentioned operators. The proof of this lemma is in the appendix. By induction on n, the next result immediately follows from the lemma.

Corollary 3.2. *V_n is MM for all $n \geq 0$.*

We use this result to derive the structure of the optimal policy.

3.2 Structure of Optimal Policy

The next theorem states the optimal policy structure.

Theorem 3.3. *a) The optimal policy for admitting an opportunity customer is a threshold policy. That is, there exist a threshold, say $T \in \mathbb{N} \cup \{0\}$, such that the optimal decision is to accept the opportunity customer if $y \leq T$, and to reject it otherwise.*
b) The optimal server speed dedicated to the opportunity customer is a monotone deceasing function in x.

Here, decreasingness is understood to be non-strict. The optimal policy structure is in line with our intuition. When the workload of regular customers is low, one is more likely to accept an opportunity customers. Also, the more regular customers in the system, the larger the fraction of the servers attention assigned to these customers. As a consequence, the server speed for the opportunity customer is decreasing.

When $c(\mu) \equiv 0$, the optimal μ is always either 0 or 1. This follows directly from the fact that in (2) a linear function in minimized in this case. Hence, the optimal policy can as well be described be a single threshold, say M, such that the optimal rate is 1 when $y \leq M$, and 0 otherwise.

In the case that no holding costs are charged for an opportunity customer in service, i.e. when $h_{opp}(1) = 0$, the opportunity customer is always accepted. However, it might receive no service ($\mu = 0$) when y is large. Even stronger, it might be the case that when taken into service, μ is positive, but as the number of regular customers is increasing, the service rate might decrease to zero.

3.3 Examples

We consider two examples, one for with $c(\mu) = 0$, and one for which it is positive.

Example 1

Consider an example with the following parameters: $\lambda_{reg} = 3$, $\lambda_{opp} = 1$, $C_{opp} = 8$, $\bar{\mu} = 10$, $c(\mu) = 0$, $h_{opp}(x) = x$ and

$$h_{reg}(y) = \begin{cases} 0.05\, y^2 & \text{if } y < 20; \\ 100\, y & \text{otherwise.} \end{cases}$$

Hence, the holding costs are more than linearly increasing. Moreover, for computational purposes we can truncate the state space for y large, as the optimal policy avoids getting to $y \geq 20$. The optimal policy for accepting opportunity customers is given by:

$x \backslash y$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	1	1	1	1	1	1	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-
1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Here a 1 indicates acceptance. So, the optimal policy is indeed a threshold policy for accepting an opportunity customer. The threshold is $T = 6$, where the opportunity customer is accepted when $y \leq T$, and rejecting otherwise.

The optimal fraction $\mu \in [0, 1]$ for the service speed of the opportunity customer is given by:

$x \backslash y$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0

As remarked in Section 3.2, the opportunity customers either gets full attention of the server, or no attention at all. The threshold for this is $M = 10$.

Example 2

We use the same parameter values as in Example 1, however, for $c(\mu)$ we now take:

$$c(\mu) = \begin{cases} 0 & \text{if } 0 \leq \mu < 0.25; \\ 0.5 & \text{if } 0.25 \leq \mu < 0.50; \\ 1 & \text{if } 0.50 \leq \mu < 0.75; \\ 1.5 & \text{if } 0.75 \leq \mu \leq 1. \end{cases}$$

Hence, $c(\mu)$ is increasing and satisfying the assumption $c(0) = 0$. The threshold for accepting opportunity customers now is $T = 5$. The optimal fraction $\mu \in [0, 1]$ for the service speed of the opportunity customer is given by:

$x \backslash y$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	0.75	0.25	0.25	0.25	0.25	0	0	0	0	0	0	0	0	0

So, the server speed for the opportunity customer is indeed decreasing.

4 Generalized model

We consider two generalizations of the model of Section 2. Firstly, we allow opportunity customers to queue as well. Secondly, we can decide whether to accept or reject regular customers. We show that both generalizations fit in the same framework.

4.1 Queueing of opportunity customers

Instead of having to take an opportunity customer directly into service, we now allow them to queue as well, incurring some holding costs $h_{opp}(\cdot)$. Here, we assume that both types of customers form separate queues. The server can only work on the first in line customers of both queues.

By only changing the $h_{opp}(\cdot)$, we can achieve this generalization. For a finite buffer, of say S_{opp} places, instead of (1), we get

$$h_{opp}(x) = \begin{cases} h_{opp}(x) & \text{if } x \leq S_{opp}; \\ K\,x & \text{if } x > S_{opp}, \end{cases}$$

where again K is a very large constant, and $h_{opp}(x)$ is assumed to be an increasing convex function. For the infinite buffer case, set $S_{opp} = \infty$, although in this case we need a more strict stability condition when $h_{opp} \equiv 0$. As all opportunity customers will be accepted, the queue can become infinitely large. To prevent this, the stability condition becomes $(\lambda_{reg} + \lambda_{opp})/\bar{\mu} < 1$.

Theorem 4.1. *(i) The optimal policy for admitting opportunity customers is a state dependent threshold policy. That is, there exist a switching curve, say $T(x)$, such that the optimal decision is to accept the opportunity customer when $y \leq T(x)$, and to reject it otherwise. Moreover, (ii) $T(x)$ is decreasing in x and (iii) in the direction $e_x - e_y$.*

Here, $e_x = (1, 0)$ and $e_y = (0, 1)$.

4.2 Accept or reject regular customers

When we allow that regular customers are rejected as well, we find the same kind of structural results for this decision. For this, instead of the operator $T_{A(2)}$, we now have:

$$T_{CA(2)} f(x, y) = \min\{V_n(x, y + 1), V_n(x, y) + C_{reg}\},$$

where C_{reg} are the costs for rejecting a regular customer. Analogously to Theorem 4.1, part (i), the optimal decision for acceptance can again be characterized by a state depended threshold.

When combining both generalizations, we have a two queue head-of-line processor sharing model, controlling the division of the service rate as well as the acceptance of customers in both queues.

5 Discussion

We presented a single server head-of-line processor sharing model. For this, we derived the structure of the optimal policy. The results are in line with one's intuition for the control of the system. We also discussed two model generalizations. Furthermore, there are generalization that are straightforward to make, such as having multiple types of opportunity customers, each having different costs for rejecting them.
An interesting option for further research is to derive the steady state probability distribution of the number of customers in the system, when executing the optimal policy. For $c(\mu) = 0$ this might be straightforward, as the optimal server speed is either 1 or 0, where for the case with positive $c(\mu)$ this might be more work. From the steady state the average costs readily follow. This can be used in a numerical study, to compare the performance of the optimal policy to that of a policy that always accepts or always rejects the opportunity customers, or a policy that always gives full attention to the opportunity customer in service.
Another interesting questions for further research is whether the structural results will remain to hold when the total service rate increases or decreases when the server divides its attention to two customers.

6 Acknowledgments

The author would like to thank N.C. Büyükkaramikli for suggesting the idea of the model. Furthermore, she would like to thank I.J.B.F. Adan and G.J. van Houtum for useful comments on an earlier draft of this paper.

A Proofs

A.1 Proof of Lemma 3.1

Proof. For $T_{CA(1)}$, $T_{A(2)}$, T_{unif}, and T_{costs}, the statements follow from [6], respectively from Theorem 7.2 (using that we have a two dimensional state space), from Theorem 7.2 again, and the latter two both from Theorem 7.1.

$\tilde{T}_{CTD(1)}$ is a small variation of $T_{CTD(1)}$ as in Definition 5.4 of [6]. It holds that $\tilde{T}_{CTD(1)}f(x,y) = T_{CTD(1)}f(x,y-1)$ for $y>0$. Hence, for $y>0$, the statement follow from Theorem 7.4 of [6]. It remain to prove the statement for $y=0$, which boils down to checking the cases $x=0$, $y=0$ and $x>0$, $y=0$. Basically, one has to prove that Supermod, SuperC(x,y), and SuperC(y,x) are preserved, as, by (8), this implies MM.

We only present the proof for Supermod here, when $x>0$ and $y=0$. Assume that a function f is MM, implying that f is Supermod, then we show that $\tilde{T}_{CTD(1)}f$ is Supermod as well. We denote the minimizers μ's in $\tilde{T}_{CTD(1)}f(x,0)$ and $\tilde{T}_{CTD(1)}f(x+1,1)$ by μ_1, respectively μ_2, both in $[0,1]$. As the optimal μ is decreasing in y, it follows that $\mu_1 \geq \mu_2$. The proof then makes use of the trivial identity $g(\mu_1) \geq \min_{\mu\in[0,1]} g(\mu)$ for all functions g.

$$
\begin{aligned}
&\tilde{T}_{CTD(1)}f(x,0) + \tilde{T}_{CTD(1)}f(x+1,1) \\
&= c(\mu_1) + \mu_1 f(x-1,0) + (1-\mu_1)f(x,0) + c(\mu_2) + \mu_2 f(x,1) + (1-\mu_2)f(x+1,0) \\
&= c(\mu_1) + c(\mu_2) + \big(\mu_1-\mu_2\big)\big(f(x-1,0)+f(x+1,0)\big) + \mu_2\big(f(x-1,0)+f(x,1)\big) \\
&\qquad + \big(1-\mu_1\big)\big(f(x,0)+f(x+1,0)\big) \\
&\geq c(\mu_1) + c(\mu_2) + 2\big(\mu_1-\mu_2\big)f(x,0) + \mu_2\big(f(x,0)+f(x-1,1)\big) \\
&\qquad + \big(1-\mu_1\big)\big(f(x,0)+f(x+1,0)\big) \\
&= c(\mu_1) + \mu_1 f(x,0) + (1-\mu_1)f(x+1,0) + c(\mu_2) + \mu_2 f(x-1,1) + (1-\mu_2)f(x,0) \\
&\geq \min_{\mu\in[0,1]} \Big\{ c(\mu) + \mu f(x,0) + (1-\mu)f(x+1,0) \Big\} \\
&\qquad + \min_{\mu\in[0,1]} \Big\{ c(\mu) + \mu f(x-1,1) + (1-\mu)f(x,0) \Big\} \\
&= \tilde{T}_{CTD(1)}f(x+1,0) + \tilde{T}_{CTD(1)}f(x,1)
\end{aligned}
$$

where the first inequality holds as f is Conv(x) (applied to the term with $\mu_1-\mu_2$, using that $\mu_1-\mu_2 \geq 0$) and Supermod (applied to the term with μ_2). Note that f is Conv(x) is implied by the fact that f is MM.

The other proofs follow along the same lines. □

A.2 Proof of Theorem 3.3

Proof. a) By Theorem 8.1 of [6], as V_n is Supermod.

b) Along the same lines as [13, Theorem on page 206]. The reasoning for the existence of an optimal policy simplifies in our case, as we study a discounted cost problem. □

A.3 Proof of Theorem 3.3

Proof. The proof makes again use of Theorem 8.1 of [6]. □

References

[1] G. Fayolle, P.J.B. King, and I. Mitrani. The solution of certain two-dimensional Markov models. *Advances in Applied Probability*, 14(2):295–308, 1982.

[2] B. Hajek. Extremal splittings of point processes. *Mathematics of operations research*, 10(4):543–556, 1985.

[3] A.G. Konheim, I. Meilijson, and A. Melkman. Processor-sharing of two parallel lines. *Journal of Applied Probability*, 18(4):952–956, 1981.

[4] G. Koole. Structural results for the control of queueing systems using event-based dynamic programming. *Queueing Systems*, 30(3):323–339, 1998.

[5] G. Koole. Stochastic scheduling with event-based dynamic programming. *Mathematical Methods of Operations Research (ZOR)*, 51(2):249–261, 2000.

[6] G. Koole. Monotonicity in Markov reward and decision chains: Theory and applications. *Foundations and Trends in Stochastic Systems*, 1(1):1–76, 2006.

[7] S.A. Lippman. Applying a new device in the optimization of exponential queueing systems. *Operations Research*, 23(4):687–710, 1975.

[8] M.L. Puterman. *Markov decision processes: Discrete stochastic dynamic programming.* John Wiley & Sons, Inc. New York, NY, USA, 1994.

[9] S. Stidham Jr. Optimal control of admission to a queueing system. *Automatic Control, IEEE Transactions on*, 30(8):705–713, 1985.

[10] D. Towsley, P.D. Sparaggis, and C.G. Cassandras. Optimal routing and buffer allocation for a class of finite capacity queueing systems. *Automatic Control, IEEE Transactions on*, 37(9):1446–1451, 1992.

[11] W. Van der Weij, S. Bhulai, and R. Van der Mei. Optimal scheduling policies for the limited processor sharing queue. *Queueing Systems*, 2009.

[12] K.M. Wasserman and N. Bambos. Optimal server allocation to parallel queues with finite-capacity buffers. *Probability in the Engineering and Informational Sciences*, 10(02):279–285, 1996.

[13] R.R. Weber and S. Stidham. Optimal control of service rates in networks of queues. *Advances in Applied Probability*, 19(1):202–218, 1987.