

Structural results for the control of queueing systems using event-based dynamic programming

Ger Koole
Vrije Universiteit
Department of Mathematics
De Boelelaan 1081a, 1081 HV Amsterdam
The Netherlands
Email: koole@few.vu.nl, Url: www.math.vu.nl/~koole

Published in *Queueing Systems* **30**:323-339, 1998

Abstract

In this paper we study monotonicity results for optimal policies of various queueing and resource sharing models. The standard approach is to propagate, for each specific model, certain properties of the dynamic programming value function. We propose a unified treatment of these models by concentrating on the events and the form of the value function instead of on the value function itself. This is illustrated with the systematic treatment of one and two-dimensional models.

Keywords: Control of queueing systems, dynamic programming, supermodularity, threshold policies.

Short title: Control of queueing systems

1 Introduction

This paper introduces *event-based dynamic programming*, a systematic approach for deriving monotonicity results of optimal policies for various queueing and resource sharing models. After introducing it rigorously we apply it to many one and two-dimensional models. To motivate the method and to introduce the concepts in a simple way, we start with an example. After the example we discuss related literature and we give an overview of the paper.

Example Consider the simplest controlled queueing model that one can think of: a single queue with an exponential server, and Poisson arrivals of which the admission can be controlled. The arrival rate is λ , the service rate μ , and we assume (without restricting generality) that $\lambda + \mu = 1$. For each customer that is not admitted we incur costs c , and there are holding costs equal to $C(x)$, if there are x customers in the system. The standard approach is to formulate the value function and to derive our results from that. Using uniformization (see Lippman [13]) we find for our example as value function ($\mathbb{N}_0 = \{0, 1, \dots\}$, $x^+ = \max\{x, 0\}$):

$$V_{n+1}(x) = C(x) + \lambda \min \{V_n(x+1), c + V_n(x)\} + \mu V_n((x-1)^+), \quad x \in \mathbb{N}_0.$$

It can be shown that if C and V_0 are convex and nondecreasing, then so is V_n for each $n > 0$. If $V_n(x+1) \leq c + V_n(x)$, then admission is optimal in x . From the convexity it follows that

$V_n(x+1) - V_n(x)$ is nondecreasing in x , and thus there is a critical level, the *threshold*, such that admission is only optimal below the level.

Now assume that we want to study a discrete time model where at each instant a customer arrives, and with geometric service times. Again the admission can be controlled. The decision instant is just after a (potential) service completion and just before the admission decision. This gives a value function of the form:

$$V'_{n+1}(x) = C(x) + \min \left\{ pV'_n(x) + (1-p)V'_n(x+1), c + pV'_n((x-1)^+) + (1-p)V'_n(x) \right\},$$

with p the probability of a service completion. Showing the optimality of a threshold policy directly is not trivial. Using event-based dynamic programming (dp) we proceed as follows. Define, for any function $f : \mathbb{N}_0 \rightarrow \mathbb{R}$, $T_{AC}f(x) = \min\{f(x+1), c + f(x)\}$, $T_Df(x) = f((x-1)^+)$ and $T_{costs}f(x) = C(x) + f(x)$, and for functions f_1 and f_2 , $T_{unif}(f_1, f_2)(x) = pf_1(x) + (1-p)f_2(x)$. These T 's are called the *event operators*. Now V'_n can be written recursively as:

$$V'_{n+1}(x) = T_{costs} \left(T_{AC} \left(T_{unif} \left[T_D(V'_n), V'_n \right] \right) \right) (x).$$

To show that a threshold policy is optimal, we need that $T_{unif}[T_D(V'_n), V'_n]$ is convex and nondecreasing (CI for short) for all n . If V'_n is CI then this follows from the fact that for all f and g CI, T_Df and $T_{unif}(f, g)$ (for each $0 \leq p \leq 1$) are CI. To show that V'_n is CI we have to assume that C and V_0 are CI, and to show that $T_{AC}f$ is CI for all f CI. The inductive proofs for the operators can be found in Section 3. This proves the optimality of threshold policies for the discrete time model.

Now observe that V_n , the value function of the continuous time model, is given recursively by $V_{n+1} = T_{costs}(T_{unif}[T_{AC}V_n, T_DV_n])$. Thus the same method applies to the continuous time model. We see that by concentrating on the events instead of the whole value function at once, we can deal with the continuous and the discrete time model at the same time. More models can be made using the same operators: we can think of different arrival streams with different blocking costs, or arrivals which take place each two interarrival epochs, etc. A less trivial direction is verifying if other operators propagate, which enables us to study other models. This is, for functions that are CI, extensively done in Section 3.

We now discuss related literature. The technique described in the beginning of the example was first applied by Lippman [13] to a number of single queue models, related to our example. Uniformization is crucial to his analysis, as he studies continuous time models. Later Hajek [7] and Weber & Stidham [20] followed a similar approach for multi-dimensional models. Here the convexity is replaced by submodularity or related properties. In [7] a routing problem to two parallel queues is considered, and it is shown that the optimal assignment policy has an increasing switching curve. A cycle of m queues where the service rates can be controlled is considered in [20]. Monotonicity results for the optimal service rates are derived.

A first effort to come to a general framework for the study of monotonicity results can be found in Glasserman & Yao [6]. They replace the state by an event-counting vector, the *score*, from which the actual state can be derived. Each possible event has its operator, which all have roughly the form of T_{AC} above. The value function is a convex combination of these operators. For our example it would mean that the scores are of the form $d = (d_1, d_2)$, with d_1 the number of arrivals and d_2 the number of departures since time 0, and $d_1 - d_2$ gives the number of customers in the system (given that it is initially empty). For each two possible events submodularity is shown to propagate. (Supermodularity is also considered in [6], but as

there is apparently an error in Lemma 4.2, only the results related to submodularity are valid.) For the example the submodularity in the score space is translated into convexity in the state space, giving essentially the same result as the standard approach for the continuous time model. Although elegant and insightful, the applicability of this method remains limited: if we would add an extra arrival stream in the example, then for the resulting three-dimensional event space submodularity does not hold anymore. Note that it is trivial to add an extra stream in event-based dp.

A new direction was taken in Altman & Koole [2]. Starting from the score space approach in [6] they consider convex combinations of concatenations of event operators. As in event-based dp this allows the study of continuous and discrete time models at the same time. The main application of this method is the analysis and generalization of the model studied in [20].

With event-based dynamic programming we work again with the actual states, and we allow for (almost) arbitrary event operators. We consider not just submodularity, but different classes of functions. With every class of functions there is associated a group of operators that propagate. From these operators models can be build, and structural results can be derived from the class of functions for each operator separately. Operators exist that model servers, (controlled) arrival streams, a finite source, but as we saw in the example there is also an operator for uniformization and for the direct costs. This way we obtain a very concise notation. Further generality is achieved by introducing an extra state component representing the environment. The associated operator generalizes the uniformization to coefficients that depend on the environment state. This can be used for example to model Markov Arrival Processes and server vacations. Further details and generalizations are given in Sections 3 and 4.

In Section 2 event-based dynamic programming is introduced. In Section 3 we consider models with one-dimensional inequalities (like convexity), in Section 4 models with two-dimensional inequalities (like sub and supermodularity) are studied.

The results for specific models include and generalize those from Lippman [13] and Stidham [18] on the admission control of single and multi-server queues, from Hajek [7] on the routing to two queues, from Weber & Stidham [20] on a cykel or a tandem of queues (we restrict ourselves to two queues), and from Lin & Kumar [12] on the scheduling of a single server queue with an additional server. Typical generalizations are the extension to discrete time models, and the inclusion of finite buffers and general arrival streams. Besides that we obtain new results on the stochastic knapsack (see Ross & Tsang [15]). Although we restrict our attention to the models mentioned above, the method has already been applied successfully to stochastic scheduling models in Koole [11].

2 Event-based dynamic programming

In this section we formulate the dynamic programming value function in general terms and prove some theorems which form the basis of our method.

We take $x \in \mathbb{N}_0^m$ to be our state space, and let \mathcal{V} be the class of functions from \mathbb{N}_0^m to \mathbb{R} . We define the operators T_0, \dots, T_{k-1} as follows. The operator $T_i : \mathcal{V}^{l_i} \rightarrow \mathcal{V}$, $l_i \in \mathbb{N}$, is given by

$$T_i(f_1, \dots, f_{l_i})(x) = \min_{a \in A_i(x)} \left\{ c_i(x, a) + \sum_{j=1}^{l_i} \sum_{y \in \mathbb{N}_0^m} p_i^j(x, a, y) f_j(y) \right\},$$

for all $f_1, \dots, f_{l_i} \in \mathcal{V}$ and $x \in \mathbb{N}_0^m$. $A_i(x)$ is called the action set, $c_i(x, a)$ the direct costs and $p_i^j(x, a, y)$ the transition probabilities. We often take $l_i = 1$, as we will see in the next Sections. In this case we get (we omit the superscript of p)

$$T_i f(x) = \min_{a \in A_i(x)} \left\{ c_i(x, a) + \sum_{y \in \mathbb{N}_0^m} p_i(x, a, y) f(y) \right\},$$

which is the standard dp operator, given that $p^1(x, a, y) \geq 0$ for all x, a, y , and that

$$\sum_y p^1(x, a, y) = 1$$

for all x, a . We saw in the example in the introduction that the uniformization operator is an important exception to $l_i = 1$. In applications we choose the event operators as simple as possible, by associating one with every possible event in the system.

Using these operators we construct the value function $V_n : \mathbb{N}_0^m \rightarrow \mathbb{R}$, $n \geq 0$. We take V_0 arbitrary. We construct V_{n+1} , $n \geq 0$, from V_n and the k (not necessarily different) operators T_1, \dots, T_k . To do this we define first $V_n^{(0)}, \dots, V_n^{(k)}$:

i) $V_n^{(k)} = V_n$;

ii) $V_n^{(j)} = T_j(V_n^{(k_1)}, \dots, V_n^{(k_{l_j})})$, $j = 0, \dots, k-1$, for k_1, \dots, k_{l_j} such that $j < k_1, \dots, k_{l_j} \leq k$.

V_{n+1} is defined by taking $V_{n+1} = V_n^{(0)}$. A value function V_n of this form will be called an *event-based dp value function*. The assumption that $k_1, \dots, k_{l_s} > j$ is made to avoid a circular definition.

The example of the Introduction, with value function $V_{n+1} = T_{costs}(T_{unif}[T_{AC}V_n, T_D V_n])$, can indeed be written in this form, by taking $k = 4$ and $V_n^{(0)} = T_{costs}(V_n^{(1)})$, $V_n^{(1)} = T_{unif}(V_n^{(2)}, V_n^{(3)})$, $V_n^{(2)} = T_{AC}(V_n^{(4)})$, and $V_n^{(3)} = T_D(V_n^{(4)})$.

Although the definition of an event-based dp value function is notationally quite complex, we saw that the intuition is simple: each step of the dp consists of the parallel and/or consecutive execution of several events. If $l_i > 1$ for some i , then also the determination of which events are to be executed depends on the state, the realization, or the action. The central idea of how to use event-based dp is summarized in the following (trivial) theorem.

Theorem 2.1 *Let \mathcal{F} be some class of functions from \mathbb{N}_0^m to \mathbb{R} , and $V_0 \in \mathcal{F}$. If, for all i , for $f_1, \dots, f_{l_i} \in \mathcal{F}$ holds that $T_i(f_1, \dots, f_{l_i}) \in \mathcal{F}$ then $V_n \in \mathcal{F}$ for all n .*

In what follows we consider special classes \mathcal{F} and show that certain event operators T are closed under \mathcal{F} . This proves that the value function $V_n \in \mathcal{F}$ for all models which can be constructed with these T . We choose \mathcal{F} such that certain structural properties of the optimal control policies can be derived from it.

On the other hand, it is interesting to show that V_n as defined can be rewritten in the standard MDP formulation, given by

$$W_{n+1}(x) = \min_{a \in A(x)} \left\{ k(x, a) + \sum_y q(x, a, y) W_n(y) \right\}.$$

This allows us to use techniques and results from the theory of MDP's.

Theorem 2.2 *The functions k and q can be chosen such that $W_n = V_n$.*

Proof The crucial step is to show that $T_0(f_1, \dots, f_l)$ with $f_i = T_i W_n$ (for T_0, \dots, T_l arbitrary) can be written in the standard form. Repeating this gives the result.

To put T_0 in the standard form we construct a new operator with the same state space; as action space we take $A_0(x) \times \prod_{j=1}^l \prod_y A_j(y)$. Denote a typical element with (a, \tilde{a}) , where \tilde{a} is a 2-dimensional matrix, i.e., $\tilde{a}_j(y)$ is the action taken in operator j if a change of state to j took place. Take

$$k(x, (a, \tilde{a})) = c_0(x, a) + \sum_{y,j} p_0^j(x, a, y) c_j(y, \tilde{a}_j(y))$$

and

$$q(x, (a, \tilde{a}), z) = \sum_{y,j} p_0^j(x, a, y) p_j(y, \tilde{a}_j(y), z).$$

This gives the equivalence as is easily seen. \square

Finally let us consider optimality criteria. Normally we assume that all $p_i^j(x, a, y) \geq 0$ and that for all i, x and a $\sum_{y,j} p_i^j(x, a, y) = 1$: then V_n represents the total minimal n -stage costs, and under certain conditions the policy minimizing V_n as $n \rightarrow \infty$ is average optimal. These conditions however are non-trivial and should be checked for each model separately, unless the state space is finite. If we make the exception that $\sum_{y,j} p_0^j(x, a, y) = \alpha$, then V_n converges (again, under certain conditions) to the minimal discounted costs. Our focus in this paper is on the properties of the value function, not on the existence of limiting policies. In the case of a finite state space existence is guaranteed for all models. For discounting results from Schäl [16] often provide the necessary existence result; for average costs some useful conditions are summarized in Cavazos-Cadena & Sennott [4].

3 Models with one-dimensional inequalities

In this Section we present the models with one-dimensional inequalities, the most important one being convexity. Let us start with the simplest models. Later in this section we add an environment variable to the system. For this reason, and because we will use the same operators in the next Section for more-dimensional models, we use a multi-dimensional notation. We use queueing terminology (e.g., we will call the state components queues), but non-queueing applications can be thought of.

Notationally we make use of the following conventions: e_i denotes the i th unity vector, e is the vector which each entry is 1, each vector (in)equality is taken componentwise, $I\{\dots\}$ is the indicator function, and increasing and decreasing are used in the non-strict sense.

The operators that we use in this Section are:

- $T_{costs} f(x) = C(x) + \alpha f(x)$. T_{costs} represents the direct (action independent) costs and the discounting. (α is the discount factor, thus we often take $\alpha \in (0, 1]$, with $\alpha = 1$ representing total costs, but we only need $\alpha \geq 0$.) Although it is notationally not correct we write $T_{costs}(C, f)$ to indicate that the conditions for f must also hold for C (see below). Sometimes we add to C a component $\bar{C} = g(x)K$, $g: \mathbb{N}_0^m \rightarrow \mathbb{N}_0$ and $K > 0$ which allows us to consider for example finite buffers. This is done by taking K large enough such that any state x with $g(x) \neq 0$ will be avoided by the optimal policy, if at all possible.

- $T_{unif}(f_1, \dots, f_l)(x) = \sum_{j=1}^l p(j) f_j(x)$ with $p(j) > 0$ for all j . This is a convex combination of the f_j . The value function of a continuous time model typically has this form, due to the uniformization. This technique was first introduced in Lippman [13], and further developed

in Serfozo [17]. It is the basis of the analysis of most continuous-time Markovian models.

- $T_{A(i)}f(x) = f(x + e_i)$. An arrival at queue i .
- $T_{FS(i)}f(x) = \begin{cases} (B - x_i)f(x + e_i) + x_i f(x) & \text{if } x_i \leq B, \\ Bf(x) & \text{otherwise.} \end{cases}$

$T_{FS(i)}$ models the finite source queue. We need to define $T_{FS(i)}$ also for $x_i > B$, because the inequalities below are defined for all $x \in \mathbb{N}_0^m$. Besides that it can be the case that another operator adds customers to queue i , making it possible that these states are reached indeed.

• $T_{AC(i)}f(x) = \min\{c + f(x), c' + f(x + e_i)\}$. T_{AC} models admission control at queue i . If a customer is rejected costs equal to c are incurred, when a customer is admitted the costs are equal to c' , which can be interpreted as a reward equal to $-c'$.

- $T_{D(i)}f(x) = f((x - e_i)^+)$. A departure for the single server queue i .
- $T_{MD(i)}f(x) = \begin{cases} x_i f(x - e_i) + (S - x_i)f(x) & \text{if } x_i < S, \\ Sf(x - e_i) & \text{otherwise.} \end{cases}$

The multi server queue with S servers.

- $T_{CD(i)}f(x) = \begin{cases} \min_{\mu \in [0,1]} \{c_\mu + \mu f(x - e_i) + (1 - \mu)f(x)\} & \text{if } x_i > 0, \\ c_0 + f(x) & \text{otherwise.} \end{cases}$

T_{CD} models controlled departures. By taking c_μ high enough for certain values we can restrict the possible rates. Note that if $c_\mu = 0$ for all μ , or, more general, linear, then a minimizer can also be found in the set $\{0, 1\}$. This phenomenon is known as bang-bang control.

We continue with the inequalities that we consider for the operators defined above. We say that a function f has a certain property if the inequality as specified holds.

- $Inc(i)$: $f(x) \leq f(x + e_i)$, for all $x \in \mathbb{N}_0^m$: increasingness.
- $Conv(i)$: $2f(x + e_i) \leq f(x) + f(x + 2e_i)$, for all $x \in \mathbb{N}_0^m$: convexity.

In what follows we will often prove relations of the type: for operator T , if f_1, \dots, f_i all have the properties $X1, \dots, Xk$, then $T(f_1, \dots, f_i)$ has property Xj . We introduce the following notation for these relations: $T: X1, \dots, Xk \rightarrow Xj$.

Lemma 3.1 *The following relations hold:*

- T_{costs} : $Inc(1) \rightarrow Inc(1)$; $Conv(1) \rightarrow Conv(1)$.
- T_{unif} : $Inc(1) \rightarrow Inc(1)$; $Conv(1) \rightarrow Conv(1)$.
- $T_{A(1)}$: $Inc(1) \rightarrow Inc(1)$; $Conv(1) \rightarrow Conv(1)$.
- $T_{FS(1)}$: $Inc(1) \rightarrow Inc(1)$; $Conv(1) \rightarrow Conv(1)$.
- $T_{AC(1)}$: $Inc(1) \rightarrow Inc(1)$; $Conv(1) \rightarrow Conv(1)$.
- $T_{D(1)}$: $Inc(1) \rightarrow Inc(1)$; $Inc(1), Conv(1) \rightarrow Conv(1)$.
- $T_{MD(1)}$: $Inc(1) \rightarrow Inc(1)$; $Inc(1), Conv(1) \rightarrow Conv(1)$.
- $T_{CD(1)}$: $Inc(1) \rightarrow Inc(1)$ if $c_0 = \min_{\mu \in [0,1]} c_\mu$; $Conv(1) \rightarrow Conv(1)$.

Proof The results for T_{costs} and T_{unif} follow directly as increasingness and convexity are closed under convex combinations. Also the results for $T_{A(1)}$ follow directly, by replacing x by $x + e_1$ in the inequalities. For $T_{FS(1)}$ certain terms cancel out. Consider $T_{AC(1)}$. Because $\min\{c + f(x), c' + f(x + e_1)\} \leq c + f(x) \leq c + f(x + e_1)$ and $\min\{c + f(x), c' + f(x + e_1)\} \leq c' + f(x + e_1) \leq c' + f(x + 2e_1)$, the first relation follows. For the convexity we need to check the different possibilities for the minimizers in x and $x + 2e_1$, denoted by a_1 and a_2 respectively. The only case of interest is $a_1 \neq a_2$. It is readily seen that $2 \min\{c + f(x + e_1), c' + f(x + 2e_1)\} \leq c' + f(x + e_1) + c + f(x + 2e_1)$, which corresponds to the case that a_1 corresponds to admission and a_2 to rejection, and $2 \min\{c + f(x + e_1), c' + f(x + 2e_1)\} \leq c + f(x) + c' + f(x + 3e_1)$ (by using convexity twice), which corresponds to the remaining case (which in fact cannot occur,

as we will see later). Consider now $T_{D(1)}$. The increasingness follows as for $T_{A(1)}$, except if $x_1 = 0$. In this case $T_{D(1)}f(x) = T_{D(1)}f(x + e_1)$. Also for the convexity the only non-trivial case is $x_1 = 0$. This reduces to $f(x) \leq f(x + e_1)$. Roughly the same arguments are used for $T_{MD(1)}$. For $T_{CD(1)}$ we should, as for $T_{AC(1)}$, consider the different possibilities for the minimizer at the right hand side of the inequality. \square

Finally we will be able to look at the first models which we can analyze with the results obtained so far. We do this by looking at sets of functions. Let \mathcal{P} be a set of properties as defined above. The set of functions $\mathcal{F}(\mathcal{P})$ consists of all functions that satisfy all properties in \mathcal{P} . We start by considering $\mathcal{F}(Inc(1), Conv(1))$.

- $\mathcal{F}(Inc(1), Conv(1))$ By Lemma 3.1 we can make our model from the operators T_{costs} , T_{unif} , $T_{A(1)}$, $T_{FS(1)}$, $T_{AC(1)}$, $T_{D(1)}$, $T_{MD(1)}$, and $T_{CD(1)}$ with $c_0 = \min_{\mu \in [0,1]} c_\mu$. The examples of the Introduction, which values functions such as $V_{n+1} = T_{costs}(C, T_{unif}[T_{AC(1)}V_n, T_{D(1)}V_n])$, clearly fall within this class, given that $C \in \mathcal{F}(Inc(1), Conv(1))$. From the convexity we can derive that the optimal policy, for each n , is of threshold form. The full argument is as follows. The state x is one-dimensional, therefore we omit the subscript 1. If $c + V_n(x) < (>) c' + V_n(x + 1)$, then V_{n+1} is minimized by the action corresponding to rejection (admission) of the customer. In case of equality both actions are optimal. By the convexity of V_n the difference $V_n(x + 1) - V_n(x)$ is increasing in x . We distinguish three cases. Either there is an $x^* > 0$ such that $c + V_n(x^* - 1) \geq c' + V_n(x^*)$ and $c + V_n(x^*) < c' + V_n(x^* + 1)$, in which case there is an optimal policy which admits customers if and only if $x < x^*$. This x^* is called the threshold. In case $c + V_n(x) \geq (<) c' + V_n(x + 1)$ for all x then admission (rejection) is always optimal, corresponding to $x^* = \infty$ ($x^* = 0$).

V_n as above is the value function, after uniformization, of the model with Poisson arrivals and exponential service times, thus a controlled $M/M/1$ queue. An interesting model is the controlled $GI/M/1$ queue, the basic model of Stidham [18]. This can be modeled by $V_{n+1} = T_{costs}(C, T_{AC(1)}T_{unif}[V_n, T_{D(1)}V_n, T_{D(1)}^2V_n, \dots])$, with $T_{D(1)}^j$ the j -fold convolution of $T_{D(1)}$, and with $p(j)$, the coefficient of $T_{D(1)}^jV_n$, the probability that j (potential) departures occur during an interarrival time.

One of the generalizations treated in [18] is batch arrivals, where each customer in the batch can be assigned individually. This corresponds to the repeated assignment of a single customer, and can thus be dealt with by taking a convex combination of convolutions of $T_{AC(1)}$. A more interesting generalization is the inclusion of finite buffers. As said, we can avoid certain states by adding a component \bar{C} to T_{costs} which takes a value K or multiples of it in states which are to be avoided, for K large enough. Of course \bar{C} has to be in $\mathcal{F}(Inc(1), Conv(1))$. If we want to model a finite buffer with size B we can take $\bar{C}(x) = (x - B)^+K$. This function is 0 for $x = 0, \dots, B$, and both increasing and convex, and thus we can model finite buffers by adding \bar{C} to the direct costs. Now we can capture the first model of Lippman [13] and the model of Ramjee et al. [14], which are both finite capacity $M/M/c$ queues with multiple customer classes, distinguished by their admission reward/blocking costs. The third model of [13] contains an arrival operator of the form $T_{CD(1)}$. $T_{AC(1)}$ can easily be generalized so as to deal with this model as well.

- $\mathcal{F}(Conv(1))$ If we consider only convexity, then we have to build our models from T_{costs} , T_{unif} , $T_{A(1)}$, $T_{FS(1)}$, $T_{AC(1)}$, and $T_{CD(1)}$, thus the departures should be controllable with idling as option. The advantage of such a choice is that we can take a non-increasing C in T_{costs} .

An example of such a C is $C(x) = -I\{x > 0\}r + cx$, where r is some positive reward related to the service of customers and c is the holding cost rate. This is the second model of [13].

Another interpretation is as follows. Let x represent the stock of a certain good. Sales are represented by T_D , and production by $T_{AC(1)}$. Again it makes sense to take a cost function of the form $C(x) = -I\{x > 0\}r + cx$, with r the profit rate, and c the inventory costs. Our result states that there is a certain level below which production should take place.

So far we used our one-dimensional inequalities for a one-dimensional model. Next we introduce an operator which uses another state component as environment.

- $T_{env(i)}(f_1, \dots, f_l)(x) = \sum_{y \in \mathbb{N}_0} \lambda(x_i, y) \sum_{j=1}^l q^j(x_i, y) f_j(x^*)$, where x^* is equal to x with the i th component replaced by y .

We present the equivalent of Lemma 3.1 for this new operator, after which we discuss in detail its modelling capabilities.

Lemma 3.2 *The following relations hold:*

- $T_{env(0)}: Inc(1) \rightarrow Inc(1); Conv(1) \rightarrow Conv(1)$.

Proof Immediate because the transitions do not depend on components other than the 0th. As they are otherwise arbitrary, there are no results for the 0th component itself. \square

Lemma 3.2 shows that we can add $T_{env(0)}$ to the classes of possible operators for the sets $\mathcal{F}(Inc(1), Conv(1))$ and $\mathcal{F}(Conv(1))$, making that all previous results still hold with the added possibilities of $T_{env(0)}$. $T_{env(0)}$ can be seen as a generalization of T_{unif} , in the sense that the operator which is to be executed depends also on the state x_0 of the environment. An example is

$$V_{n+1} = T_{costs} \left(C, T_{unif} \left[T_{env(0)} \left(T_{AC(1)}^{(1)} V_n, \dots, T_{AC(1)}^{(l)} V_n \right), T_{D(1)} V_n \right] \right),$$

with the $T_{AC(1)}^{(j)}$ the operators of l arrival streams, having different rejection costs. Thus the $\lambda(x_0, y)$ in $T_{env(0)}$ are the normalized transition rates of a Markov process (with bounded rates) on \mathbb{N}_0 , and q^j gives the probability of an arrival in class j at the event of the transition. With this *Markov Arrival Process* (MAP) each marked point process can be approximated arbitrarily close, see Asmussen & Koole [3]. This shows how to generalize the results above to general arrival streams. The value function can be rewritten as

$$V_{n+1} = T_{costs} \left(C, T_{env(0)} \left[T_{AC(1)}^{(1)} V_n, \dots, T_{AC(1)}^{(l)} V_n, T_{D(1)} V_n \right] \right),$$

giving also the possibility to let the departures depend on the environment. This opens numerous possibilities for generalizations. Examples are server vacations, by letting x_0 denote the number of active servers, or a finite source with a varying number of customers. Note however that although the optimal policy keeps its structure, it will depend on the state of the environment: the structural results hold for each environment state separately. Our results for the one-dimensional models can be summarized as follows.

Theorem 3.3 *For an event-based value function V_n , constructed with some or all operators of the form T_{costs} , $T_{env(0)}$, $T_{A(1)}$, $T_{FS(1)}$, $T_{AC(1)}$, $T_{D(1)}$, $T_{MD(1)}$, and $T_{CD(1)}$ with $c_0 = \min_{\mu \in [0,1]} c_\mu$, we have that $V_n \in \mathcal{F} = \mathcal{F}(Inc(1), Conv(1))$ if $C \in \mathcal{F}$ (for C from T_{costs}) and $V_0 \in \mathcal{F}$. If C and $V_0 \in \mathcal{F}(Conv(1))$ then T_{costs} , $T_{env(0)}$, $T_{A(1)}$, $T_{FS(1)}$, $T_{AC(1)}$, and $T_{CD(1)}$ (without the condition on c_0) can be used.*

It is essential that the environment behaves independently of the queues. It would also be of interest to allow for, say, controlled server vacations. This can be done, in the case that we do not put a structure like convexity on the environment, with a *Markov Decision Arrival Process* (MDAP), as introduced in Hordijk & Koole [9] for routing to parallel queues. Unfortunately the MDAP cannot be used in the case of convexity. (For an example of the use of MDAP's in the context of event-based dp, see [11].) Another possibility is to incorporate the environment also in the inequalities which define $\mathcal{F}(\mathcal{P})$, giving it a two-dimensional structure. This type of inequalities is the subject of the next Section.

4 Models with two-dimensional inequalities

In line with the presentation of the previous Section, we start with presenting several operators which complement the set of one-dimensional operators already defined in the previous Section.

- $T_{ACF(I)}f(x) = \min\{c + f(x), f(x + \sum_{i \in I} e_i)\}$: admission control of a fork to all queues in the set I , $|I| > 0$.
- $T_{R(I)}f(x) = \min_{i \in I} f(x + e_i)$: routing to one of the queues in I , $|I| > 0$.
- $T_{MS(I)}f(x) = \min_{i \in I} f((x - e_i)^+)$: a movable server which can serve one of the queues in I , $|I| > 0$.
- $T_{CJ(i,j)}f(x) = \begin{cases} \min_{\mu \in [0,1]} \{c_\mu + \mu f(x - e_i + e_j) + (1 - \mu)f(x)\} & \text{if } x_i > 0 \\ c_0 + f(x) & \text{otherwise, } i \neq j: \text{ controlled jockeying.} \end{cases}$

We consider the following inequalities:

- *Super*(i, j): $f(x + e_i) + f(x + e_j) \leq f(x) + f(x + e_i + e_j)$ for all $x \in \mathbb{N}_0^m$: supermodularity.
- *Sub*(i, j): $f(x) + f(x + e_i + e_j) \leq f(x + e_i) + f(x + e_j)$ for all $x \in \mathbb{N}_0^m$: submodularity.
- *SuperC*(i, j): $f(x + e_i) + f(x + e_i + e_j) \leq f(x + e_j) + f(x + 2e_i)$ and $f(x + e_j) + f(x + e_i + e_j) \leq f(x + e_i) + f(x + 2e_j)$ for all $x \in \mathbb{N}_0^m$. This inequality is related to supermodularity and convexity, as we will see below. We call the property *superconvexity*, inspired by the terminology of Ghoneim & Stidham [5].
- *SubC*(i, j): $f(x + e_i) + f(x + e_i + e_j) \leq f(x) + f(x + 2e_i + e_j)$ and $f(x + e_j) + f(x + e_i + e_j) \leq f(x) + f(x + e_i + 2e_j)$ for all $x \in \mathbb{N}_0^m$. This inequality is related to submodularity and also convexity. We call it subconvexity.

$$\begin{array}{l}
\text{Conv: } r \not\leq r \text{ and } \begin{matrix} r \\ \not\leq \\ r \end{matrix} \quad \text{Super: } \begin{matrix} l & r \\ r & l \end{matrix} \quad \text{SuperC: } \begin{matrix} r & l \\ \cdot & l \end{matrix} r \text{ and } \begin{matrix} r \\ l \\ \cdot \\ r \end{matrix} \\
\text{Inc: } \quad l & r \text{ and } \begin{matrix} r \\ l \end{matrix} \quad \text{Sub: } \begin{matrix} r & l \\ l & r \end{matrix} \quad \text{SubC: } \quad r \begin{matrix} l \\ l \end{matrix} r \text{ and } \begin{matrix} r \\ l \\ l \\ r \end{matrix}
\end{array}$$

Figure 1. A graphical representation of the inequalities.

The inequalities are illustrated in Figure 1. The terms on the left (right) hand side of the \leq sign are indicated with l (r). It is easily seen that, using the Figure, combining *Super*(i, j) with *SuperC*(i, j) gives *Conv*(i) and *Conv*(j), convexity in i and in j . Combining *Super*(i, j) and *Conv*(i) and *Conv*(j) gives in its turn *SubC*(i, j). Also *Sub*(i, j) and *SubC*(i, j) combined gives convexity in both components, and *Sub*(i, j) and *Conv*(i) and *Conv*(j) gives *SuperC*(i, j).

Lemma 4.1 *The following relations hold, for all j and k separately, including i , unless otherwise indicated:*

- T_{costs} : $Inc(j) \rightarrow Inc(j)$; $Super(j, k) \rightarrow Super(j, k)$; $Sub(j, k) \rightarrow Sub(j, k)$;
 $SuperC(j, k) \rightarrow SuperC(j, k)$; $SubC(j, k) \rightarrow SubC(j, k)$.
- $T_{env(i)}$: $Inc(j) \rightarrow Inc(j)$ ($i \neq j$); $Super(j, k) \rightarrow Super(j, k)$ ($i \neq j, k$);
 $Sub(j, k) \rightarrow Sub(j, k)$ ($i \neq j, k$); $SuperC(j, k) \rightarrow SuperC(j, k)$ ($i \neq j, k$);
 $SubC(j, k) \rightarrow SubC(j, k)$ ($i \neq j, k$).
- $T_{A(i)}$: $Inc(j) \rightarrow Inc(j)$; $Super(j, k) \rightarrow Super(j, k)$; $Sub(j, k) \rightarrow Sub(j, k)$;
 $SuperC(j, k) \rightarrow SuperC(j, k)$; $SubC(j, k) \rightarrow SubC(j, k)$.
- $T_{FS(i)}$: $Inc(j) \rightarrow Inc(j)$; $Super(j, k) \rightarrow Super(j, k)$; $Sub(j, k) \rightarrow Sub(j, k)$.
- $T_{AC(i)}$: $Inc(j) \rightarrow Inc(j)$; $Super(j, k) \rightarrow Super(j, k)$ ($i = j, k$);
 $Sub(j, k) \rightarrow Sub(j, k)$ ($i = j, k$); $Super(j, k), SuperC(j, k) \rightarrow SuperC(j, k)$ ($i = j, k$);
 $Sub(j, k), SubC(j, k) \rightarrow SubC(j, k)$ ($i = j, k$).
- $T_{D(i)}$: $Inc(j) \rightarrow Inc(j)$; $Super(j, k) \rightarrow Super(j, k)$; $Sub(j, k) \rightarrow Sub(j, k)$;
 $Inc(j), Inc(k), Super(j, k), SuperC(j, k) \rightarrow SuperC(j, k)$ ($i = j, k$);
 $Inc(j), Inc(k), Sub(j, k), SubC(j, k) \rightarrow SubC(j, k)$ ($i = j, k$).
- $T_{MD(i)}$: $Inc(j) \rightarrow Inc(j)$; $Super(j, k) \rightarrow Super(j, k)$; $Sub(j, k) \rightarrow Sub(j, k)$.
- $T_{CD(i)}$: $Inc(i) \rightarrow Inc(i)$ if $c_0 = \min_{\mu \in [0,1]} c_\mu$; $Inc(j) \rightarrow Inc(j)$ ($j \neq i$);
 $Super(j, k) \rightarrow Super(j, k)$ ($i = j, k$); $Sub(j, k) \rightarrow Sub(j, k)$ ($i = j, k$);
 $Super(j, k), SuperC(j, k) \rightarrow SuperC(j, k)$ ($i = j, k$);
 $Sub(j, k), SubC(j, k) \rightarrow SubC(j, k)$ ($i = j, k$).
- $T_{ACF(I)}$: $Inc(j) \rightarrow Inc(j)$; $Sub(j, k), SubC(j, k) \rightarrow Sub(j, k)$ ($I = \{j, k\}$);
 $Sub(j, k), SubC(j, k) \rightarrow SubC(j, k)$ ($I = \{j, k\}$).
- $T_{R(I)}$: $Inc(j) \rightarrow Inc(j)$; $Super(j, k), SuperC(j, k) \rightarrow Super(j, k)$ ($I = \{j, k\}$);
 $Super(j, k), SuperC(j, k) \rightarrow SuperC(j, k)$ ($I = \{j, k\}$).
- $T_{MS(I)}$: $Inc(j) \rightarrow Inc(j)$; $Inc(j), Inc(k), Super(j, k), SuperC(j, k) \rightarrow Super(j, k)$ ($I = \{j, k\}$);
 $Inc(j), Inc(k), Super(j, k), SuperC(j, k) \rightarrow SuperC(j, k)$ ($I = \{j, k\}$).
- $T_{CJ(i,j)}$: $Inc(k) \rightarrow Inc(k)$ ($k \neq i$); $Inc(i), Inc(j) \rightarrow Inc(i)$ if $c_0 = \min_{\mu \in [0,1]} c_\mu$;
 $Super(i, j), SuperC(i, j) \rightarrow Super(i, j)$; $SuperC(i, j) \rightarrow SuperC(i, j)$.

The proof is tedious but straightforward, using the same ideas as employed for Lemma 3.1. We did not include T_{unif} , as it is a special case of $T_{env(i)}$.

We see that often $Super(i, j)$ and $SuperC(i, j)$ or $Sub(i, j)$ and $SubC(i, j)$ go together. In both cases convexity holds. Together with convexity (which is a one-dimensional property) we have either $Super(i, j)$ or $Sub(i, j)$: this indicates how the queues interact: attraction ($Sub(i, j)$) or distraction ($Super(i, j)$).

As for the one-dimensional equalities we consider one by one some relevant classes of functions. First we consider $\mathcal{F}(Inc(1), Inc(2), Super(1, 2), SuperC(1, 2))$. By Lemma 4.1 we derive the following result.

Theorem 4.2 *For an event-based value function V_n , constructed with some or all operators of the form T_{costs} , $T_{env(0)}$, $T_{A(1)}$ and $T_{A(2)}$, $T_{AC(1)}$ and $T_{AC(2)}$, $T_{R(1,2)}$, $T_{MS(1,2)}$, $T_{D(1)}$ and $T_{D(2)}$, $T_{CD(1)}$ and $T_{CD(2)}$ with $c_0 = \min_{\mu \in [0,1]} c_\mu$, and $T_{CJ(1,2)}$ and $T_{CJ(2,1)}$, also with $c_0 = \min_{\mu \in [0,1]} c_\mu$, we have that $V_n \in \mathcal{F} = \mathcal{F}(Inc(1), Inc(2), Super(1, 2), SuperC(1, 2))$ if $C \in \mathcal{F}$ (for C from T_{costs}) and $V_0 \in \mathcal{F}$.*

Before looking at the models that can be constructed from the list of operators of Theorem 4.2 we derive for the operators which are controllable the monotonicity results. As said, combining $Super(1, 2)$ and $SuperC(1, 2)$ gives convexity in both components. Therefore the optimal admission rule for $T_{AC(1)}$ is of threshold form in the variable in x_1 . The same holds for $T_{AC(2)}$ in the variable x_2 . From $Super(1, 2)$ it follows that $T_{AC(1)}$ is also of threshold form in x_2 , and vice versa. Thus both $T_{AC(1)}$ and $T_{AC(2)}$ have a decreasing switching curve, below which customers are admitted to the system. In fact, using $SuperC(1, 2)$ directly, these results can be strengthened. For $T_{AC(1)}$ it follows that if a customer is rejected in $x + e_2$, then it is also rejected in $x + e_1$. For $T_{AC(2)}$ holds that if a customer is rejected in $x + e_1$, then the same is done in $x + e_2$. Thus for $T_{AC(1)}$ ($T_{AC(2)}$) the switching curve is nowhere horizontal (vertical). The monotonicity results for T_R are obtained from $SuperC(1, 2)$. It is readily seen that there is an increasing switching curve above (below) which customers are assigned to queue 1 (2). The operators $T_{CD(1)}$ and $T_{CD(2)}$ can be seen as dual to $T_{AC(1)}$ and $T_{AC(2)}$, with corresponding results. For $T_{CJ(1,2)}$ we derive from $SuperC(1, 2)$ that the optimal control is increasing in x_1 and decreasing in x_2 . Thus we see again an increasing switching curve, above which jockeying occurs. For $T_{CJ(2,1)}$ there is also an increasing switching curve, *below* which jockeying occurs.

Note furthermore that $\bar{C} = (x_1 - B)^+K, (x_2 - B)^+K$ or $(x_1 + x_2 - B)^+K \in \mathcal{F}$. Thus the queues are allowed to have separate buffers or one joint buffer. Now we consider the models.

The routing model in Hajek [7] has the following value function (see Eq. (2.4) on p. 492):

$$V_{n+1} = T_{costs} \left(C, T_{unif} \left[T_{A(1)}V_n, T_{A(2)}V_n, T_{D(1)}V_n, T_{D(2)}V_n, T_{R(1,2)}V_n, \right. \right. \\ \left. \left. T_{MS(1,2)}V_n, T_{CJ(1,2)}V_n, T_{CJ(2,1)}V_n \right] \right),$$

with $c_\mu = 0$ in $T_{CJ(i,j)}$. All these operators are allowable, and $C(x) = c_1x_1 + c_2x_2$ with $c_0 \geq 0$ falls also in $\mathcal{F}(Inc(1), Inc(2), Super(1, 2), SuperC(1, 2))$, thus the monotonicity results as described above for the various operators hold. In [7] the same results are obtained. Event-based programming however directly gives many generalizations. For example, we can replace T_{unif} by $T_{env(0)}$ giving general arrival streams, but which also allows changing service rates. Of course we can add other operators from the list in Theorem 4.2. We can replace the dedicated arrival streams $T_{A(i)}$ by controlled arrivals. This can also be done for the routing operator: $T_{R(1,2)}$ is equivalent to $T_{A(1)}T_{CJ(1,2)}$ with $c_\mu = 0$; now replace $T_{A(1)}$ by $T_{AC(1)}$. This allows us to safely introduce one or two finite buffers.

Now take $V_{n+1} = T_{costs} \left(C, T_{unif} \left[T_{A(1)}V_n, T_{A(2)}V_n, T_{CJ(1,2)}V_n, T_{CJ(2,1)}V_n \right] \right)$, with c_μ not necessarily 0. This is the cycles of queues (in two dimensions) studied in Weber & Stidham [20]. They also consider a tandem system, which is the same model with $T_{CJ(2,1)}$ replaced by $T_{CD(2)}$, and $T_{A(1)}$ replaced by $T_{AC(1)}$. In fact, in [20] the inequalities $Super(1, 2)$ and $SuperC(1, 2)$ are shown to propagate, meaning that they took $\mathcal{F}(Super(1, 2), SuperC(1, 2))$ as class of allowable functions. Indeed, the considered operators propagate all, the complete list of possible operators is $T_{env(0)}$, $T_{A(1)}$ and $T_{A(2)}$, $T_{AC(1)}$ and $T_{AC(2)}$, $T_{R(1,2)}$, $T_{CD(1)}$ and $T_{CD(2)}$, and $T_{CJ(1,2)}$ and $T_{CJ(2,1)}$. The advantage is that we can take non-decreasing cost functions (in [20] they are separable, non-negative and convex, but not necessarily increasing), the disadvantage is that we cannot allow standard single server queues: the current analysis shows well at what price (increasing direct costs) one could add them. The relation between the models in [20] and [7] (i.e., the same underlying inequalities) is also observed in Stidham &

Weber [19]. In fact, $\mathcal{F}(\text{Super}(1, 2), \text{SuperC}(1, 2))$ is the set of two-dimensional *multimodular* functions (see Hajek [8] or [19]).

Another well-known model that can be solved with the theory developed so far is that of Lin & Kumar [12]. It concerns a single queue, with an additional (slower) server that can be used when needed. Customers cannot be sent back from the slower server, thus intuitively this server should be used when there is enough work in the system. In [12] it is shown that the optimal assignment policy is indeed of threshold type, using another technique, namely policy iteration. The value function is as follows: $V_{n+1} = T_{\text{costs}}(C + \bar{C}, T_{\text{unif}}[T_{A_1}T_{CJ(1,2)}V_n, T_{D_1}T_{CJ(1,2)}V_n, T_{D_2}T_{CJ(1,2)}V_n])$, with $c_\mu = 0$ in $T_{CJ(1,2)}$, $C = x_1 + x_2$, and $\bar{C} = (x_2 - 1)^+K$. Note that after each event in the system we can decide to move a customer to the second server. As the operators and the costs are allowable the optimality of a threshold policy for $x_2 = 0$ follows. If $x_2 > 0$ it is suboptimal to send a customer to the second server, due to \bar{C} . As for the previous models, we can allow for many generalizations, like a larger capacity in the second queue, admission control, dedicated arrivals to the second queue, higher holding costs in the second queue, general arrival streams, etc. A possible generalization we can not allow for is multiple servers at the second (and neither the first) queue. This is a possible explanation why this models is still unsolved, despite the research efforts from various researchers.

Although the force of event-based dp lies in its capacity to combine results from existing models to find results for new models, we summarize the main generalizations of existing models for which the value function falls into the current class of functions.

Corollary 4.3 *The following results can be obtained by event-based dp:*

- i) The optimality of an increasing switching curve in the routing model of Hajek [7]. Cost functions must be elements of $\mathcal{F}(\text{Inc}(1), \text{Inc}(2), \text{Super}(1, 2), \text{SuperC}(1, 2))$;*
- ii) The monotonicity of optimal service rates in the two-dimensional model of Weber & Stidham [20]. Cost functions must be elements of $\mathcal{F}(\text{Super}(1, 2), \text{SuperC}(1, 2))$. If the costs function is also increasing then uncontrolled service rates are also allowed;*
- iii) The optimality of a threshold policy in the single server queue with an extra server of Lin & Kumar [12]. Cost functions must be elements of $\mathcal{F}(\text{Inc}(1), \text{Inc}(2), \text{Super}(1, 2), \text{SuperC}(1, 2))$. Main generalizations for all three models consist of finite buffer(s), discrete time models, and more general cost functions.*

A disadvantage of taking $\mathcal{F}(\text{Inc}(1), \text{Inc}(2), \text{Super}(1, 2), \text{SuperC}(1, 2))$ is that it excludes $T_{MD(i)}$. Indeed, if we try to propagate $\text{SuperC}(1, 2)$, then we need $\text{Super}(1, 2)$ and $\text{Sub}(1, 2)$ at the same time. However, $T_{MD(i)}$ propagates through $\text{Super}(1, 2)$; this allows us to obtain some (although weak) results for resource sharing models.

Theorem 4.4 *For an event-based value function V_n , constructed with some or all operators of the form T_{costs} , $T_{\text{env}(0)}$, $T_{A(1)}$ and $T_{A(2)}$, $T_{AC(1)}$ and $T_{AC(2)}$, $T_{FS(1)}$ and $T_{FS(2)}$, $T_{D(1)}$ and $T_{D(2)}$, $T_{MD(1)}$ and $T_{MD(2)}$, $T_{CD(1)}$ and $T_{CD(2)}$, we have that $V_n \in \mathcal{F} = \mathcal{F}(\text{Super}(1, 2))$ if $C \in \mathcal{F}$ (for C from T_{costs}) and $V_0 \in \mathcal{F}$.*

Combining $T_{MD(i)}$ with $T_{AC(i)}$, and costs $\bar{C}(x) = (b_1x_1 + b_2x_2 - B)^+K$ gives us the stochastic knapsack, as introduced in Ross & Tsang [15]. Each customer of type i takes capacity b_i , of a total of B . An example is a telecommunication model where a trunk line is dynamically divided between different types of calls (for example voice and video, the latter

needing more bandwidth). In this type of resource sharing models it can be optimal to block a customer even if the capacity is not fully used as to keep capacity available for arrivals of the other type. Indeed, inequality $Super(1,2)$ shows us that the admission control for class 1 is decreasing in class 2 and v.v. As $SuperC(1,2)$ is not included we have no further results on the form of the switching curve, in general it is not monotone. See [1] for more results on this specific model.

A completely different class of models is those in which the state components are related by the submodularity property.

Theorem 4.5 *For an event-based value function V_n , constructed with some or all operators of the form T_{costs} , $T_{env(0)}$, $T_{A(1)}$ and $T_{A(2)}$, T_{ACF} , $T_{D(1)}$ and $T_{D(2)}$, $T_{CD(1)}$ and $T_{CD(2)}$, we have that $V_n \in \mathcal{F} = \mathcal{F}(Inc(1), Inc(2), Sub(1,2), SubC(1,2))$ if $C \in \mathcal{F}$ (for C from T_{costs}) and $V_0 \in \mathcal{F}$.*

Because convexity holds, the optimal admission control to a queue as a function of the queue length is still of threshold form. However, when looking at the admission control as a function of the other queue, the situation as compared with

$$\mathcal{F}(Inc(1), Inc(2), Super(1,2), SuperC(1,2))$$

is reversed, due to inequality $Sub(1,2)$. For $T_{AC(1)}$ ($T_{AC(2)}$) there is an increasing switching curve above (below) which customers are admitted. Also the effects of $T_{CD(i)}$ amount to balancing in some sense the two queues. The two queues “attract” each other. $T_{ACF(1,2)}$ has a decreasing switching curve below which customers are admitted. We can introduce finite buffers by taking $\bar{C} = (x_1 - B)^+K$ or $(x_2 - B)^+K$. As direct costs we can take $C(x) = c_1x_1 + c_2x_2$ with $c_i > 0$, but also $C(x) = \max\{x_1, x_2\}$. This cost function, together with the operators $T_{ACF(1,2)}$, $T_{CD(1)}$ and $T_{CD(2)}$ constitute a model of [2]. $\max\{x_1, x_2\}$ signifies the number of forked customers of which not all tasks have finished their service, assuming each queue works in FIFO order. Note that we can freely add $T_{A(i)}$ as operator. This could not be done in [2], which is due to the restrictive nature of the score state approach (see [6]) taken there.

This attraction phenomenon is also observed for telecommunication models, as noted at p. 386 of Key [10]. Here the set of functions $\mathcal{F}(Sub(1,2))$ is more appropriate, as this includes $T_{MD(i)}$ as well. A possible application is that of two trunk lines, with three classes of traffic, two dedicated streams, and one that utilizes a line on both trunks. The value function would have the form $V_{n+1} = T_{costs}\left(C, T_{unif}\left[T_{AC(1)}V_n, T_{AC(2)}V_n, T_{ACF(1,2)}V_n, T_{MD(1)}V_n, T_{MD(2)}V_n\right]\right)$. Note that because of the larger set of functions we have no results on the assignment of the fork (which was based on $SubC(1,2)$). From a modelling point of view it would perhaps be better to let both tasks of a fork leave the system together. This however would imply a three-dimensional state space.

5 Conclusion

We presented a general method to obtain monotonicity results for the control of queueing models. Using this method we generalized results from Lippman [13], Stidham [18], Hajek [7], Weber & Stidham [20], Lin & Kumar [12], Altman & Koole [2] and we analyzed several new models.

Many other operators and sets of functions can be analyzed in a similar way, leading to numerous other results. Some of it has been reported in Koole [11]; research on other models is going on.

References

- [1] E. Altman, T. Jiménez, and G.M. Koole. On optimal call admission control. In *Proceedings of the 37th IEEE Conference on Decision and Control*, 1998. To appear.
- [2] E. Altman and G.M. Koole. On submodular value functions and complex dynamic programming. *Stochastic Models*, 14(5), 1998. To appear.
- [3] S. Asmussen and G.M. Koole. Marked point processes as limits of Markovian arrival streams. *Journal of Applied Probability*, 30:365–372, 1993.
- [4] R. Cavazos-Cadena and L.I. Sennott. Comparing recent assumptions for the existence of average optimal stationary policies. *Operations Research Letters*, 11:33–37, 1992.
- [5] H.A. Ghoneim and S. Stidham, Jr. Control of arrivals to two queues in series. *European Journal of Operations Research*, 21:399–409, 1985.
- [6] P. Glasserman and D.D. Yao. Monotone optimal control of permutable GSMPs. *Mathematics of Operations Research*, 19:449–476, 1994.
- [7] B. Hajek. Optimal control of two interacting service stations. *IEEE Transactions on Automatic Control*, 29:491–499, 1984.
- [8] B. Hajek. Extremal splitting of point processes. *Mathematics of Operations Research*, 10:543–556, 1985.
- [9] A. Hordijk and G.M. Koole. On the assignment of customers to parallel queues. *Probability in the Engineering and Informational Sciences*, 6:495–511, 1992.
- [10] P. Key. Some control issues in telecommunications networks. In F.P. Kelly, editor, *Probability, Statistics and Optimisation*. Wiley, 1994.
- [11] G.M. Koole. Stochastic scheduling with event-based dynamic programming. In W.K. Klein Haneveld, O.J. Vriete, and L.C.M. Kallenberg, editors, *Ten years LNMB*, pages 161–169. CWI, Amsterdam, 1997. CWI Tract 122.
- [12] W. Lin and P.R. Kumar. Optimal control of a queueing system with two heterogeneous servers. *IEEE Transactions on Automatic Control*, 29:696–703, 1984.
- [13] S.A. Lippman. Applying a new device in the optimization of exponential queueing systems. *Operations Research*, 23:687–710, 1975.
- [14] R. Ramjee, R. Nagarajan, and D. Towsley. On optimal call admission control in cellular networks. In *Proceedings of IEEE Infocom '96*, pages 43–50, 1996.
- [15] K.W. Ross and D.H.K. Tsang. The stochastic knapsack problem. *IEEE Transactions on Communications*, 37:740–747, 1989.

- [16] M. Schäl. Conditions for optimality in dynamic programming and for the limit of n -stage optimal policies to be optimal. *ZOR - Mathematical Methods of Operations Research*, 32:179–196, 1975.
- [17] R.F. Serfozo. An equivalence between continuous and discrete time Markov decision processes. *Operations Research*, 27:616–620, 1979.
- [18] S. Stidham, Jr. Socially and individually optimal control of arrivals to a $GI|M|1$ queue. *Management Science*, 24:1598–1610, 1970.
- [19] S. Stidham, Jr. and R.R. Weber. A survey of Markov decision models for control of networks of queues. *Queueing Systems*, 13:291–314, 1993.
- [20] R.R. Weber and S. Stidham, Jr. Optimal control of service rates in networks of queues. *Advances in Applied Probability*, 19:202–218, 1987.