Computers Math. Applic. Vol. 26, No. 6, pp. 141–148, 1993
Printed in Great Britain. All rights reserved

0898-1221/93 $6.00 + 0.00

Stochastic Scheduling Games with Markov Decision Arrival Processes

E. ALTMAN
INRIA-Sophia Antipolis
2004 Route des Lucioles, 06565 Valbonne Cedex, France

G. KOOLE
CWI, P.O. Box 4079
1009 AB Amsterdam, The Netherlands

Abstract—In Hordijk and Koole [1,2], a new type of arrival process, the Markov Decision Arrival Process (MDAP), was introduced, which can be used to model certain dependencies between arrival streams and the system at which the arrivals occur. This arrival process was used to solve control problems with several controllers having a common objective, where the output from one controlled node is fed into a second one, as in tandems of multi-server queues. In the case that objectives of the controllers are different, one may choose a min-max (worst case) approach where typically a controller tries to obtain the best performance under the worst possible (unknown) strategies of the other controllers. We use the MDAP to model such situations, or situations of control in an unknown environment. We apply this approach to several scheduling problems, including scheduling of customers and scheduling of servers. We consider different information patterns including delayed information. For all these models, we obtain several structural results of the optimal policies.

1. INTRODUCTION

Recently Hordijk and Koole [1,2] introduced the *Markov Decision Arrival Process* (MDAP), a Markovian arrival process by which not only independent arrivals can be modeled, but also arrival processes which depend in a certain way on the system into which the customers arrive. An MDAP is a generalization of the well-known Markov Arrival Process (MAP), which is a Markov process where arrivals can occur at the transitions. The MDAP generalizes the MAP by allowing the transition rates and arrival probabilities to be controlled dynamically, i.e., the transition rates and arrival probabilities are functions of actions that are sequentially chosen by a controller. The transition rates of the MDAP are independent of the system the customers arrive at; the actions, however, can depend on it. Thus, through the actions, the dependence is modeled.

A typical example of the use of an MDAP concerns the following tandem model. Customers arrive according to a Poisson process at m parallel $M|M|1$ queues. Dynamically, the customers have to be assigned to one of the queues. After being served, the customers arrive at a second station, where we have again m parallel queues to choose from. The question is how to assign the customers at the second center (for example, in the case that all service parameters are equal), assuming that the first center is operated optimally. In general, the optimal action in the first center will not only depend on the state of the first center, but also on the state of the second center. Thus, the arrival process at the second center depends, through the actions in the first center, on the state of the second center. Therefore, we cannot use the standard results on the optimality of shortest queue routing for independent arrivals. Hordijk and Koole [1] show that the arrivals from the first center can be modeled as an MDAP, and that for this type of arrivals

---

This research is supported by the European Grant BRA-QMIPS of CEC DG XIII.

Typeset by $\mathcal{A}_{\mathcal{M}}\mathcal{S}$-TEX

(in the case of equal service times), shortest queue routing at the second center is again optimal, for various objective functions. This result is proven using dynamic programming, where the action consists of two components, *viz.*, the action in the MDAP and the assignment to one of the parallel queues (of the second center). This can also be seen as two controllers, one at the MDAP and one at the queues, who are cooperating.

In this paper, we consider the situation where the controllers do not cooperate and might have different objectives. If the second controller does not know the objective of the controller of the MDAP, then he may still try to use a min-max approach, i.e., to design a control strategy that guarantees the best performance under the worst possible strategy of the MDAP controller. This naturally leads to a zero-sum stochastic game, where the MDAP controller plays against the second controller.

Another possible motivation for this model is when there is only one controller of a queueing system; the arrival process to that system is characterized by some parameters that may change in time in a way unpredictable by the controller. The controller may wish to design a control strategy that guarantees the best performance under the worst possible (time dependent) parameters of the arrival process. Here again, we end up with a zero-sum game between the MDAP (player 1), that models the arrival process, and the controller (player 2).

The use of the MDAP in the setting of a zero-sum game allows us to obtain structural results for the optimal strategy of player 2. We illustrate this by two examples where the optimal min-max strategies of player 2 are in fact explicitly obtained. In the next section, we will show the optimality result for the asymmetric model of [1] for scheduling of customers, of which the optimality of shortest queue routing (known as SQP) is a special case. We use a model known as a "stochastic game with complete information," for which deterministic policies exist for both players (see, e.g., the survey by Raghavan and Filar [3]).

In Section 3, we extend this result to the case of control with delayed information. More specifically, we first consider the problem where the state of the MDAP reaches the controller (player 2) after some delay. This, too, results in a "stochastic game with complete information." We then study the case where the information on both the state and the action of the MDAP is delayed. This results in a standard stochastic game for which the optimal policy for both players may need randomization.

In Section 4, we consider the model of [2] where one or more servers are to be assigned to customers of different classes. Here the results for player 2 are different than in the case of cooperation between the players, but in the same spirit.

We finally mention other references where stochastic games were applied in queueing models and the structure of optimal policies was obtained. Altman and Shimkin considered in [4] a nonzero-sum game with an infinite number of players to solve the problem of choosing between the use of an individual personal computer and a central computer, whose capacity is simultaneously shared between different users. Using coupling and sample-path methods, all Nash optimal policies were shown to be of threshold type. This then enabled the computation of an optimal threshold. Hsiao and Lazar [5] obtained threshold equilibrium policies for a decentralized flow control into a network using the product form of the network as well as Norton's equivalent. The threshold policy is then obtained through a Linear Program. Other results on flow control problems under worst case service conditions have recently been obtained by Altman [6,7] using tools from zero-sum stochastic games. Küenle [8] used dynamic programming, and especially value iteration, to solve an inventory control problem under worst case demand conditions. He modeled the problem as a stochastic zero-sum game with full information and identified the structure of an optimal policy of the controller, known as the (s,S) policy. For a recent survey on stochastic games, see Raghavan and Filar [3].

2. RESULTS ON THE CUSTOMER ASSIGNMENT MODEL

We start by formulating the model of the stochastic game for the uniformized model.

We consider a state space given by a product of two spaces: the state of the MDAP $\mathbf{X}$, assumed to be finite, times the state of the queueing system $\mathbf{L} = \prod_{j=1}^{M} \mathbf{L}_j$, where $\mathbf{L}_j = \{0, 1, \dots, L_j\}$, and $L_j > 0$ is the size of queue j (which may be either finite or infinite). Let $L = (L_1, \dots, L_j)$ be the vector of queue lengths. A typical element of the state space is denoted by (x, i), with $x \in \mathbf{X}$ and $i = (i_1, \dots, i_m) \in \mathbf{L}$ the number of customers in the m queues *including* the ones in service. Let e_i denote an m-dimensional vector with all entries zero except for the i^{th} entry, which is one.

The probability of a successful service in queue j is μ_j. Without loss of generality, we assume that $\mu_1 \geq \cdots \geq \mu_m$.

The finite space of actions of the MDAP (player 1) is $\mathbf{A}$ (different actions may be available in different states). λ_{xay} is the probability that the MDAP moves from x to y if action a was chosen, and q_{xay} is the probability that a customer arrives if the arrival process moves from x to y using action a. We assume, without loss of generality, that for any (x, i) and a, $\sum_y \lambda_{xay} + \sum_{j=1}^{m} \mu_j = 1$.

The finite space of actions available to the second player (controller) is $\mathbf{B} = \{1, \dots, m\}$. An action $b \in \mathbf{B}$ has the meaning of assigning a customer to queue b. However, we assume that if a queue is full, then a customer cannot be assigned to it. If all queues are full, then the customer is lost. We assume in this section that player 2 takes an action immediately after an arrival occurs (hence, after a transition in the MDAP occurs), already knowing the new state of the MDAP.

A precise description of the decision process (transition probabilities and the state space) is given in [9, Section 5.1]. (The state should in fact include a mark of whether an arrival just occurred. Only at such states can player 2 take actions, whereas player 1 can take actions in the remaining states. The resulting game is then seen to be in fact a "complete information" game. It is known that for these games there exist optimal policies which do not require randomizations.) In our model, it will, however, suffice to construct the dynamic programming equation.

Let U be the set of policies for player 1 and W the set of policies for player 2. Let $(X(s), I(s))$, $A(s), B(s)$, $s = 0, \dots, n$, denote the state and action processes.

The cost for a horizon of length n, for an initial state (x, i) and policies u, w is denoted by $v^n_{(x,i,u,w)}$. We assume that there is a terminal cost $v^0_{(x,i)} = v^0_{(x,i,u,w)}$ that satisfies the following equations (where $n = 0$):

$$v^n_{(x,i+e_{j_1})} \leq v^n_{(x,i+e_{j_2})} \quad \text{if } i_{j_1} \leq i_{j_2},\ j_1 \leq j_2,\ i + e_{j_1} + e_{j_2} \leq L \tag{1}$$

$$v^n_{(x,i)} \leq v^n_{(x,i+e_j)} \quad \text{if } i + e_j \leq L \tag{2}$$

$$v^n_{(x,i)} \leq v^n_{(x,i^*)} \quad \text{if } \begin{cases} \max(i_{j_1}, i_{j_2}) \leq \max(L_{j_1}, L_{j_2}), \\ i_{j_1} > i_{j_2},\ j_1 \leq j_2 \end{cases} \text{and } i^*_j = \begin{cases} i_j & \text{if } j \neq j_1, j_2, \\ i_{j_2} & \text{if } j = j_1, \\ i_{j_1} & \text{if } j = j_2. \end{cases} \tag{3}$$

In particular, we may choose $v^0_{(x,i)} = v^0_{(x,i,u,w)} = \sum_{j=1}^{m} i_j$. $v^n_{(x,i,u,w)}$ is given by $v^n_{(x,i,u,w)} = E^{u,w}_{(x,i)} v^0_{(X(n),I(n))}$.

The objective of the controller (player 2) is (P0): find a policy w^* that achieves

$$\sup_{u \in U} v^n(x, i, u, w) \geq \sup_{u \in U} v^n(x, i, u, w^*) =: v^n_{(x,i)}, \qquad \forall w \in W.$$

It is well-known that $v^n_{(x,i)} = \sup_{u \in U} \inf_{w \in W} v^n_{(x,i,u,w)}$, and moreover, there is a pair of policies (u^*, w^*) for the two players such that

$$\inf_{w \in W} v^n_{(x,i,u^*,w)} = v^n_{(x,i,u^*,w^*)} = v^n_{(x,i)}.$$

A policy for player 2 is called Shorter Queue Faster Server Policy (SQFSP) if it satisfies the following property: if $i_{j_1} < i_{j_2}$ and $j_1 < j_2$ (and $i + e_{j_1} + e_{j_2} \leq L$), then an arriving customer will not be assigned to queue j_2. In particular, if the fastest queue has the smallest number of customers, then w^* sends an arriving customer to it. This gives the optimality of the SQP in the symmetric case.

THEOREM 1. *There is an optimal policy w^* for player 2 which is SQFSP.*

PROOF. We formulate the dynamic programming equation. If the system is not full, then

$$v^{n+1}_{(x,i)} = \max_a \left\{ \sum_y \lambda_{xay} \left(q_{xay} \min_b \{v^n_{(y,i+e_b)}\} + (1 - q_{xay}) v^n_{(y,i)} \right) \right\} + \sum_{j=1}^{m} \mu_j v^n_{(x,(i-e_j)^+)}. \tag{4}$$

When the system is full, then we have:

$$v^{n+1}_{(x,i)} = \max_a \left\{ \sum_y \lambda_{xay} v^n_{(y,i)} \right\} + \sum_{b=1}^{m} \mu_b v^n_{(x,(i-e_b))}.$$

We show by induction on n that v^n satisfies (1), (2), and (3). The optimality of SQFSP follows from (1) and is seen to be independent of the action in the MDAP.

The terms in (4) corresponding to the departures $\sum_{j=1}^{m} \mu_j v^n_{(x,(i-e_j)^+)}$ satisfy the properties following the same arguments as in the proof of Theorem 3.1 in [1]. We show how the proof for the other term, corresponding to the arrivals, deviates from the proof of Theorem 3.1 in [1]. We begin with (1). Introduce the following notation:

$$f(i,a) = \sum_y \lambda_{xay} \left(q_{xay} \min_j \{v^n_{(y,i+e_j)}\} + (1 - q_{xay}) v^n_{(y,i)} \right).$$

It is shown in [1] that $f(i + e_{j_1}, a) \le f(i + e_{j_2}, a)$ for all a and suitable j_1 and j_2 (i.e., as specified in equation (1)). If a^* is the minimizing action for the MDAP in state $(x, i + e_{j_2})$ (note that in [1] both the MDAP and the controller minimize), then the proof in [1] proceeds as follows:

$$\min_a f(i + e_{j_1}, a) \le f(i + e_{j_1}, a^*) \le f(i + e_{j_2}, a^*) = \min_a f(i + e_{j_2}, a).$$

This can easily be adapted to the current setting. Let a^* be maximizing action of the MDAP (player 1) in $(x, i + e_{j_1})$. Then

$$\max_a f(i + e_{j_1}, a) = f(i + e_{j_1}, a^*) \le f(i + e_{j_2}, a^*) \le \max_a f(i + e_{j_2}, a),$$

which establishes (1). The remaining proofs of (2) and (3) are similar. ∎

REMARK 1. Note that there can be more than one optimal policy. This happens, for example, in the case of a symmetric model (i.e., $\mu_1 = \cdots = \mu_m$) if there is more than one shortest queue. It is also possible that there is an optimal policy which is not a SQFSP. This happens, for example, in the trivial case that $v^0_{(x,i)} = 0$ for all x and i. However, there is always at least one optimal policy which is SQFSP. This explains the formulation of Theorem 1.

3. CUSTOMER ASSIGNMENT MODEL WITH DELAYED INFORMATION

We consider the same model as in the previous section with one exception. Player 2 takes an action immediately after an arrival occurs; however, due to information delay, it does not have the knowledge of the new state of the MDAP. As a result, we may consider this action to have been taken already prior to the arrival (since no new information is obtained by player 2 in the arrival epoch).

We shall thus assume that the decision instants for the players are the same; each time a transition occurs (departure or a transition in the MDAP), both players take a decision. The decision of player 2 should be interpreted, however, as the action to be taken when there will be a future arrival.

We further consider two versions of that game, depending on whether or not the information on the action of player 1 is delayed too.

(P1) When a customer arrives, then player 2 already has the information on the last action of player 1. Hence, at each decision epoch, player 1 takes a decision first, and only then player 2 takes a decision, knowing the decision of player 1.

(P2) When a customer arrives, then player 2 does not yet have the information on the last action of player 1. Hence, at each decision epoch, the players take their actions independently.

To summarize, the information available to each player at a given decision epoch consists of all previous states and actions of both players, as well as the current state of the system. Moreover, in (P1), at any time n, player 2 has the information on the decision of player 1 at time n. Problem (P1) is known as a stochastic game with complete information. It is known that for these games there exist optimal policies which do not require randomizations (for both players), whereas in (P2), randomized policies are usually needed to obtain optimality. Since the action of player 2 is interpreted as the decision to be taken when a future arrival occurs, the knowledge of the current state indeed grasps the fact that information is delayed, and thus, when that arrival will occur and the MDAP will change its state, the new state will not be available to player 2.

Since the amount of information that player 2 possesses in (P1) when making a decision is less than in (P2), and that is less of the information he has in (P0) (of the previous section), the value v^n will satisfy

$$v^n_{(P0)} \le v^n_{(P1)} \le v^n_{(P2)}.$$

The transition probabilities (for all three scenarios) are given by:

$$\mathcal{P}_{(x,i),a,b,(y,k)} = \begin{cases} \lambda_{xay} q_{xay} & \text{if } k = i + e_b, \\ \lambda_{xay}(1 - q_{xay}) + \sum_{j=1}^{m} \mu_j 1\{i_j = 0, y = x\} & \text{if } k = i, \\ \mu_j & \text{if } y = x,\ k = i - e_j,\ i_j > 0, \\ 0 & \text{otherwise.} \end{cases}$$

(As in the previous section, we assume that the rates are already normalized so for any (x,i), a, and b, we have $\sum\limits_{y,k} \mathcal{P}_{(x,i),a,b,(y,k)} = 1$.) If all queues are full, then

$$\mathcal{P}_{(x,i),a,b,(y,k)} = \begin{cases} \lambda_{xay} & \text{if } k = i, \\ \mu_j & \text{if } y = x,\ k = i - e_j, \\ 0 & \text{otherwise.} \end{cases}$$

For a set Π, let $M(\Pi)$ be the set of probability measure over Π. For a function (matrix) $f : \mathbf{A} \times \mathbf{B} \to \mathbb{R}$ and $\alpha \in M(\mathbf{A})$, $\beta \in M(\mathbf{B})$, we denote

$$\begin{aligned} f(\alpha, b) &:= \int_{\mathbf{A}} f(a,b)\alpha(da), \\ f(a, \beta) &:= \int_{\mathbf{B}} f(a,b)\beta(db), \\ f(\alpha, \beta) &:= \int_{\mathbf{B}} \int_{\mathbf{A}} f(a,b)\alpha(da)\beta(db). \end{aligned}$$

Let $\operatorname{val} f$ denote the value of the matrix game f, which is given by

$$\operatorname{val} f = \sup_{\alpha \in M(\mathbf{A})} \inf_{\beta \in M(\mathbf{B})} f(\alpha, \beta).$$

val f is known to satisfy

$$\operatorname{val} f = \inf_{\beta \in M(\mathbf{B})} \sup_{\alpha \in M(\mathbf{A})} f(\alpha, \beta).$$

Moreover, there exists a pair (α^*, β^*), $\alpha^* \in M(\mathbf{A})$, $\beta^* \in M(\mathbf{B})$, such that

$$\operatorname{val} f = \inf_{\beta \in M(\mathbf{B})} f(\alpha^*, \beta) = \sup_{\alpha \in M(\mathbf{A})} f(\alpha, \beta^*) = f(\alpha^*, \beta^*).$$

(α^*, β^*) are said to be optimal for the matrix game f.

THEOREM 2. *Consider problems (P1) and (P2). For each one of these problems, there is an optimal policy w^* for player 2 which is SQFSP.*

PROOF. We formulate the dynamic programming equation for the two problems. If the system is not full, then

$$\text{(P1)}: \qquad v^{n+1}_{(x,i)} = \max_a \min_b \left\{ \sum_y \lambda_{xay} \left(q_{xay} v^n_{(y,i+e_b)} + (1 - q_{xay}) v^n_{(y,i)} \right) \right\} + \sum_{j=1}^m \mu_j v^n_{(x,(i-e_j)^+)},$$

$$\text{(P2)}: \qquad v^{n+1}_{(x,i)} = \operatorname{val} \left\{ \sum_y \lambda_{xay} \left(q_{xay} v^n_{(y,i+e_b)} + (1 - q_{xay}) v^n_{(y,i)} \right) \right\} + \sum_{j=1}^m \mu_j v^n_{(x,(i-e_j)^+)}.$$

When the system is full, then we have for both problems:

$$v^{n+1}_{(x,i)} = \max_a \left\{ \sum_y \lambda_{xay} v^n_{(y,i)} \right\} + \sum_{b=1}^m \mu_b v^n_{(x,(i-e_b))}.$$

We show by induction on n that v^n satisfies (1), (2), and (3). Property (1) will then enable us to obtain the optimality of SQFSP.

The terms for (P1) and (P2) corresponding to the departures, $\sum_{j=1}^m \mu_j v^n_{(x,(i-e_j)^+)}$, satisfy the properties following the same arguments as in the proof of Theorem 3.1 in [1].

We show that the arrival terms in the expression for v^{n+1} of (P1) satisfy (1). Let

$$f(i, a) = \min_b \left\{ \sum_y \lambda_{xay} \left(q_{xay} v^n_{(y,i+e_b)} + (1 - q_{xay}) v^n_{(y,i)} \right) \right\}.$$

Let b^* be the minimizing action for $f(i + e_{j_2}, a)$. We show that $f(i + e_{j_1}, a) \le f(i + e_{j_2}, a)$ for all a and suitable j_1 and j_2 (as specified by equation (1)). First consider the case $b^* = j_1$. Then

$$f(i + e_{j_1}, a) \le \sum_y \lambda_{xay} \left(q_{xay} v^n_{(y,i+e_{j_1}+e_{j_2})} + (1 - q_{xay}) v^n_{(y,i+e_{j_1})} \right) \le f(i + e_{j_2}, a),$$

the last inequality following by (1). If $b^* \ne j_1, j_2$ then

$$f(i + e_{j_1}, a) \le \sum_y \lambda_{xay} \left(q_{xay} v^n_{(y,i+e_{j_1}+e_{b^*})} + (1 - q_{xay}) v^n_{(y,i+e_{j_1})} \right) \le f(i + e_{j_2}, a),$$

the last inequality following again by (1). By (1) we can choose $b^* \ne j_2$. This establishes that $f(i + e_{j_1}, a) \le f(i + e_{j_2}, a)$ for all a and suitable j_1 and j_2. Using the same arguments as in the proof of Theorem 1, we conclude that (1) holds for the arrival term in the expression for v^{n+1}.

Now consider problem (P2). Define, for $\alpha \in M(\mathbf{A})$,

$$\begin{aligned} \tilde{f}(i, \alpha) &= \min_\beta \left\{ \sum_b \beta(b) \sum_a \alpha(a) \sum_y \lambda_{xay} \left(q_{xay} v^n_{(y,i+e_b)} + (1 - q_{xay}) v^n_{(y,i)} \right) \right\} \\ &= \min_b \left\{ \sum_a \alpha(a) \sum_y \lambda_{xay} \left(q_{xay} v^n_{(y,i+e_b)} + (1 - q_{xay}) v^n_{(y,i)} \right) \right\}. \end{aligned} \tag{5}$$

Similarly to the previous case, we can show that $\tilde{f}(i+e_{j_1},\alpha) \le \tilde{f}(i+e_{j_2},\alpha)$, and therefore, (1) holds for the arrival term in the expression for v^{n+1}.

It can be shown similarly that v^{n+1} satisfies (2) and (3) for both models (P1) and (P2). Hence, we established by induction that v^n satisfies (1), (2), and (3) for all n, for both models (P1) and (P2).

For problem (P1), the optimality of SQFSP follows immediately from (1) and is seen to be independent of the action in the MDAP. Consider (P2) and choose $\beta \in M(\mathbf{B})$. Suppose $\beta(j_2) > 0$, while there is a j_1 such that j_1 and j_2 satisfy the conditions of equation (1). Then the term in the first curly bracket in equation (5) does not increase, for each policy α of player 1 by using $\tilde{\beta}$, with $\tilde{\beta}(j) = \beta(j)$ if $j \ne j_1, j_2$, $\tilde{\beta}(j_1) = \beta(j_1) + \beta(j_2)$, and $\tilde{\beta}(j_2) = 0$. Hence, we obtain the optimality of a SQFSP for player 2. ∎

REMARK 2. In general, in (P2), both players need to randomize, but player 2 can restrict to actions which belong to a SQFSP policy. However, in some states (in some models like the symmetric, in all states), player 2 uses one action with probability 1. In these states, player 1 does not randomize either.

REMARK 3. In [1], model (P0) was analyzed in an MDP framework, i.e., both players cooperate in order to minimize the (common) cost. (P1) and (P2) could also be considered in such a framework and would yield the optimality of the SQFSP policy by the same technique as we used in this section. Note, however, that in the MDP case, (P1) and (P2) have the same optimal values and optimal (deterministic) policies. Indeed, since in the MDP case an optimal deterministic policy is known to exist, the controllers need not be told the information on the actions of each other in order to know them accurately, since these actions can be deduced directly from the state of the system.

4. RESULTS ON THE SERVER ASSIGNMENT MODEL

In [2], both multiple and single server systems are studied. We start with the single server model. Customers arrive according to an MDAP (again with transition probabilities λ_{xay}) in m queues, where q^j_{xay} is the probability of an arrival in queue j. Customers in queue j have an exponential service time distribution with parameter μ_j. In this model, there is a single server, and player 2 has to decide to which queue the server will be assigned.

We shall use the same notation as in Sections 2 and 3; the only difference is that this time the meaning of an action $b \in \mathbf{B}$ of player 2 is to assign the server to queue b. Each player takes decisions based on the history of previous states and actions as well as the current state. Denote $\mu = \max_j \mu_j$. We get the following dynamic programming equation:

$$v^{n+1}_{(x,i)} = \max_a \left\{ \sum_y \lambda_{xay} \left(\sum_{j=1}^m q^j_{xay} v^n_{(y,i+e_j)} + \left(1 - \sum_{j=1}^m q^j_{xay} \right) v^n_{(y,i)} \right) \right\}$$
$$+ \min_b \left\{ \mu_b v^n_{(x,i-e_b)} + (\mu - \mu_b) v^n_{(x,i)} \right\}.$$

Note that the dynamic programming equation resembles the one for (P1) or (P2) in the previous section rather than (P0). This is related to the fact that in (P0) in Section 2, player 2 could take an action only immediately after an arrival occurred (and a transition in the state of the MDAP).

For independent arrivals and linear costs, i.e., $v^0_{(x,i)} = \sum_j c_j i_j$, the μc-rule [2] is known to be optimal. Reorder the queues such that $\mu_1 c_1 \ge \cdots \ge \mu_m c_m$. For arrivals according to an MDAP, the extra condition $\mu_1 \le \ldots \mu_m$ was needed in [2]. In the present setting, where maximizing actions are chosen in the MDAP, we have to assume $\mu_1 \ge \cdots \ge \mu_m$ instead. Indeed, under this assumption, we can rewrite the following basic inequality for the proof of optimality:

$$\mu_{j_1} v^n_{(x,i-e_{j_1})} + (\mu - \mu_{j_1}) v^n_{(x,i)} \le \mu_{j_2} v^n_{(x,i-e_{j_2})} + (\mu - \mu_{j_2}) v^n_{(x,i)} \qquad \text{for } j_1 < j_2$$

as

$$\mu_{j_1} v^n_{(x,i-e_{j_1})} \leq \mu_{j_2} v^n_{(x,i-e_{j_2})} + (\mu_{j_1} - \mu_{j_2}) v^n_{(x,i)} \qquad \text{for } j_1 < j_2.$$

Similarly, as in the previous section, if we write

$$f(i,a) = \min_a \left\{ \sum_y \lambda_{xay} \left(\sum_{j=1}^{m} q^j_{xay} v^n_{(y,i+e_j)} + (1 - \sum_{j=1}^{m} q^j_{xay}) v^n_{(y,i)} \right) \right\}$$

and if a^* is the maximizing action in $(x, i - e_{j_1})$, then

$$\begin{aligned} \mu_{j_1} \max_a f(i - e_{j_1}, a) &= \mu_{j_1} f(i - e_{j_1}, a^*) \leq \mu_{j_2} f(i - e_{j_2}, a^*) + (\mu_{j_1} - \mu_{j_2}) f(i, a^*) \\ &\leq \mu_{j_2} \max_a f(i - e_{j_2}, a) + (\mu_{j_1} - \mu_{j_2}) \max_a f(i, a), \end{aligned}$$

proving the optimality of the μc-rule if $\mu_1 \geq \cdots \geq \mu_m$. For the multiple server case, $\mu_1 \leq \cdots \leq \mu_m$ is also required for proving the inequality for the terms concerning service of customers. Thus, in this case, the μc-rule is only optimal if $\mu_1 = \cdots = \mu_m$.

REFERENCES

1. A. Hordijk and G. Koole, On the assignment of customers to parallel queues, *Probability in the Engineering and Informational Sciences* **6**, 495–511 (1992).
2. A. Hordijk and G. Koole, On the optimality of LEPT and μc rules for parallel processors and dependent arrival processes, Technical Report TW-92-01, University of Leiden, (1992); *Advances in Applied Probability*, (December 1993) (to appear).
3. T.E.S. Raghavan and J.A. Filar, Algorithms for stochastic games—A survey, *Zeitschrift für Operations Research* **35**, 437–472 (1991).
4. E. Altman and N. Shimkin, Individually optimal dynamic routing in a processor sharing system: Stochastic game analysis, EE Publication No. 849, Technion, (1992); *IEEE Transactions on Automatic Control*, (submitted).
5. M.T. Hsiao and A.A. Lazar, Optimal decentralized flow control of Markovian queueing networks with multiple controllers, CTR Technical Report CUCTR-TR-19, Columbia University, (1986).
6. E. Altman, Flow control using the theory of zero-sum Markov games, In *Prodeedings of the 31st IEEE Conference on Decision and Control*, Tucson, Arizona, December 1992, (1992).
7. E. Altman, Monotonicity of optimal policies in a zero sum game: A flow control model, In *Proceedings of the 5th Symposium on Dynamic Games and Applications*, Grimentz, July 1992, (1992); *Annals of Dynamic Games* (submitted).
8. H.U. Küenle, On the optimality of (s,S)-strategies in a minimax inventory model with average cost criterion, *Optimization* **22**, 123–138 (1991).
9. G. Koole, Stochastic scheduling and dynamic programming, Ph.D. Thesis, (on request available from the author), University of Leiden, (1992).