Adv. Appl. Prob. **24,** 234–237 (1992)
Printed in N. Ireland

THE μc-RULE IS NOT OPTIMAL IN THE SECOND NODE OF THE TANDEM QUEUE: A COUNTEREXAMPLE

ARIE HORDIJK* AND
GER KOOLE,* *University of Leiden*

Abstract

In this note we give a counterexample which shows that the μc-rule is not optimal in the second node of the tandem queue. This counterexample contradicts the interchange argument in Nain [1] and Nain et al. [2].

STOCHASTIC SCHEDULING

AMS 1991 SUBJECT CLASSIFICATION: PRIMARY 90B25
SECONDARY 90C40

1. Introduction

We consider two queueing centres in tandem. Both centres have one exponential server. There are M customer classes with different exponential service times and holding cost rates. Customers served at the first centre join the second centre and keep their class number.

It is well known that in case of one node the μc-rule minimizes the total discounted holding cost. Clearly, this is not always the case in the first node of the tandem model if the decision rule may also depend on the number of customers in the second centre. For example, suppose that the holding cost rates in the second node are equal to those of the first node and suppose there are many customers of any type in the first centre. If the second centre is empty then a good decision rule in node 1 will serve customers with a large parameter of the exponential distribution even if the μc-rule would assign the server to jobs with a small parameter. Indeed, in this way the rate of the customer stream from node 1 to node 2 will be higher and hence the server in node 2 will start serving earlier. Consequently, the departure process of the network is faster and hence the reduction of the total holding cost rate of the network can be larger.

The holding cost reduction in the second node is similar to that of an isolated centre. Hence it seems plausible that the optimal decision rule in the second node is the μc-rule. This result can be found in Section 4 of Nain [1] and in Section 2 of Nain et al. [2]. However, it is not true. In this note we give a counterexample which contradicts it.

In order to keep our example simple we assume that there are no arrivals and there is no discounting. However, similar examples can be constructed with arrivals and discounting. In our example there are three customers present at time 0. There are two customer classes, and customers of type i will join queue i, $i = 1, 2$, in both nodes. The parameters μ_{ij}, c_{ij}, $i, j = 1, 2$, are given in Figure 1 of Section 2. In order to keep the computation simple we take the μ_{12} equal to zero. However, it is easy to construct counterexamples for which all μ's are positive and such that the holding cost rates in node 2 are strictly smaller than in node 1.

Let us summarize here the conclusions which follow from the computations of Section 2. The optimal decision rule in node 2 serves queue 2 first, whereas, the μc-rule gives priority to the customer in queue 1. Note that the optimal decision rule in node 1 depends on the numbers of customers in the queues of node 2. Indeed, the job in the first queue starts its

Received 5 July 1991; revision received 9 September 1991.

* Postal address for both authors: Department of Mathematics and Computer Science, University of Leiden, P.O. Box 9512, 2300 RA Leiden, The Netherlands.

service at the instant that the job in the second queue of the second node finishes its service. Until that instant the server in node 1 serves queue 2, i.e. the server remains idle.

The proof in Section 2 of Nain et al. [2] and Section 4 of Nain [1] of the optimality of the μc-rule in the second node uses an interchange argument for the order of service in the second node. They compare the optimal policy, say R^*, with a policy, say R_0, which is constructed such that the service process in the first node generates the same arrival process for node 2 as policy R^*. In the second node the R_0 policy follows the μc-rule. This means in our example that service in node 2 is started with the job in queue 1. In order to know when the service to the job in queue 1 of node 1 has to be started, a random variable is generated at time 0 with the same distribution as the service time of the job in queue 2 of node 2.

From the cited papers it is not clear whether the arrival process in node 2 is constructed to be dependent or independent of the service process in this node. The relevant passages are [1], p. 183, line 4 of the proof of Theorem 4.1: 'node 1 has the same behavior under both policies', and [2], p. 818, line 4: 'in the above proof, policies π and $\tilde{\pi}$ result in identical arrivals for node 2'. Therefore we consider two cases. In case 1 the idle time of the server in node 1 is equal to the service time of the job in queue 2 of node 2. In case 2 this idle time is equal in distribution but independent of this service time. Before we proceed to this case, we remark that the definition of $\tilde{\pi}$ on p. 817 of [2] generates in our example a definitely different arrival process in node 2.

Case 1. Assume that the generated service time is equal to the required service time of the job in queue 2 of node 2. In this case the decision rule of R_0 in node 1 is anticipative, it uses the information of the length of the service time of the job in queue 2 of node 2 before the actual service of this job is finished. Using the interchange argument as in the cited literature shows that R_0 has a smaller total expected holding cost. However, R_0 is not a feasible policy.

Case 2. Assume that the required service time is independent of the generated copy. In this case the server of node 1 stays idle for an exponential period with parameter 1 and then starts serving the job in queue 1. In node 2 the μc-rule is followed. Let us call this policy R_1 to distinguish it from R_0. It is a policy which uses only the information of the service process in the first node. Therefore it is likely less good than the optimal policy which uses the information of service process in both nodes. In Section 2 we compute the total holding cost for the R_1 policy, and we find that it is not optimal.

2. Counterexample

There are three customers present, one in the first queue of node 1 and one in each of the queues of node 2. The parameters of the exponential distributions and the holding cost rates are given in Figure 1.

The optimal policy. Denote by K_{ijk} the total expected holding cost when at time 0 there are i customers in the first queue of centre 1, j customers in the first queue of centre 2, and k customers in the second queue of node 2. It follows from the optimality of the μc-rule in a one-node network that the optimal policy in centre 2 is the μc-rule when centre 1 is empty. Clearly, idleness in centre 2 is suboptimal. Hence the total expected holding cost for the

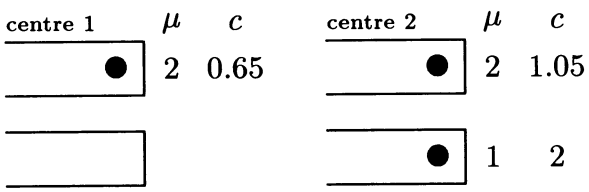

Figure 1

https://doi.org/10.2307/1427742 Published online by Cambridge University Press

optimal policy in starting states with empty first node are:

$$K_{001} = \frac{2}{1} = 2;$$

$$K_{010} = \frac{1{\cdot}05}{2} = 0{\cdot}525;$$

$$K_{011} = \frac{3{\cdot}05}{2} + K_{001} = 3{\cdot}525;$$

$$K_{020} = \frac{2{\cdot}1}{2} + K_{010} = 1{\cdot}575;$$

$$K_{021} = \frac{4{\cdot}1}{2} + K_{011} = 5{\cdot}575;$$

$$K_{100} = \frac{0{\cdot}65}{2} + K_{010} = 0{\cdot}85.$$

In starting states with customers in both centres the total expected holding cost is the minimum of terms corresponding to different actions. Denote by (i, j), $i = 1, 2$, $j = 1, 2$ the actions to serve queue i in node 1 and queue j in node 2. The successive terms in the computation below correspond to the action pairs $(1, 1)$, $(1, 2)$, $(2, 1)$, $(2, 2)$ respectively, where terms belonging to actions corresponding to idleness in centre 2 are deleted. The optimal action pair in starting state ijk is denoted by a_{ijk}.

$$K_{101} = \min\left\{\frac{2{\cdot}65}{3} + \frac{1}{3}K_{100} + \frac{2}{3}K_{011}; \frac{2{\cdot}65}{1} + K_{100}\right\}$$

$$= \min\{3{\cdot}517; 3{\cdot}5\} = 3{\cdot}5;$$

$$a_{101} = (2, 2);$$

$$K_{110} = \min\left\{\frac{1{\cdot}7}{4} + \frac{1}{2}K_{020} + \frac{1}{2}K_{100}; \frac{1{\cdot}7}{2} + K_{100}\right\}$$

$$= \min\{1{\cdot}638; 1{\cdot}7\} = 1{\cdot}638;$$

$$a_{110} = (1, 1);$$

$$K_{111} = \min\left\{\frac{3{\cdot}7}{4} + \frac{1}{2}K_{021} + \frac{1}{2}K_{101}; \frac{3{\cdot}7}{3} + \frac{1}{3}K_{110} + \frac{2}{3}K_{021}; \frac{3{\cdot}7}{2} + K_{101}; \frac{3{\cdot}7}{1} + K_{110}\right\}$$

$$= \min\{5{\cdot}463; 5{\cdot}496; 5{\cdot}35; 5{\cdot}338\} = 5{\cdot}338;$$

$$a_{111} = (2, 2).$$

From $a_{110} = (1, 1)$, $a_{101} = (2, 2)$ and $a_{111} = (2, 2)$ we conclude that the server in node 1 will start serving the job in queue 2 of node 2 has finished its service. Hence the optimal action in node 1 depends on the state in node 2. Since $a_{111} = (2, 2)$, the server in node 2 will serve the job in queue 2 before the job in queue 1. Note that the optimal action in node 2 depends on the state in node 1.

The R_1-policy. The R_1 uses the μc-rule in node 2. In node 1 the server will remain idle for an exponential time with mean 1. Thereafter the job in queue 1 will be served. Let K_{111} (K^*_{111}) denote the total expected holding cost when the server in node 1 is idle (serves queue 1). Since the policy R_1 is fixed there is no minimization in the computation. The total expected holding cost for starting states 001, 010, 011, 020 and 021 are equal to those of the

https://doi.org/10.2307/1427742 Published online by Cambridge University Press

optimal policy. The computation of the other values is as follows:

$$K^*_{100} = \frac{0{\cdot}65}{2} + K^*_{010} = 0{\cdot}85;$$

$$K_{100} = \frac{0{\cdot}65}{1} + K^*_{100} = 1{\cdot}5;$$

$$K^*_{101} = \frac{2{\cdot}65}{3} + \frac{2}{3}K^*_{011} + \frac{1}{3}K^*_{100} = 3{\cdot}517;$$

$$K_{101} = \frac{2{\cdot}65}{2} + \frac{1}{2}K^*_{101} + \frac{1}{2}K_{100} = 3{\cdot}834;$$

$$K^*_{111} = \frac{3{\cdot}7}{4} + \frac{1}{2}K^*_{021} + \frac{1}{2}K^*_{101} = 5{\cdot}471;$$

$$K_{111} = \frac{3{\cdot}7}{3} + \frac{1}{3}K^*_{111} + \frac{2}{3}K_{101} = 5{\cdot}613.$$

We conclude that K_{111}, i.e. the total expected holding cost in starting state 111 is smaller for R_0 than for R_1. Hence R_1 is not optimal.

References

[1] NAIN, P. (1989) Interchange arguments for classical scheduling problems in queues. *Syst. Control Lett.* **12,** 177–184.

[2] NAIN, P., TSOUCAS, P. AND WALRAND, J. (1989) Interchange arguments in stochastic scheduling. *J. Appl. Prob.* **27,** 815–826.

https://doi.org/10.2307/1427742 Published online by Cambridge University Press