Analysis of a customer assignment model with no state information

A. Hordijk
University of Leiden
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

G.M. Koole*
CWI
P.O. Box 94079
1090 GB Amsterdam
The Netherlands

J.A. Loeve**
University of Leiden
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

In this paper we analyse a queueing network consisting of parallel queues and arriving customers which have to be assigned to one of the queues. The assignment rule may not depend on the numbers of customers in the queues. Our goal is to find a policy which is optimal with respect to the long run average cost. We will consider two cases, holding costs and waiting times. A recently developed algorithm for Markov decision chains with partial state information is applied. It turns out that the periodic policies found by this algorithm are close, if not equal, to the optimal ones.

Keywords: Queueing system, customer assignment, Markov decision processes, partial information, periodic policies.

Published in: Probability in the Engineering and Informational Sciences **8**:419–429, 1994.

* The research of this author has been supported by the European Grant BRA-QMIPS of CEC DG XIII.

** The research of this author has been supported by the Netherlands Organization for Scientific Research (N.W.O.).

1. Introduction

In this paper we analyse a queueing system with customer assignment. The system has a finite capacity and consists of parallel queues with exponential servers. The service rates may be different. Customers arrive according to a Poisson process and have to be assigned to one of the queues without knowing the numbers of customers in the queues. We will present an algorithm to find good periodic policies for assigning the customers.

There is an extensive literature on the subject of customer assignment. In the symmetric case, when all the servers have the same rate, many results are known for infinite as well as finite capacity queues, and for different kinds of information (for some recent results, see [15], [6], [12], [7], [9], [11], [1], [14], [13]).
These results show that in many variations of the model where the queue lengths are known the shortest queue policy, which sends the customers to the shortest queue, or a generalization of this policy is optimal. In various other cases, where the only information is on past routing decisions, the round robin policy is proven to be optimal. The round robin policy sends arriving customers cyclically to the servers. In the case where the service time rates are not identical it is more difficult to find the optimal policy. Partial results on the structure of the optimal policy for the asymmetric model can be found in [4] and [7].

For the model in this paper, which is also asymmetric, we give numerical results produced by an algorithm which is generally applicable to models with partial state information.
Combé and Boxma ([2]) have also developed an algorithm that can be used to find good periodic policies for this model. We will compare the policies found by both methods.

The outline of the paper is as follows. In section 2 we introduce the model with two different cost functions. The algorithm can be found in section 3, it is a special case of an algorithm for Markov decision chains with partial state information, introduced in Hordijk and Loeve [8]. Section 4 gives numerical results and a comparison between our results and the results found in Combé and Boxma [2].

2. Queueing model

Consider m parallel queues, where each queue has its own server. Customers arrive according to a Poisson process with parameter λ and the service times are exponentially distributed with parameter μ_i for the server of queue i, $i = 1, \ldots, m$. We denote the state of the system as the m-tuple $(i_1, \ldots, i_m)$, with i_j the number of customers in the j-th queue, including the customer in service.

The arriving customers have to be assigned to one of the queues, without having information available about the number of customers in the system. Thus the assignment is independent of the state of the system. Furthermore, we assume the system has a finite capacity, and hence we have a finite state space. Let N_1 denote the maximum number of customers in the system. Customers which arrive when the system is full are rejected. We will see the finite system as an approximation of the infinite system. More precisely, for the policy found by the algorithm in the finite system we computed the average cost in the model with unbounded capacity.

We will analyse two different cost functions. In the first case we assume a holding cost in queue i equal to c_i per time unit. Thus the cost per time unit in state $(i_1, \ldots, i_m)$ is $c_1 i_1 + \cdots \ldots + c_m i_m$.

In the second case each arriving customer adds a cost to the system which is equal to its expected waiting time.

A policy which minimizes the long run average cost for a given initial state will generally have a very complicated structure, when its decisions depend on the past states and actions. Moreover, there are no efficient algorithms available for computing an optimal history-dependent policy.

Indeed, in the literature on partially observed Markov decision processes we have not found an algorithm which can solve problems of the size we consider in this paper. Therefore, we restrict the class of admissible policies to Markov policies and our goal in this paper is to find good policies in the class of periodic policies. Of course, also from point of view of the implementation of a given policy in practice, the periodic polices are far more useful than history-dependent ones.

3. The algorithm

In this section we will give an algorithm to compute good periodic policies for Markov decision models with no state information. This algorithm is a special case of the more general algorithm described in [8].

Consider a Markov decision chain with state space $E = \{1, 2, \ldots, N\}$. There is no information available about the state of the system at any decision moment. Thus an admissible decision rule prescribes the same decision in each state. Note that if the decision rule is deterministic it is completely determined by the action it prescribes.
We call a Markov policy admissible if its decision rule at any time point is admissible. An admissible deterministic Markov policy is denoted by $R = (f_1, f_2, \ldots)$, where f_n is the action chosen at time n. R is called periodic if $f_n = f_{n+L}$ for fixed L and all n and then it is denoted by $(f_1, \ldots, f_L)^\infty$.
We will restrict ourselves to the class of periodic deterministic admissible policies.

In the general model with partial state information the state space is partitioned into sets E_s, $s = 1, \ldots, K$, such that the only information available about the system is the set of the partition in which the state of the system is contained (cf. [10], [8]).

We assume that the action set in each state is the same, namely $\{1, 2, \ldots, m\}$. Here action i means that an arriving customer is assigned to queue i, $i = 1, \ldots, m$. The transition probabilities when in state i action a is chosen are denoted by p_{iaj}, $j = 1, \ldots, N$ and the expected one step cost by c_{ia}. For the transition matrix when decision rule f is chosen we will write $P(f)$.

The algorithm we used in our computations is as follows.

Algorithm.

Choose an initial admissible deterministic decision rule f^1 *and* $\epsilon > 0$.

Choose x^1 *and* $v^1 \in \mathbb{R}^N$.

Define for $n = 1, 2, \ldots$:

$$g^n(a) = \sum_{i \in E} x_i^n \left\{ c_{ia} + \sum_{j=1}^{N} p_{iaj} v_j^n \right\},\ a = 1, \ldots, m.$$

If $f^n \in \operatorname{argmin}_a\ g^n(a)$ *then* $f^{n+1} = f^n$,
else $f^{n+1} = a$ *for some* $a \in \operatorname{argmin}_a\ g^n(a)$;

$$x^{n+1} = x^n P\left(f^{n+1}\right);$$

$$v^{n+1} = c\left(f^{n+1}\right) + P\left(f^{n+1}\right) v^n.$$

Stop if there is an L *(as small as possible) such that* $|x_i^{n+L} - x_i^n| < \epsilon$ *for* $i = 1, \ldots, N$ *and* $\operatorname{span}(v^{n+L} - v^n) < \epsilon$.

Note that if $L = 1$ the algorithm finds a stationary deterministic policy and if $L > 1$ the algorithm gives a periodic deterministic policy.
Under certain conditions the policies found by the algorithm are locally optimal (or saddle points) in the class of all admissible periodic policies (for more details, see [8]).

4. Numerical results

In this section we give the numerical results found by the algorithm of the previous section, applied to the queueing model of section 2. We used a semi-Markov model with decision moments the arrival moments of the customers. In the semi-Markov model the actions in a period, say of length L, are the assignments of L successively arriving customers.
For ϵ we used the value of 10^{-10}.

In the model with holding cost, c_{ia} is the total expected holding cost between two arrivals. Thus,

$$c_{(i_1,\ldots,i_m)a} = \mathbb{E} \int_0^\infty \lambda \mathrm{e}^{-\lambda \mathrm{t}} \sum_{\mathrm{k}=1}^{\mathrm{m}} \sum_{\mathrm{l}=1}^{\mathrm{i_k}+\delta(\mathrm{a}=\mathrm{k})} \mathrm{c_k} \min\{\tau_{(\mathrm{k,l})}, \mathrm{t}\}\ \mathrm{dt},$$

where $\tau_{(k,l)}$ is the departure time of the l-th customer in queue k, which has an Erlang-l distribution with parameter μ_k.

For the model with waiting cost, c_{ia} is the expected waiting time of an arriving customer,

$$c_{(i_1,\ldots,i_m)a} = \frac{i_a + 1}{\mu_a}.$$

The transition probabilities $p_{(i_1,\ldots,i_m)a(j_1,\ldots,j_m)}$ are 0 if $j_k > i_k + \delta(a = k)$ and otherwise

$$p_{(i_1,\ldots,i_m)a(j_1,\ldots,j_m)} = \int_0^\infty \lambda e^{-\lambda s} \prod_{k=1}^{m} \frac{(\mu_k s)^{i_k - j_k + \delta(a=k)}}{(i_k - j_k + \delta(a = k))!} e^{-\mu_k s}\, ds,$$

if $j_k > 0$ for all k.
If $j_k = 0$ for some k then the expression becomes somewhat more complex, for example if $j_1 = 0$ and $j_k > 0$ for $k = 2, \ldots, m$ we have

$$p_{(i_1,\ldots,i_m)a(j_1,\ldots,j_m)} = \int_0^\infty \lambda e^{-\lambda s} \sum_{l=i_1+\delta(a=1)}^{\infty} \frac{(\mu_1 s)^l}{l!} e^{-\mu_1 s} \prod_{k=2}^{m} \frac{(\mu_k s)^{i_k - j_k + \delta(a=k)}}{(i_k - j_k + \delta(a = k))!} e^{-\mu_k s}\, ds.$$

For other cases we get analogous expressions.

The rather long computation times were a serious restriction. Therefore we did not apply the algorithm to queueing models with large buffer size N_1. A typical value of N_1 was 25 when m was 2 and 20 when m was equal to 3. However, for the policies found by the algorithm we computed the long run average cost for the model with unlimited queue sizes. In order to do this we computed the costs in the different queues separately and we summed these costs. For the computation of the costs in each queue we enlarged the buffer until further enlargement did not alter the cost, thus until the long run average cost of the policies is independent of the buffer size. Hence, in this sense the cost in the tables is the cost in the system with infinite capacity when the given policies are followed. However, the policies itself are found in the system with finite capacity.

The buffer sizes needed per queue to achieve that the long run average cost was independent of the buffer size, were different and increased when the traffic intensity increased. For example, for the first entries in Table 4 we needed a buffer size of 25 and for the last entries a buffer size of 185.

4.1. Queueing networks with holding cost

The results in Table 1 are found in the model with holding cost and two queues ($m = 2$). 'Fraction action 1' denotes the total number of decisions in the period which are equal to action 1 divided by the length L of the period. The policy corresponding to a given fraction is found to be the most symmetric distribution of the actions 1 and 2 over the period. For example, if we have a period of 9 and the fraction of action 1 is $\frac{2}{9}$, then the actions in the period will be 222212221. Thus the policy is uniquely determined by the period and the fraction of actions in the period equal to action 1 (see section 4.2).

The first three problem instances of Table 1 show that all customers are assigned to the fastest server when the traffic intensity is low. However, if the traffic intensity increases, the fraction of customers assigned to the slowest server increases also. Hence our algorithm finds that for higher traffic it becomes better to assign also customers to the slow server which likely will have a shorter waiting queue than assigning all customers to the fast server where the waiting queue can be long.

Basically, the same thing happens in the second part of the table. Here the rates of the service time distributions are the same for both servers but in the second queue the holding cost is higher. In this case the fraction of customers assigned to the most costly queue becomes larger when the traffic intensity increases.

However, if we consider a completely symmetric system the algorithm finds for all traffic intensities the round robin policy, which alternately assigns customers to the first and the second queue (this policy is optimal, see [15]).
Note that when this policy is followed, each queue can be seen as an Erlang arrival model $E_2/M/1$. In this case there is an explicit formula for the cost (see [3]: section 4.3)

$$\text{Long run average cost in one queue } = \frac{\lambda}{4 + 4\sqrt{\lambda + 1} - 2\lambda}.$$

Using this formula for $\lambda \in \{1, 2, 3\}$ in Table 1, gives the same values as those found by using the algorithm.

Note that if the traffic intensity is close to 1, the assignment of the arriving customers becomes independent of the holding cost in the queues. For example, if $\mu_1 = 4$, $\mu_2 = 1$ and λ close to 5, the best policy will be to assign 4 out of 5 arriving customers to the first queue and the remaining one of the 5 arrivals to the second queue. In this case it does not matter that the holding costs may be different.

λ	μ_1	μ_2	c_1	c_2	Period	Fraction action 1	Cost per time unit
0.5	4	1	1	1	1	1	0.142857
1	4	1	1	1	1	1	0.333333
2	4	1	1	1	1	1	1.000000
3	4	1	1	1	6	$\frac{5}{6}$	2.263505
3.5	4	1	1	1	29	$\frac{24}{29}$	3.460522
4	4	1	1	1	138	$\frac{113}{138}$	5.849738
0.5	3	3	1	2	1	1	0.200000
1	3	3	1	2	4	$\frac{3}{4}$	0.487735
2	3	3	1	2	5	$\frac{3}{5}$	1.200628
3	3	3	1	2	7	$\frac{4}{7}$	2.329025
4	3	3	1	2	9	$\frac{5}{9}$	4.539892
1	4	4	1	1	2	$\frac{1}{2}$	0.261204
2	4	4	1	1	2	$\frac{1}{2}$	0.577350
3	4	4	1	1	2	$\frac{1}{2}$	1.000000

Table 1. Two queues with holding costs

For the model with three queues ($m = 3$) the results are presented in Table 2. In this case the policies also have very symmetric distributions of the different actions in the period, but they are not uniquely determined by the period and the fractions of decisions in the period equal to action 1 and action 2. However, the policies are such that they can not be made more symmetric. For example, look at the one but last entry in Table 2. We find a policy with a period of 25 where the

fraction of action 1 is $\frac{1}{25}$ and the fraction of action 2 is $\frac{8}{25}$. The policy found by the algorithm is 123323323323233233233233, but we can also construct the intuitively not less 'symmetric' policies 133233233233233233233233 and 132332332332332332332323.
The first two policies have the same long run average cost, while the cost of the last one is somewhat less. That the cost is not the same for all three policies can be explained by the fact that the distributions of the actions in the period is not the same. Actions 1 and 2 have the same distributions in each policy, but action 3 has the same distribution in the first two policies and a different one in the last policy.

In Table 2 we see again that when the traffic intensity is low, the customers are served by the fastest server. When the traffic intensity increases, more customers are served by the second fastest server. When the arrival rate is still higher, a fraction of the customers will also be assigned to the slowest server.

λ	μ_1	μ_2	μ_3	c_1	c_2	c_3	Period	Fraction action 1	Fraction action 2	Cost per time unit
1	3	3	3	1	1	1	3	$\frac{1}{3}$	$\frac{1}{3}$	0.338825
1	3	3	3	2	1	1	2	0	$\frac{1}{2}$	0.358258
1	1	4	7	1	1	1	1	0	0	0.166667
2	1	4	7	1	1	1	4	0	$\frac{1}{4}$	0.390380
3	1	4	7	1	1	1	3	0	$\frac{1}{3}$	0.645986
4	1	4	7	1	1	1	3	0	$\frac{1}{3}$	0.955744
5	1	4	7	1	1	1	3	0	$\frac{1}{3}$	1.359657
6	1	4	7	1	1	1	3	0	$\frac{1}{3}$	1.917833
7	1	4	7	1	1	1	3	0	$\frac{1}{3}$	2.752564
8	1	4	7	1	1	1	25	$\frac{1}{25}$	$\frac{8}{25}$	3.934037
9	1	4	7	1	1	1	101	$\frac{5}{101}$	$\frac{32}{101}$	5.964467

Table 2. Three queues with holding costs

4.2. On the symmetry of the computed policies

It is shown in Hajek [5], that when a fraction p of Poisson arrivals has to be assigned to an exponential server, then the long run average queue size is minimized when the following zero-one sequence is used:

$$\lfloor p \rfloor, \lfloor 2p \rfloor - \lfloor p \rfloor, \lfloor 3p \rfloor - \lfloor 2p \rfloor, \lfloor 4p \rfloor - \lfloor 3p \rfloor, \ldots .$$

where a 0 denotes that a customer is blocked and a 1 denotes that a customer is admitted to the queue.

If we apply our algorithm to the model with $m = 2$, we find policies with a very nice property. Namely, if the policy is periodic the different actions in the period are distributed in a most regular way. For example, we find a periodic policy with period 29 in the model with holding cost for the

following parameters $\lambda = 3.5$, $\mu_1 = 4$, $\mu_2 = 1$, $c_1 = c_2 = 1$ and $\frac{24}{29}$ as fraction of action 1. Then the subsequent actions in 1 period are 11111211111211111211111211112, thus we have 4 blocks consisting of 5 times action 1 and once action 2 followed by 4 times action 1 and once action 2.

In each case where we found a periodic policy we computed the fraction p of the decisions that choose a fixed action a. Then our algorithm found that the actions a in the period are distributed like the ones in the zero-one sequence above. Therefore, we know that in our results the algorithm finds the best possible policy for assigning the customers, given the fraction of arriving customers to be assigned to the first queue.

In some cases we found that the algorithm does not find the fraction with minimal expected cost. The reason is probably that the algorithm is applied to the queueing model with finite capacity, whereas the average cost is computed for the model with unrestricted queue sizes.

We have an example of a case where the algorithm does not find the right fraction in the following table. It concerns the model with holding cost and the following parameters $m = 2$, $\lambda = 3.5$, $\mu_1 = 4$, $\mu_2 = 1$, $c_1 = c_2 = 1$ where the policies are found in the model with buffer size 25.

Fraction action 1	Cost per time unit
$\frac{24}{29}$	3.460522
$\frac{23}{29}$	3.794361
$\frac{25}{29}$	3.548548
$\frac{119}{145}$	3.484499
$\frac{121}{145}$	3.451432

Table 3. Long run average cost for different fractions

In this table we see that fraction $\frac{24}{29}$ is the best fraction when the period is 29, but if we increase the period length to $5 * 29 = 145$ we find smaller costs for other fractions. Thus the policy found by the algorithm may be suboptimal, but our experience is that it is a good policy.

4.3. Queueing model with expected waiting times

Now we want to look at the results for the model where the cost for each customer is equal to his expected waiting time. Table 4 gives the results for $m = 2$ and Table 5 for $m = 3$.

λ	μ_1	μ_2	Period	Ratio action $1:2$	$\mathbb{E}$W per customer
0.5	1	4	4	1 : 3	0.019971
1	1	4	4	1 : 3	0.056244
1.25	1	4	5	1 : 4	0.072906
2	1	4	5	1 : 4	0.162552
2.5	1	4	11	2 : 9	0.247740
3	1	4	11	2 : 9	0.383852
3.5	1	4	11	2 : 9	0.612793
3.75	1	4	11	2 : 9	0.797160
4	1	4	49	9 : 40	1.077483
4.5	1	4	43	8 : 35	2.521454

Table 4. Two queues with expected waiting times

λ	μ_1	μ_2	μ_3	Period	Ratio action $1:2:3$	$\mathbb{E}$W per customer
3	1	4	7	12	1 : 4 : 7	0.030911
6	1	4	7	87	6 : 28 : 53	0.119546
9	1	4	7	58	4 : 19 : 35	0.411938

Table 5. Three queues with expected waiting times

In these two tables we see customers served by the slowest server even when the traffic intensity is low. When the traffic intensity increases the fraction of customers served by the slowest server(s) first decreases and then increases. This is a consequence of the fact that in this case only the expected waiting time of a customer is counted. Hence, if a queue is empty the additional cost of an arriving customer is zero. When the traffic intensity is very low, some customers are assigned to the slowest server but not enough to cause a waiting queue. These customers do not increase the total cost. However, when the traffic intensity increases, less customers are assigned to the slowest queue to avoid a waiting queue.
The increasing fraction when the traffic intensity increases further can be explained by the same arguments as in section 4.1.

4.4. Comparison with the results of Combé and Boxma

For the model with expected waiting times there already exists a method to find a good periodic policy. This method was developed in Combé and Boxma [2]. The next table gives some of their results which can be compared to the corresponding entries in Table 4 and Table 5. The costs in Table 6 are computed from their policies in the way described in the first paragraph of section 4.

m	λ	μ_1	μ_2	μ_3	Period	Ratio action $1 : 2 (: 3)$	IEW per customer
2	1.25	1	4	-	46	9 : 37	0.072952
2	2.5	1	4	-	59	10 : 49	0.245182
2	3.75	1	4	-	45	8 : 37	0.795461
3	3	1	4	7	63	5 : 21 : 37	0.031997
3	6	1	4	7	95	6 : 31 : 58	0.120667
3	9	1	4	7	89	6 : 29 : 54	0.414366

Table 6. Results Combé and Boxma

These results show that for the given parameters and $m = 3$, our algorithm finds a policy with less expected waiting costs than the Combé and Boxma algorithm, while in some cases for $m = 2$, it is the other way around. However, the difference in the long run average cost of the policies found by the two methods is small. In all cases the algorithm of section 3 found a policy with a smaller period than the method proposed by Combé and Boxma.

Acknowledgement

The authors would like to thank M.B. Combé and O.J. Boxma for the use of their results.

References

[1] Chang, C.S. (1992). A new ordering for stochastic majorization: Theory and applications. *Advances in Applied Probability* 24: 604-634.

[2] Combé, M.B., Boxma, O.J. (1993). Optimization of static traffic allocation policies. Technical Report BS-R9302, CWI, Amsterdam. To appear in *Theoretical Computer Science A* 125 (1994).

[3] Gross, D., Harris, C.M. (1974). *Fundamentals of Queueing Theory.* New York: John Wiley & Sons, Inc.

[4] Hajek, B. (1984). Optimal control of two interacting service stations. *IEEE Transactions on Automatic Control* 29: 491-499.

[5] Hajek, B. (1985). Extremal splitting of point processes. *Mathematics of Operations Research* 10: 543-556.

[6] Hordijk, A., Koole, G. (1990). On the optimality of the generalized shortest queue policy. *Probability in the Engineering and Informational Sciences* 4: 477-487.

[7] Hordijk, A., Koole, G. (1992). On the assignment of customers to parallel queues. *Probability in the Engineering and Informational Sciences* 6: 495-511.

[8] Hordijk, A., Loeve, J.A. (1993). Undiscounted Markov decision chains with partial information; an algorithm for computing a locally optimal periodic policy. Technical report TW-93-10, University of Leiden. To appear in *ZOR-Mathematical Methods of Operations Research* 40, issue 3 (1994).

[9] Koole, G. (1992). On the optimality of FCFS for networks of multi-server queues. Technical Report BS-R9235, CWI, Amsterdam.

[10] Kulkarni, V.G., Serin, Y. (1990). Optimal implementable policies : average cost and minimax criteria. Technical report no. UNC/OR TR-90/9, Department of Operations Research, University of North Carolina, Chapel Hill.

[11] Liu, Z., Towsley, D. (1992). Optimality of the round robin routing policy. COINS Technical report 92-55, University of Massachusetts at Amherst.

[12] Menich, R., Serfozo, R.F. (1991). Monotonicity and optimality of symmetric parallel processing systems. *Queueing Systems, Theory and Applications* 9: 403-418.

[13] Towsley, D., Sparaggis, P.D. (1993). Optimal routing in systems with ILR service time distributions. CMPSCI Technical report 93-13, University of Massachusetts at Amherst.

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

[15] Walrand, J. (1988). *An Introduction to Queueing networks.* Englewood Cliffs: Prentice-Hall.