J. Indian Inst. Sci., 66, Nov.–Dec. 1986, pp. 655–667

Dynamic arrival routing – a generalized problem and an optimal routing algorithm

V.S. LAKSHMANAN* AND M.A.L. THATHACHAR†
Indian Institute of Science, Bangalore 560 012, India.

Received on June 17, 1986.

Abstract

A generalized problem in dynamically routing the arrivals in a parallel queueing network is proposed and an optimal routing algorithm developed. Customers arrive in an arbitrary stream into a queueing network composed of n parallel G | M | 1 queues. The individual mean service times of the n servers are distinct; the individual queue lengths are measurable. It is desired by appropriately regulating routing of arrivals, to minimize the running cost of the system, *viz.*, the expected total time for the completion of service on all customers arriving within a specific time interval $[O, T]$. Optimality of a proposed algorithm is proved and is further substantiated by simulation results.

Key words: Queueing network, dynamic routing policy, routing algorithm, recursion.

1. Introduction

Routing problems arising in queueing networks naturally fall into two broad categories: static and dynamic. In static routing, the strategies are stationary in the sense that they are not responsive to changes in system parameters. Such strategies were studied by Fretta *et al*[1] and Gallager[2]. On the other hand, the study of dynamic routing has been relatively less extensive. A dynamic strategy bases its choice for the route of an arrival on the information *currently* available. In this paper we propose a generalized dynamic routing problem in the context of n parallel queues and develop an appropriate routing algorithm whose optimality we subsequently establish. The organization of the paper is as follows. The remainder of this section describes the background and formulates the problem. In Section II, the routing algorithm is developed and physically interpreted using a diagram. In Section III, the algorithm is formally proved to be optimal. Section IV discusses the results of simulation of the problem and draws useful inferences. Finally, Section V concludes with a brief summary of the ideas and results provided in this paper.

A service center employs a queueing network consisting of n parallel G | M | 1 queues, where the individual mean service rates $m_i (i = 1, 2, \ldots\ldots, n)$ are known *a priori* and are in general distinct (fig. 1). Incoming arrivals are to be routed among the n parallel queues. It is desired to develop a routing algorithm that exhibits optimal behaviour w.r.t. the running cost associated with the system – the expected total time for the completion

* School of Automation; † Department of Electrical Engineering.

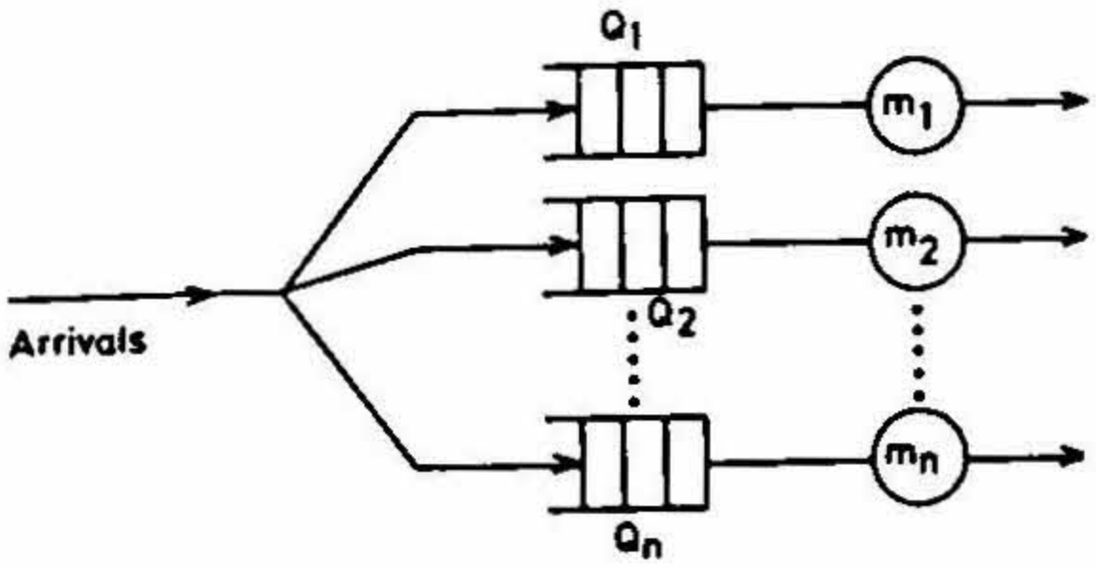

FIG. 1. A queueing system composed of n G | M | 1 queues among which arrivals are to be routed.

of service on all customers arriving within a specific time interval $[O, T]$. Ephremides *et al*[3] consider a similar routing problem and develop an optimal routing algorithm. Their study is confined to the routing problem in a queueing network consisting of two parallel queues, each served by an identical exponential server. Winston[4] deals with a somewhat more general problem of n identical exponential servers. In both the cases the authors have established the optimality of what they term the Send-to-the-Shortest queue (SS) policy, w.r.t. the cost defined above. In this paper we shall be concerned with a very generalized situation in dynamic routing where n distinct parallel exponential servers comprise the service facility. We see that a special routing algorithm that we propose is optimal, and that SS policy ceases to be optimal. Our conclusions are supported by theoretical proofs as well as results of computer simulation.

There are other papers which deal with related queueing problems although they use models quite different from ours. For instance, Agrawala *et al*[5] study the problem of scheduling a given set of jobs among n processors while Yum[6] analyses the performance of deterministic routing sequences which maximize traffic bifurcation in a queueing network. Lin and Kumar[7] consider routing customers from a single queue to two heterogeneous servers. Rosberg and Towsley[8] consider arrival routing to n servers with unequal service rates but without any queuing facility in the system.

Applications of the routing problem considered here are manifest in the context of routing individual packets among several links in a computer network employing store-and-forward and packet switching technologies. Another interesting area is that of transfer lines[9] where routing semi-finished products among alternative channels is a rather important problem.

2. Development of the optimal routing algorithm

We may assume without loss of generality that the service rates m_i $(i = 1, 2, \ldots, n)$ are arranged such that

$$m_1 \geq m_2 \geq \ldots \geq m_n. \tag{1}$$

The cost function defined in the previous section, namely, the expected total time for the completion of service of all customers arriving in $[O, T]$ is seen[3] to be given by

$$K(T) = E\left[\int_0^T \left(\sum_{i=1}^{n} x_t^i \right) dt + \frac{1}{2} \sum_{i=1}^{n} \frac{1}{m_i} x_T^i (x_T^i + 1) \right] \tag{2}$$

where x_t^i denotes the length of the i-th queue, $(i = 1, 2, \ldots, n)$ at time t and E stands for the expectation w.r.t. the random variables x_t^i. We will need some definitions to aid in our exposition of the algorithm development and the proof of optimality. Let

$$x_t = (x_t^1, x_t^2, \ldots, x_t^n), \tag{3}$$

$$a_i = \frac{1}{2m_i}, \ (i = 1, 2, \ldots, n), \tag{4}$$

and

$$C(x^i, x^j) = \left(\frac{x^i + 1}{x^j + 1} \leq \frac{a_j}{a_i}\right). \tag{5}$$

It should be noted that $C(x^i, x^j)$ is really a dyadic predicate which has been assigned a binary relation over the domain

$$D = \{(x^i, x^j)/x^i \text{ and } x^j \text{ are non-negative integers}\} = N^2. \tag{6}$$

Also, for a given sum $\Sigma_{i=1}^n x^i = k$, let $K_k(x^1, x^2, \ldots, x^n)$ represent the value of $K(T)$ computed per (2), assuming that $x_0 = (x^1, x^2, \ldots, x^n)$ and that a fixed routing policy is adopted throughout. Clearly K_k thus computed would depend for its value on the initial load configuration $(x^1, x^2, \ldots, x^n) = x_0$. In the following, we shall first state the optimal routing algorithm and then undertake an interpretive discussion in order to hopefully develop an intuitive feel for the algorithm.

Algorithm 1

At any arrival epoch $t \in [O, T]$, the optimal routing choice for the arrival is computed as follows:

$$j = arg\ [\min_i a_i(x_t^i + 1)] \tag{7}$$

where *arg* is an operator which yields the value of the argument i over $\{1, 2, \ldots, n\}$, satisfying the intended operation (which in this case is minimization) on the given function. Routing performed using the above computation is guaranteed to behave optimally w.r.t. the cost function $K(T)$.

For clarity and ease of handling we shall consider the simple case $n = 2$ of the dynamic routing problem in the following discussion. Dropping subscripts denoting the arrival epoch, we readily see that

$$C(x^1, x^2) \Longrightarrow (x^1 \leq x^2)$$

or

$$((x^1 > x^2) \quad and \quad (a_1(x^1 + 1) \leq a_1(x^2 + 1))) \tag{8}$$

where $\Longrightarrow$ denotes logical implication and the predicate C is as defined by Buzacott and Hanifin[9]. This idea is clearly brought out in fig. 2 which portrays plots of $K_k(x^1, x^2)$ for a

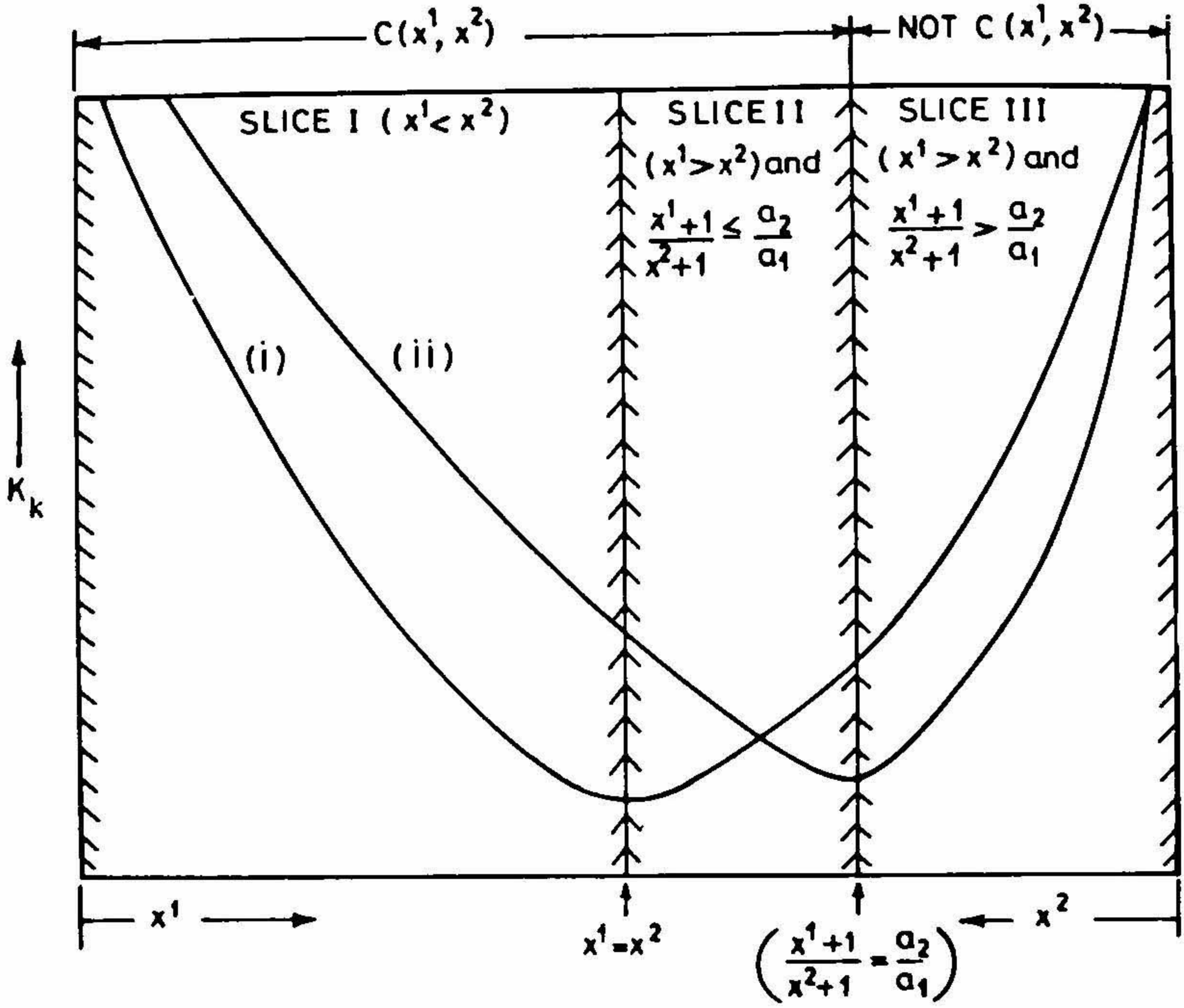

FIG. 2. Form plot of K_k, the waiting overhead, against x^1 and x^2 arranged back-to-back.

fixed sum k, against x^1 and x^2 arranged 'back-to-back'. The diagram depicts forms of the plot for the two cases – (i) $a_1 = a_2$ and (ii) $a_1 < a_2$. The plots for these two cases are intentionally superposed on the same graph in order to enable us readily see which algorithm can be expected to perform optimally in each of the cases.

Recalling the interpretation given earlier for $K_k(x^1, x^2)$, we immediately infer from fig. 2 that there is what is known as an optimal initial load configuration (x^1, x^2) for a fixed sum $x^1 + x^2 = k$, w.r.t. the resulting value of K_k. In the case of identical exponential servers this optimal initial configuration is just the balanced distribution $([k/2], [k/2])$. This case is identified by the plot corresponding to $a_1 = a_2$. On the other hand, in the more general case of distinct exponential servers ($a_1 < a_2$, *e.g.*,) the optimal initial configuration departs from the balanced one as suggested in fig. 2. Here it is a bit skewed, and is given by (x^1, x^2) such that $(x^1 + 1)/(x^2 + 1)$ has the closest possible value to the ratio a_2/a_1. This observation is the key to the following discussion aimed at a physical interpretation of Algorithm 1. Before proceeding with that we make a final observation that our foregoing considerations hold even in the general case where

$K_k(x^1, x^2)$ is interpreted using $x_t = (x^1, x^2)$ instead of $x_0 = (x^1, x^2)$ where $x^1 + x^2 = k$. In other words, consider the routing situation where $x_{t-} = (x^1, x^2)$, $x^1 + x^2 = k$, $t-$ denoting the instant just prior to the arrival. Once the arrival is routed, $x_{t+} = (x^1 + 1, x^2)$ or $x_{t+} = (x^1, x^2 + 1)$ so that $x^1 + x^2 + 1 = k + 1$. Now, $K_{(k+1)}$ can be interpreted with reference to x_{t+} which might have one of two possible values. It is to our obvious advantage to base the route for the arrival on that choice which leads to the smaller value for $K_{(k+1)}$. With this preparation we launch into the interpretive discussion. The coordinate space in fig. 2 is dissected into three slices. Within slice I, $x^1 \leq x^2$. For values of x_t^1 and x_t^2 falling in this slice, Algorithm 1 advocates routing to the first queue. This concurs with one's intuition – server 1 is the quicker (recall that $m_1 \geq m_2$) of the two and at time t queue 1 happens to be the shorter one; naturally, therefore, routing should be done to the first queue in order to achieve optimality.

Within slice II, however, things are a little strange. Though $x^1 > x^2$, Algorithm 1 still rules that routing be done to the first queue. One natural way of viewing this situation is to expect that the routing policy which makes for the 'proportionate' distribution of the load between the servers, exhibit optimal behaviour. Thus, this should take account of x_t^1, x_t^2, as well as the individual server capacities. In other words, although $x_t^1 > x_t^2$, since $m_1 > m_2$ (there would be no slice II if $m_1 = m_2$), *viz.*, server 1 on the average serves faster than does server 2, he can afford to take more load up to some 'limit', whose value is dependent on m_1 and m_2. Again Algorithm 1 is seen to reinforce our reasoning.

Inside slice III, $(x^1 > x^2)$ *and* $(not\ C(x^1, x^2))$. For values of x_t^1, x_t^2 corresponding to this coordinate slice, it is clear that the limit up to which server 1, by virtue of his faster average rate of service, can take excess load without upsetting the proportionate distribution of the existing load, has been exceeded. Therefore, as Algorithm 1 rightly points out, queue 2 is the optimal routing choice.

Observing the form plots of fig. 2, we are really trying to approach the minimum point in the K_k plot w.r.t. the load distribution. SS policy seeks to divide the load evenly between the two servers irrespective of their individual capacities (mean service times). Such a policy leads to an optimal behaviour only in the case of identical mean service times (case (i), $a_1 = a_2$, fig. 2). For the case $a_1 < a_2$, *viz.*, of distinct capacities, SS policy never approaches the minimum point on the K_k plot for case (ii) in fig. 2. In that case, however, our algorithm – Algorithm 1 – is seen (qualitatively at this stage) to make for the minimum point. In reality, Algorithm 1 coincides with SS policy only for the case $a_1 = a_2$.

The optimal routing algorithm developed is different from SS policy in the crucial region – slice II. In the next section, we shall exhibit a proof of optimality for Algorithm 1. In Section IV, we compare the behaviour of the two algorithms in the light of simulation results.

3. Proof of optimality

We present the proof in two stages. Let us assume in the sequel that Algorithm 1 is adopted as the dynamic routing policy for the queueing network with n parallel queues.

Let $x_t = (x_t^1, x_t^2, \ldots, x_t^n)$ be observed continuously; let $V(t, x^1, x^2, \ldots, x^n)$ denote the expected cost incurred over $[t, T]$ where $x_t = (x^1, x^2, \ldots, x^n)$. Finally, for any arrival epoch $t \in [0, T]$, let $t-$ and $t+$ denote instants immediately prior to, and after the arrival.

Lemma 1

(i) $V(T, x^1, x^2, \ldots, x^n) = \sum_{i=1}^{n} a_i x^i (x^i + 1)$.

(ii) If t is any arrival epoch, then

$$V(t-, x^1, x^2, \ldots, x^n) = V(t+, x^1, \ldots, x^i + 1, \ldots, x^n)$$

where $a_i(x^i + 1) = \min_i a_i(x^i + 1)$.

(iii) If there is no arrival during the interval $[t - dt, t]$, then

$$V(t - dt, x^1, x^2, \ldots, x^n) = \sum_{i=1}^{n} V(t, x^1, \ldots, (x^i - 1)^+, \ldots, x^n) \cdot m_i \, dt + \left(1 - \sum_{i=1}^{n} m_i \, dt\right) \cdot V(t, x^1, x^2, \ldots, x^n) + \sum_{i=1}^{n} x^i \, dt,$$

where $(x)^+ = x$ if $x > 0$, otherwise $(x)^+ = 0$.

Proof

Result (i) is a direct consequence of (2), defining the expression for the cost function $K(T)$, as substitution will readily reveal.

Result (ii) is but a formal mathematical statement of Algorithm 1. Since we have assumed that Algorithm 1 is being adopted as the routing policy for our network, result (ii) is trivially true.

As for result (iii), suppose that there is no arrival during the interval $[t - dt, t]$. If $x_{t-dt} = (x^1, x^2, \ldots, x^n)$, then the changes in x_t are caused only by service completions at the individual queues. It will thus be seen that result (iii) is easily obtained using the relation

$$V(t - dt, x_t^1, x^2, \ldots, x^n) = E[V(t, x_t^1, x_t^2, \ldots, x_t^n)/x_{t-dt} = (x^1, x^2, \ldots, x^n)].$$

In establishing the optimal behaviour of Algorithm 1, we employ an indirect path. We first develop an optimal routing policy from dynamic programming considerations and show later that Algorithm 1 behaves identically with this policy. For this, suppose that r_i be the probability that an arrival be routed to the ith queue, at some arrival epoch t. Evidently $\sum_{i=1}^{n} r_i = 1$. Again, the problem being one of dynamic routing, r_i, $(i = 1, 2, \ldots, n)$ in general vary with time. From dynamic programming considerations (An excellent treatment of dynamic programming is offered in Larson and Casti.[10]) the

expected cost must satisfy the following equation irrespective of the routing policy adopted.

$$V(t-,\ x^1,\ x^2,\ldots,\ x^n) = \sum_{i=1}^{n} V(t+,\ x^1,\ldots,\ x^i+1,\ldots,\ x^n)\cdot r_i \tag{9}$$

at any arrival epoch t, where $x_{t-} = (x^1, x^2, \ldots, x^n)$. Our interest is centered on arrival epochs rather than any other instants.

The principle of optimality[10] implies that any segment of an optimal state space trajectory between the two given points is still an optimal trajectory between the intermediate points framing the segment. In consequence, it is easy to infer from the recursive equation (9) that at any arrival epoch t, the optimal routing choice should seek to minimize $V(t+, x^1, \ldots, x^i + 1, \ldots, x^n)$. In consequence of this observation, we have the following policy.

Policy 1

Let t be any arrival epoch with $x_{t-} = (x^1, x^2, \ldots, x^n)$. The optimal routing choice for the arrival should be identified with the value of i over $\{1, 2, \ldots, n\}$ which minimizes $V(t+, x^1, \ldots, x^i + 1, \ldots, x^n)$.

In other words, Policy 1 suggests that r_i be made unity and other $r_j, j \neq i$ set to zero for the optimal routing choice i. The optimality of Policy 1 is the direct outcome of the validity of the recursive equation (9). We may now proceed to prove the optimality of Algorithm 1. We will need a couple of more definitions to achieve notational elegance. To this end, let

$$e_i = (0,\ 0, \ldots,\ 1,\ 0, \ldots,\ 0) \tag{10}$$

where the ith entry is unity and the rest are zeros in the n-tuple. Now suppose that for $x = (x^1, \ldots, x^n)$ an arrival is routed to the ith stream. The new state can be denoted by $x + e_i$, which for our convenience, we denote as $+_i(x)$. Thus,

$$+_i(x) = x + e_i \tag{11}$$

where x and e_i have been defined earlier.

Theorem 1

Algorithm 1 is optimal w.r.t. dynamically minimizing the cost function $K(T)$.

Proof

As outlined earlier, we establish optimality by proving identity with Policy 1. As rightly suits a recursive equation, we exhibit in what follows, a recursive proof.

Base step

Without loss of generality, let T be the final arrival epoch. Clearly $T+ = T$, the routing decision being assumed to consume infinitesimal time. With $x_{T-} = (x^1, x^2, \ldots, x^n)$, we have,

$$V(T-, x^1, x^2, \ldots, x^n) = \sum_{i=1}^{n} r_i \cdot V(T, +_i(x_{T-}))$$

$$= \sum_{i=1}^{n} r_i \left[\sum_{l=1}^{n} a_l \cdot x^l(x^l + 1) + 2a_i(x^i + 1) \right]. \tag{12}$$

Since in (12) the first term inside braces is a constant, a chosen j will minimize $V(T, +_i(x_{T-}))$ iff it minimizes $a_i(x^i + 1)$. Thus, r_j, where j is chosen to minimize $a_i(x^i + 1)$ should be made unity and r_i, $i \neq j$, set to zero. Note that r_i, $(i = 1, 2, \ldots, n)$ constitute the decision variables of any routing policy.

Recursive step

Let us assume the recursive hypothesis – Algorithm 1 and Policy 1 behave identically, or in other words Algorithm 1 shows optimal behaviour, for some instant $t+$, where t is an arrival epoch (Note that a decision about the queue to which a new customer is to be routed is made at the arrival time. This decision is based entirely on the server parameters and the state of the system (*i.e.* the lengths of queues) at that instant. Thus departures from the system do not affect the routing decision directly. They only affect it indirectly by changing the state of the system. Since the state of the system is anyway taken into account in the formulation of the optimal policy, it is not necessary to consider intervening service completions explicitly in the proof. It suffices just to consider the arrival epochs.)

viz., $[m = \arg\min_i a_i(x^i + 1)] \Longrightarrow$

$$m = \arg\min_i V(t+, +_i(x)) \tag{13}$$

where

$$x_{t-} = x = (x^1, \ldots, x^n).$$

Define $y^i = x^i$, $i \neq m$

$$y^m = x^m + 1. \tag{14}$$

Clearly y^i are the components of $+_m(x)$.

$$\text{Let } a_j(y^j + 1) = \min_i a_i(y^i + 1). \tag{15}$$

Two cases need to be distinguished now.

Case (i)

$j = m$.

This qualitatively means that the mth stream qualifies for routing as determined by the criterion of Algorithm 1, at two successive arrival epochs, assuming no intervening service completions.

Per Algorithm 1

$$V(t-,\ +_m(x)) = V(t+,\ +_m(+_m(x))). \tag{16}$$

Also, since the mth stream constitutes the optimal routing choice for the given criterion, for the distribution $+_m(x) = (x^1, \ldots, x^m + 1, \ldots, x^n)$, so it will for a distribution $+_i(x)$, $i = 1, 2, \ldots, n$. Thus we have (cf. Appendix I for a proof),

$$V(t-,\ +_i(x)) = V(t+,\ +_m(+_i(x)))\ (i = 1, 2, \ldots, n) \tag{17}$$

under the control of Algorithm 1.

For the present case

$$a_m(y^m + 1) = \min_i a_i(y^i + 1).$$

From this and the recursive hypothesis (13), with $y = +_m(x)$

$$V(t+,\ +_m(+_m(x))) = \min_i V(t+,\ +_i(+_m(x))). \tag{18}$$

as follows by modus ponens.

The LHS of (18) is evidently equal to $V(t-,\ +_m(x))$ for case (i), while for the RHS observe that

$$+_i(+_m(x)) = +_m(+_i(x)).$$

Thus,

$$V(t+,\ +_i(+_m(x))) = V(t+,\ +_m(+_i(x)))$$
$$= V(t-,\ +_i(x))$$

by virtue of (17).

All the foregoing considerations allow us to rewrite (18) as

$$V(t-,\ +_m(x)) = \min_i V(t-,\ +_i(x)). \tag{19}$$

This concludes the recursive step for case (i).

Case (ii)

$j \neq m$.

This means that the distribution of the queue lengths $\{x_i\}$ is such that for two consecutive arrivals between which there are no service completions, the same queue does not constitute the routing choice twice, for the criterion used by Algorithm 1.

In view of Algorithm 1,

$$V(t-,\ +_m(x)) = V(t+,\ +_j(+_m(x)))\ j \neq m. \tag{20}$$

For case (ii),

$$a_j(y^j + 1) = \min_i a_i(y^i + 1),\ j \neq m.$$

From this and the recursive hypothesis (13),

$$V(t+,\ +_j(+_m(x))) = \min_i V(t+,\ +_i(t_m(x))). \tag{21}$$

We also observe that

$$V(t-,\ +_i(x)) = V(t+,\ +_m(+_i(x))),\ i \neq m \tag{22a}$$

and

$$V(t-,\ +_m(x)) = V(t+,\ +_j(+_m(x))) \tag{22b}$$

under the control of Algorithm 1.

The RHS of (21) is transformed as follows:

$$+_i(+_m(x)) = +_m(+_i(x)),$$

$$V(t+,\ +_i(+_m(x))) = V(t+,\ +_m(+_i(x))),$$

$$= V(t-,\ +_i(x)),\ i \neq m.$$

As for $V(t+,\ +_m(+_m(x)))$ we know that it is greater than or equal to $V(t+,\ +_j(+_m(x)))$. Hence we can write

$$V(t-,\ +_m(x)) = \min_i V(t-,\ +_i(x)) \tag{23}$$

by virtue of (21), (22a, b), and the above considerations. That concludes the recursive step and the proof is complete.

Algorithm 1 and Policy 1 have been shown to behave identically and hence the optimality of Algorithm 1 w.r.t. dynamically minimizing the cost function $K(T)$.

4. Simulation results and inferences

Simulation of the problem was carried out using SIMULA 67, a powerful programming language providing for discrete event simulation, on the DEC 1090 system running the TOPS 10 operating system. The following user-specifiable features were incorporated in the development of the program:

(i) selection of number of queues,
(ii) selection of a service policy out of several,
(iii) selection of mean service rates of each of the exponential servers,
(iv) selection of the mean arrival rate of customers.

Table I
Performance of routing policies

Sl. no.	No. of arrivals	Waiting overhead cost		
		RR Policy*	OP Policy†	SS Policy‡
1	174	221·00	13·20	56·17
2	163	298·57	29·00	58·17
3	87	114·00	24·00	48·00
4	96	73·00	10·00	44·00
5	220	666·53	21·00	56·43
6	112	223·00	15·00	39·00
7	222	613·00	25·00	117·50
8	116	47·00	12·00	37·00
9	190	99·80	22·00	63·43
10	154	231·00	20·00	45·00
11	199	233·43	34·00	88.33
12	147	183·00	14.08	46·00

*Round-Robin Policy; †Optimal Policy; ‡Send-to-Shortest queue policy.
* The theory does not make any assumptions on the nature of the arrival time distribution. However, in computer simulation, for the sake of convenience, the Poisson arrival process was used throughout.

Incorporation of these features has facilitated computation of cost for various values of the parameters involved. Three distinct service policies – Round-Robin Policy (RR), Send-to-Shortest-queue Policy (SS), and Optimal Policy (OP) were implemented and cost computations carried out for diverse values of number of queues, mean service rates, mean arrival rate, etc. The entire collection of simulation results is compiled in Lakshmanan's thesis[11]. However, in this paper we have collected in Appendix II a small typical portion of the results for our reflection. Specifically, for fixed values of the mean arrival rate, the number of queues, and the individual mean service rates given, we compare in Table I the performances of the three routing policies. RR has been included in simulation for purposes of comparison of performance. It is readily seen that the OP developed in the preceding section by far produces the best performance w.r.t. the cost defined for the parallel queueing network. These results bear out the desirability of adopting the OP in minimizing the running system cost – the waiting overhead defined by Gallager[2].

5. Conclusion

In this paper, we have proposed a generalized version of the dynamic routing problem of the type most likely to be manifest in real life applications like computer network message switching and transfer line production systems. We have developed a routing algorithm, established its optimality, and verified the same through computer simulation. Before we conclude, we make a note that a modified version of the same routing

problem where the individual mean service rates of the exponential servers are not known *a priori*, is posed in Lakshmanan's thesis[11] and a learning automaton model of the modified problem has been proposed. It is expected that this paper will act as a stimulus for further research in extensions of the routing problem such as service centers with non-exponential servers.

References

1. Fratta, L., Gerla, M. and Kleinrock, L. — The flow deviation method: An approach to store- and forward communication networks, *Networks*, 1973, 3, 97–133.
2. Gallager, R. — A minimum delay routing algorithm using distributed computation, *IEEE Trans. Commun.*, 1977, COM-25, 73–85.
3. Ephremides, A., Varaiya, P. and Walrand, J. — A simple dynamic routing problem, *IEEE Trans. Automat. Control*, 1980, AC-25, 690–697.
4. Winston, W. — Optimality of the shortest line discipline, *J. Appl. Prob.*, 1977, 14, 181–189.
5. Agrawala, A.K., Coffman, E.G., Garey, M.R. and Tripathy, S.K. — A stochastic optimization algorithm minimizing expected flow times on uniform processors, *IEEE Trans. Computers*, 1984, C-33, 351–356.
6. Yum, T.S.P. — The design and analysis of a semi-dynamic deterministic routing rule, *IEEE Trans. Commns*, 1981, COM-29, 498–504.
7. Lin, W. and Kumar, P.R. — Optimal control of a queueing system with two heterogeneous servers, *IEEE Trans. Autom. Control*, 1984, AC-29, 696–703.
8. Rosberg, Z. and Towsley, D. — Customer routing to parallel servers with different rates, *IEEE Trans. Autom. Control*, 1985, AC-30, 1140–1143.
9. Buzacott, J.A. and Hanifin, L.F. — Models of automatic transfer lines with inventory banks: A review and comparison, *AIIE Trans.*, 1978, 10, 197–207.
10. Larson, R.F. and Casti, J.L. — *Principles of dynamic programming*, Part I, Marcel Dekker, 1977.
11. Lakshmanan, V.S. — *Learning in queueing environments*, M.E. Thesis, Indian Institute of Science, Bangalore, 1983.

Appendix I

Here we show how to deduce (17) from the various considerations applicable to case (i) of the proof for Theorem 1, given in Section III. To this end, let us define

$$Z^i = x^i + 1 \text{ for the given } i \neq m$$

$$Z^l = x^l,\ l \neq i\ (l = 1, \ldots, n). \tag{A1}$$

From (15) it follows for case (i) ($j = m$) that

$$a_m(y^m + 1) = \min_l a_l(y^l + 1). \tag{A2}$$

From the definition of y^l (14), this in turn implies that

$$a_m(x^m + 1) = \min_l a_l(x^l + 1). \tag{A3}$$

Now, (A2) and (A3) together imply

$$a_m(Z^m + 1) = \min_l a_l(Z^l + 1). \tag{A4}$$

It is easily seen that (A4) is true for all $i = 1, 2, \ldots, n$. Realizing that Z^l are the components of $+_i(x)$, where $x = (x^1, x^2, \ldots, x^n)$, (A4) is seen to indicate that the mth queue is the optimal routing choice for a distribution given by $+_i(x)$ at some arrival epoch. A formal version of this statement is

$$V(t-, +_i(x)) = V(t+, +_m(+_i(x))) \tag{A5}$$

which is the same as (17)

Appendix II

Results of simulation on the generalized dynamic routing problem

Number of servers (n) = 10
Mean arrival* rate (λ) = 10

Individual mean service rates

(i) $m_1 = 4$
(ii) $m_2 = 5$
(iii) $m_3 = 4.6$
(iv) $m_4 = 5.8$
(v) $m_5 = 6$
(vi) $m_6 = 12$
(vii) $m_7 = 16.8$
(viii) $m_8 = 11.8$
(ix) $m_9 = 9$
(x) $m_{10} = 25$.

$[0, T]$ is the interval of observation. The *waiting overhead cost* defined in (2) relates to this interval. A *simulation time* of 3000 sec. was used. For successive intervals $[0, T]$ with $T = 15$ sec., the waiting cost was computed for each routing policy. A portion of the complete simulation results[11] is reproduced here.

The (random) number of arrivals for several successive time slices of the same duration T and the associated waiting cost incurred by each policy are given in Table I.

* See footnote below Table I.