Queueing Systems 43, 147–165, 2003

Routing in Queues with Delayed Information

NELLY LITVAK N.litvak@math.utwente.nl
Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands, EURANDOM, P.O. Box 513, 5600 MB Eindhoven, The Netherlands

URI YECHIALI uriy@post.tau.ac.il
Department of Statistics and Operations Research, School of Mathematical Sciences, Tel-Aviv University, Tel-Aviv 69978, Israel

Received 26 September 2001; Revised 13 August 2002

Abstract. We compare two routing-control strategies in a high-speed communication network with c parallel channels (routes), where information on service completions in down-stream servers is randomly delayed. The controller can either hold arriving messages in a common buffer, dispatching them to servers only when the delayed information becomes available (Wait option), or route jobs to the various channels, in a round-robin fashion, immediately upon their arrival. Interpreting the delays as servers's vacations and considering overall queue sizes as a measure of performance, we show that the Wait strategy is superior as long as the mean information delay is below a threshold. We calculate threshold values for various combinations of load and c and show that, for a given load, the threshold increases with c and, for fixed c, the threshold decreases with an increasing load. If information is delayed on arrival instants, rather than on service completions, we show that the system can be viewed as a tandem queue and derive a generalization of a queue-decomposition result obtained by Altman, Kofman and Yechiali.

Keywords: routing, delayed information, multiple-server queues, vacation models, decomposition

1. Introduction

The main goal of this work is to compare, in terms of overall queue sizes, two routing-control policies in communication networks where information on service completions is delayed. In general, analysis and control of queueing systems with delayed information, either on service completions or on arrivals, are complex issues that have been studied very little in the literature (see, e.g., [1,4]). These issues are crucial in high-speed networks where routing decisions have to be made based on delayed information on the actual state of down-stream nodes. The lack of full information makes the problem of *optimal* routing of packets, among various possible channels, extremely difficult.

Consider a network with c parallel channels (servers) and a controller. Variable-length messages (jobs) arrive randomly, and the controller has to route (assign) them to the various channels. If the controller has full information on the state of each server, then holding a central single buffer for all queues and assigning a job to a server as soon

as the latter becomes available, is the best policy in terms of minimizing queue lengths and waiting times. However, if the information about the actual state of each queue reaches the controller only after some considerable delay, then the problem of optimal assignment of jobs to the various queues becomes much more complex.

Suppose, indeed, that the information about each service completion reaches the controller only after some random delay. Then one can think of two possible routing strategies: Wait or No-Wait policies. In the Wait option all arriving messages are held in a single common buffer and the controller forwards a message to a server only when he/she knows for sure that the server is free. This procedure implies that there are no separate buffers for the individual servers and that each arriving message will be assigned to a server only after an additional random delay, which clearly increases queue sizes and waiting times of the jobs. According to the No-Wait option the controller assigns jobs to the various queues "blindly", as soon as they arrive, without fully knowing the state of each server. In such a case the assignment can be done either randomly, or by using a round-robin (cyclic) procedure. The cyclic procedure has been shown to be much better than the (independent and) uniform random assignment in terms of minimizing waiting times (see, e.g., [8]). Another random assignment method is advocated in [4]. The procedure studied there is, given the *delayed* information on the queue size of each of the c servers, the router, upon arrival of a new job, selects randomly two out of the c processors, and dispatches the newly arriving job to the least loaded processor among the two. By applying a fluid-limit approach, leading to a deterministic model corresponding to the limiting system as $c \to \infty$, it is demonstrated via simulations that "the strategy ... performs well under a large range of system parameters".

In this work we will mainly analyze the Wait policy of holding all arriving jobs in a common buffer, which heavily depends on the delays in obtaining information on service completions, and will compare it to the No-Wait round-robin strategy. We will show that when the mean delay of obtaining information is below a (calculated) threshold value, the Wait policy is better than the No-Wait policy of assigning the jobs "blindly", thus partially answering the question of efficient routing when information on service completions is delayed. The key observation is that the analysis of the Wait policy leads to a special vacation model where each server, after completing servicing a job, takes a random-length vacation.

The motivation for the Wait policy stems from the following observation. Consider a $G_1/M/c$ queueing system with i.i.d. inter-arrival times T having probability distribution function (p.d.f.) $G_1(t) = P(T \leqslant t)$ with mean $E(T) = 1/\lambda$. Service times B are i.i.d. with exponential p.d.f. and mean $E(B) = 1/\mu$. The system is stable iff $\rho = \lambda/(c\mu) < 1$. Call this configuration "system-1". Suppose that system-1 has to be partitioned into c separate (probabilistically identical) queues, each forming a $G_2/M/1$ queue with i.i.d. inter-arrival times X having p.d.f. $G_2(t) = P(X \leqslant t)$ and mean $E(X) = c/\lambda$. Call this configuration "system-2". Clearly, the traffic intensity for each separate queue in system-2 is $\rho = (\lambda/c)/\mu$, and system-2 is stable iff $\rho < 1$, similarly to system-1. The method by which X is generated from T could be either a probabilistic assignment, by which a newly arrived customer is routed to queue i $(i = 1, 2, \ldots, c)$

with probability $1/c$, or a cyclic (round-robin) mechanism by which the $(kc+i)$th arrival is assigned to queue i $(k = 0, 1, \ldots)$. It is important to indicate that in system-1 there is a single common buffer where all arriving jobs are accumulated and from which, having full information on the state of each server (busy or idle), the controller assigns jobs to free servers; whereas in system-2 there are c separate queues and the controller, regardless of whether or not he maintains real-time information on the various queue sizes or the state of each individual server, assigns each new arrival to a channel as soon as the job arrives.

Denote by $E[W_1]$ the mean waiting time (not including service) in system-1, and by $E[W_2]$ the corresponding value in system-2. Let $r(\rho) = E[W_2]/E[W_1]$. Then it has been shown in [8] that for G_1 belonging to the family of $\Gamma(n, \lambda n)$ distributions (i.e. Erlang (Gamma) p.d.f. with n exponential phases, each having mean $1/(\lambda n)$), the ratio $r(\rho)$ possesses the following properties:

(1) $r(\rho)$ is a monotone decreasing function of ρ, $0 \leqslant \rho \leqslant 1$.

(2) For a probabilistic assignment, $r(\rho)$ tends to infinity as ρ approaches 0, and

$$\lim_{\rho \to 1} r(\rho) = \frac{2cn - n + 1}{n + 1}.$$

(3) For a cyclic (round-robin) assignment

$$\lim_{\rho \to 0} r(\rho) = c(c!)^n,$$

whereas

$$\lim_{\rho \to 1} r(\rho) = \frac{cn + 1}{n + 1}.$$

That is, under probabilistic (random) assignment, the smallest value of $r(\rho)$ is $r(1) = c$ ($n = 1$, T exponential), while $r(1) \to 2c - 1$ when T becomes a deterministic random variable ($n = \infty$). However, for any n, $r(0) = \infty$. Under the cyclic assignment $r(0) = c(c!)$ for $n = 1$ (T exponential) and is approaching infinity as n becomes large (T deterministic), whereas $r(1) = (c + 1)/2$ for $n = 1$, while $r(1) = c$ for $n = \infty$.

To summarize, for the family of Erlang inter-arrival distributions, the mean waiting time of an individual job in system-2 is at least $(c+1)/2$ times larger than its corresponding waiting time in system-1. Indeed, a much smaller mean waiting time for individual jobs can be achieved if information on the system's state is available when assigning them to servers.

Thus, we present in section 2 an $M/M/c$-type queueing model where each server, after every service completion, takes an exponentially distributed vacation. In section 3 we present two methods of analysis for this model: (1) we use a (partial) generating functions approach (see, e.g., [2,3]) which requires finding roots of a polynomial in order to be able to calculate the (two-dimensional) steady-state probabilities; (2) we employ Neuts's [5] matrix-geometric formulation, which also requires numerical calculation of

a matrix R, which is the solution of a quadratic matrix equation in R. We implement the first solution for small values of c.

For larger values of c we use in section 4 a direct approximation formula for calculating mean queue size and waiting times and derive a threshold value such that, if the mean delay is below that value, it is better to wait for the delayed information to arrive before routing jobs to various channels, rather than routing them as soon as they arrive but with lack of information on system's state. We then extend the calculations to the case where the delay is *deterministic* and calculate the threshold value for this case as well. Finally, in section 5, we discuss some issues regarding delayed information on arrivals.

2. The model

Consider an $M/M/c$-type queueing system with c parallel servers, each capable of serving jobs at a rate of μ jobs per unit time. There is a common buffer from which the controller dispatches waiting jobs to servers. Such an assignment is performed only after the controller obtains information that the designated server is free and idle. However, contrary to the classical $M/M/c$ queue where information on service completions becomes available instantaneously, in our case this information is delayed. We assume that the length of the delay is an exponentially distributed random variable with mean $1/\gamma$. For the controller, the server becomes available only when the delay is over. This state of affairs leads to a two-dimensional birth-and-death process as follows. We interpret the delay of information on service completion as a vacation: following each service completion of a job the server takes an exponentially distributed vacation (with mean $1/\gamma$) at the termination of which it becomes available again. We say that a server is "*operative*" if it is not on vacation (i.e. either busy or idle, ready to serve).

The present model differs from that of Levy and Yechiali [2] in that in the latter a server takes a vacation only when the common buffer is empty, whereas in our case a server takes a vacation after each service completion, regardless of whether the common buffer is empty or not.

It readily follows that the stability condition for that model is

$$\rho = \frac{\lambda(\mu + \gamma)}{c\gamma\mu} < 1. \tag{1}$$

Condition (1) says that for each server the mean inter-arrival time c/λ should be greater than the quantity $1/\mu + 1/\gamma$, which can be interpreted as the mean duration of a generalized service time being composed of two consecutive stages: service and vacation.

Assume that condition (1) is satisfied. In a steady state, let N be the number of jobs in the system and J be the number of operative servers. Clearly, if $N \geqslant J$, all servers are busy and $N - J$ jobs are queueing. Further, let

$$p_{j,n} = \mathrm{P}(J = j,\ N = n), \quad j = 0, 1, \ldots, c;\ n \geqslant 0,$$

be the steady-state probabilities that there are j operative servers and n jobs in the system. Then, for $j = 0, 1, \ldots, c$, the balance equations are given by

$$\begin{aligned}
&\big[\lambda + (c-j)\gamma + n\mu\big]p_{j,n} = (n+1)\mu p_{j+1,n+1} + \lambda p_{j,n-1} + (c-j+1)\gamma p_{j-1,n}, \\
&\quad n < j, \\
&\big[\lambda + (c-j)\gamma + j\mu\big]p_{j,n} = (j+1)\mu p_{j+1,n+1} + \lambda p_{j,n-1} + (c-j+1)\gamma p_{j-1,n}, \\
&\quad n \geqslant j.
\end{aligned} \tag{2}$$

Here $p_{j,-1} = 0$ and $p_{-1,n} = p_{c+1,n} = 0$ for all $n \geqslant 0$.

Now, let L be the queue length in a steady state. Then

$$E[L] = \sum_{j=0}^{c} \sum_{n=j+1}^{\infty} (n-j)p_{j,n}.$$

In the next section we describe two ways to compute the steady-state probabilities $p_{j,n}$'s and the value $E[L]$ from equations (2). This would serve our goal to compare the two routing strategies in terms of the steady-state mean queue size and show under what conditions one strategy is better than the other.

3. Analysis

3.1. Analysis via generating functions

For $j = 0, 1, \ldots, c$ define the (partial) generating function

$$G_j(z) = \sum_{n=0}^{\infty} p_{j,n} z^n.$$

Then, multiplying every equation of (2) by z^n and summing over n, we obtain

$$\begin{aligned}
&\big[\lambda(1-z) + (c-j)\gamma + j\mu\big]G_j(z) \\
&\quad = (j+1)\mu z^{-1} G_{j+1}(z) + (c-j+1)\gamma G_{j-1}(z) \\
&\qquad + \sum_{n=0}^{j-1} (j-n)\mu p_{j,n} z^n - z^{-1} \sum_{n=0}^{j} (j-n+1)\mu p_{j+1,n} z^n.
\end{aligned} \tag{3}$$

Set

$$b_0(z) = -\mu p_{1,0};$$

$$b_j(z) = z \sum_{n=0}^{j-1} (j-n)\mu p_{j,n} z^n - \sum_{n=0}^{j} (j-n+1)\mu p_{j+1,n} z^n, \quad 1 \leqslant j \leqslant c-1;$$

$$b_c(z) = \sum_{n=0}^{c-1} (c-n)\mu p_{c,n} z^n$$

and

$$f_j(z) = z\big[\lambda(1-z) + (c-j)\gamma + j\mu\big], \quad 0 \leqslant j \leqslant c-1;$$
$$f_c(z) = \lambda(1-z) + c\mu.$$

Also, define the matrix $A(z)$ and the vectors $b(z)$ and $g(z)$ as

$$A(z) = \begin{bmatrix} f_0(z) & -\mu & 0 & 0 & \dots & 0 & 0 & 0 \\ -c\gamma z & f_1(z) & -2\mu & 0 & \dots & 0 & 0 & 0 \\ 0 & -(c-1)\gamma z & f_2(z) & -3\mu & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & -2\gamma z & f_{c-1}(z) & -c\mu \\ 0 & 0 & 0 & 0 & \dots & 0 & -\gamma & f_c(z) \end{bmatrix},$$

$$b(z) = \big(b_0(z), b_1(z), \dots, b_c(z)\big)^{\mathrm{T}},$$
$$g(z) = \big(G_0(z), G_1(z), \dots, G_c(z)\big)^{\mathrm{T}}.$$

Then (3) becomes: $A(z)g(z) = b(z)$.

To obtain $G_j(z)$ we use Cramer's rule and write $|A(z)|G_j(z) = |A_j(z)|,\ 0 \leqslant j \leqslant c$, where $|A|$ is the determinant of the matrix A, and $A_j(z)$ is a matrix obtained from $A(z)$ by replacing the jth column by $b(z)$. It follows that the functions $G_j(z)$ are expressed in terms of $c(c+1)/2$ unknown probabilities $p_{j,n},\ 0 \leqslant n < j \leqslant c$, appearing in the expressions for $b_j(z)$. From the first part of (2) we have $c(c+1)/2$ linear equations for those probabilities, but with c more variables $p_{j,j}$ for $j = 0, 1, \dots, c-1$. Using the equation for $p_{0,0}$ from the second part of (2) which involves only $p_{0,0}$ and $p_{1,1}$, we still have to find $c-1$ additional equations. The following theorem and the use of equations (6) and (7) in the sequel give $c-2$ of them.

Theorem 3.1. For any $c = 2c_0,\ c = 2c_0+1$, where $c_0 \geqslant 0$, the polynomial $|A(z)|$ has a root of multiplicity c_0 at $z_0 = 0$, a root at $z_* = 1$ and exactly $c - c_0 - 1$ roots in $(0, 1)$.

Proof. Let $q_0(z) = 1$, and define the minors of the diagonal of $A(z)$, starting from the lower right-hand side corner, as follows:

$$q_1(z) = f_c(z), \qquad q_2(z) = \begin{vmatrix} f_{c-1}(z) & -c\mu \\ -\gamma & f_c(z) \end{vmatrix}, \quad \dots, \quad q_{c+1}(z) = \big|A(z)\big|. \tag{4}$$

The polynomials $q_j(z),\ 0 \leqslant j \leqslant c+1$, satisfy the following equations:

$$\begin{aligned} q_1(z) &= f_c(z)q_0(z), \\ q_2(z) &= f_{c-1}(z)q_1(z) - c\mu\gamma, \\ q_{k+1}(z) &= f_{c-k}(z)q_k(z) - k(c-k+1)\mu\gamma z q_{k-1}(z), \quad 2 \leqslant k \leqslant c. \end{aligned} \tag{5}$$

From (4), (5) we see that

(a) $q_0(z)$ has no roots;

(b) $q_k(z)$, where $k \geqslant 1$, is a polynomial of degree $2k-1$;

(c) $q_k(z)$ and $q_{k+1}(z)$ have no joint roots in $(0,\infty)$, because, if they do have such a joint root, then it is also a root of $q_{k-1}(z), q_{k-2}(z), \ldots, q_0(z)$, but $q_0(z)$ possesses no roots;

(d) $q_1(0) > 0, q_2(0) < 0$;

(e) $q_{2l+1}(z)$ and $q_{2l+2}(z)$, where $l \geqslant 1$, have a root $z_0 = 0$ of multiplicity l; the lth derivative of $q_{2l+1}(z)$ at $z = 0$ has a sign $(-1)^l$; the lth derivative of $q_{2l+2}(z)$ at $z = 0$ has a sign $(-1)^{l+1}$;

(f) if $z' > 0$ is a root of $q_k(z)$, then $q_{k+1}(z')$ and $q_{k-1}(z')$ are opposite in sign to each other;

(g) $q_k(1) = (c!\mu^k)/(c-k)!,\ 1 \leqslant k \leqslant c$;

(h) $q_{c+1}(1) = 0,\ q'_{c+1}(1) = c!(\mu+\gamma)^{c-1}[c\mu\gamma - \lambda(\mu+\gamma)]$;

(i) the sign of $q_k(\infty)$ is $(-1)^k$.

Let us now subsequently consider the roots of $q_1(z), q_2(z), \ldots, q_{c+1}(z)$. Clearly, $q_1(z)$ has only one root $z_{1,1} = 1 + c\mu/\lambda > 1$. Further, $q_2(0) = -c\mu\gamma < 0, q_2(1) > 0$, $q_2(z_{1,1}) < 0,\ q_2(\infty) > 0$, and thus $q_2(z)$ has roots $z_{2,1} \in (0,1), z_{2,2} \in (1, z_{1,1})$ and $z_{2,3} \in (z_{1,1}, \infty)$. There are no other roots, because $q_2(z)$ is of degree 3. Similarly, $q_3(z)$ is of degree 5; it has a root $z_0 = 0$, a root $z_{3,1} \in (z_{2,1}, 1)$, and there are also three other roots: $z_{3,2} \in (1, z_{2,2}),\ z_{3,3} \in (z_{2,2},\ z_{2,3}),\ z_{3,4} \in (z_{2,3}, \infty)$. Now, $q_4(0) = 0$, $q_4(+0) > 0, q_4(z_{3,1}) < 0, q_4(1) > 0$. Hence, there are roots $z_0 = 0, z_{4,1} \in (0, z_{3,1})$ and $z_{4,2} \in (z_{3,1}, 1)$. Also, $q_4(z)$ has four other roots in $(1,\infty)$. Proceeding further, we see that polynomial $q_{c+1}(z) = |A(z)|$ whose degree is $2c+1$ has a root of multiplicity c_0 at $z_0 = 0$. Also, it has a root $z_* = 1$. Since (1) implies that $q'_{c+1}(1) > 0$, it follows that there are roots $z_{c+1,l} \in (0,1),\ l = 1, 2, \ldots, c - c_0 - 1$, and $c+1$ other roots in $(1,\infty)$. This completes the proof. □

Now, from theorem 3.1 we have

$$\frac{\mathrm{d}^k}{\mathrm{d}z^k}\big|A_j(z)\big|\bigg|_{z=0} = 0, \quad 0 \leqslant j \leqslant c;\ 1 \leqslant k < c_0, \tag{6}$$

$$\big|A_j(z_{c+1,l})\big| = 0, \quad 0 \leqslant j \leqslant c;\ 1 \leqslant l < c - c_0. \tag{7}$$

For $0 \leqslant j \leqslant c$, equations (6), as well as equations (7), differ only by a constant multiplier. Hence, from (6) and (7) we get $c-2$ new equations for the unknown probabilities. In order to use these equations, we have to find at least one of the determinants $|A_j(z)|$, $j = 0, 1, \ldots, c$. The determinant $|A_0(z)|$ can be found by the following recursive procedure. Put

$$a_0(z) = b_c(z); \qquad a_k(z) = b_{c-k}(z)q_k(z) + (c-k+1)\mu a_{k-1}(z), \quad k = 1, \ldots, c. \tag{8}$$

Then $|A_0(z)| = a_c(z)$.

We now need one more equation in order to solve for the $G_j(z)$. This equation comes from the normalizing condition. Denote $p_j = G_j(1),\ j = 0, \ldots, c$. The normalizing condition is

$$\sum_{j=0}^{c} p_j = 1. \tag{9}$$

From theorem 3.1 it follows that $|A(z)| = (z-1)B(z)$ and $|A_j(z)| = (z-1)B_j(z),\ j = 0, \ldots, c$, where $B(z)$ and $B_j(z)$ are some polynomials in z, and $B(1) = q'_{c+1}(1)$. Since $G_j(z) = B_j(z)/B(z)$, the normalizing condition can be rewritten as

$$\sum_{j=0}^{c} B_j(1) = B(1) = q'_{c+1}(1) = c!(\mu+\gamma)^{c-1}\big[c\mu\gamma - \lambda(\mu+\gamma)\big]$$

giving the last equation needed to completely determine the generating functions $G_j(z),\ j = 0, 1, \ldots, c$.

However, if one is mainly interested in finding the mean queue size $E[L]$, then computing the determinants $|A_j(z)|$ for all $j = 0, 1, \ldots, c$ can be avoided. Instead, it might be easier to get the needed number of equations by considering p_j, $j = 0, 1, \ldots, c$, as additional unknowns. Then the normalizing condition remains in the form (9), and we can use (3) to find $c+1$ additional equations. By substituting $z = 1$ in (3), we get

$$\begin{aligned}
&\big[(c-j)\gamma + j\mu\big]p_j \\
&\quad = (j+1)\mu p_{j+1} + (c-j+1)\gamma p_{j-1} \\
&\qquad + \sum_{n=0}^{j-1}(j-n)\mu p_{j,n} - \sum_{n=0}^{j}(j-n+1)\mu p_{j+1,n}, \quad j = 0, \ldots, c,
\end{aligned} \tag{10}$$

which gives c linear independent equations. Further, differentiating equations (3) at point $z = 1$ we obtain

$$\begin{aligned}
&-\lambda p_j + \big[(c-j)\gamma + j\mu\big]n_j \\
&\quad = -(j+1)\mu p_{j+1} + (j+1)\mu n_{j+1} + (c-j+1)\gamma n_{j-1} \\
&\qquad + \mu\sum_{n=0}^{j-1} n(j-n)p_{j,n} + \mu\sum_{n=0}^{j}(1-n)(j-n+1)p_{j+1,n}, \quad j = 0, \ldots, c,
\end{aligned} \tag{11}$$

where $n_j = G'_j(1)$. Summing over $j = 0, \ldots, c$ and using the normalization, we derive the last needed equation for the p_j's:

$$\sum_{j=1}^{c} jp_j - \sum_{j=1}^{c}\sum_{n=0}^{j-1}(j-n)p_{j,n} = \frac{\lambda}{\mu}. \tag{12}$$

Formula (12) can also be interpreted in the following way. In the left-hand side, the first term equals to $E[J]$, and the second term equals to $E[J_0]$, where J_0 is the number of operative but idle servers in a steady state. Note that $J_1 = J - J_0$ equals to the number of busy servers. Thus, it follows from (12) that $E[J_1] = \lambda/\mu$. Furthermore, since each busy server can serve only one job at a time, we get $J_1 = N - L$. This yields

$$E[L] = E[N] - \frac{\lambda}{\mu} = \sum_{j=0}^{c} n_j - \frac{\lambda}{\mu}, \tag{13}$$

where $n_j = \sum_{n=0}^{\infty} np_{j,n} = G'_j(1)$. Hence, in order to find $E[L]$, it suffices to compute the n_j's. Since (11) gives c linear independent equations for $c+1$ variables $n_0, \ldots, n_c$, we need one more equation which we obtain by taking the second derivative of (3) at $z = 1$, summing over $j = 0, \ldots, c$ and applying (12). This yields

$$\lambda E[N] = \mu \sum_{j=1}^{c} jn_j - \lambda - \mu \sum_{j=1}^{c} \sum_{n=0}^{j-1} n(j-n)p_{j,n}. \tag{14}$$

Now $n_0, \ldots, n_c$ can be computed from (11) and (14). Then we apply (13) to find $E[L]$.

Equation (14) can be also rewritten as $\lambda(1 + E[N]) = \mu(E[JN] - E[J_0N])$. Together with (13), this enables us to find the covariance between N, the number of jobs in the system, and J_1, the number of busy servers:

$$\mathrm{cov}[N, J_1] = E[J_1N] - E[J_1]E[N] = E\big[(J - J_0)N\big] - \frac{\lambda}{\mu}E[N] = \frac{\lambda}{\mu}.$$

As mentioned earlier and will further be exploited in section 4, a server's vacation can be viewed as the second phase of a generalized service. Since the mean number of arrivals per unit time is λ and since each service is followed by a vacation with mean $1/\gamma$, it follows that the mean number of servers on vacation equals $E[c - J] = \lambda/\gamma$, implying $E[J] = c - \lambda/\gamma$. If there are no delays, i.e. $1/\gamma = 0$, we merely get $E[J] = c$. Finally, the mean number of servers involved in the generalized service (either busy or on vacation) is given by $E[c - J_0] = \lambda(\gamma + \mu)/\mu\gamma = c\rho$, so that $E[J_0] = c(1 - \rho)$.

Implementation. The discussion above suggests the following algorithm of finding $E[L]$ when $c = 2c_0$ or $c = 2c_0 + 1$.

1. Initialization. Set $\mathcal{S}_c = \emptyset$.
2. If $c - c_0 > 1$ then:
   (a) compute $|A(z)|$ using (5);
   (b) find the roots $z_{c+1,l} \in (0, 1)$, $1 \leqslant l < c - c_0$, of $|A(z)|$;
   (c) compute $|A_0(z)|$ using (8);
   (d) add equations (7) for $j = 0$ to $\mathcal{S}_c$;
   (e) if $c_0 > 1$ then add equations (6) for $j = 0$ to $\mathcal{S}_c$.

3. Add to $\mathcal{S}_c$ equations (2) for $0 \leqslant n < j \leqslant c$ and $j = n = 0$; (10) and (11) for $0 \leqslant j < c$; (9), (12) and (14).
4. Solve the system of equations $\mathcal{S}_c$ with respect to $p_{j,n}$, p_j and n_j, $0 \leqslant j \leqslant c$; $0 \leqslant n \leqslant \min\{c-1, j\}$.
5. Find $E(L)$ from (13).

When c is not too large, this algorithm is suitable for a software like MAPLE or MATLAB. In the present work, we used MAPLE.

For $c = 2$, the polynomial $|A(z)|$ does not have roots on $(0, 1)$. Thus, there is no need to compute $|A(z)|$ and $|A_0(z)|$. The algorithm results in a closed form expression for $E[L]$. This formula can be written in terms of the values ρ and ξ, where $\rho = \lambda(\mu+\gamma)/(c\mu\gamma)$ is the utilization factor, and $\xi = \mu/\gamma$ is the ratio between the mean delay and the mean service time:

$$\begin{aligned} E[L] = {} & 2\rho^3\big(\xi^6 + 3\xi^5\rho + 5\xi^5 + 7\xi^4\rho + 2\xi^4\rho^2 + 12\xi^4 + 9\xi^3\rho + 2\xi^3\rho^2 \\ & + 16\xi^3 + 7\xi^2\rho + 2\xi^2\rho^2 + 12\xi^2 + 3\xi\rho + 5\xi + 1\big) \\ & \times\big[(1+\xi)^2(1-\rho)\big(\xi^4 + \xi^4\rho + 4\xi^3 + 7\xi^3\rho + 3\xi^3\rho^2 + 6\xi^2\rho^2 \\ & + 4\xi^2\rho^3 + 12\xi^2\rho + 6\xi^2 + 4\xi + 7\xi\rho + 3\xi\rho^2 + \rho + 1\big)\big]^{-1}. \end{aligned}$$

When the system does not have delays, i.e. $\xi = 0$, the above formula reduces to $E[L] = 2\rho^3/(1-\rho^2)$ which is the mean queue length in the $M/M/2$ queue (see formula (16) below).

For $c > 2$, the polynomial $|A(z)|$ has roots in $(0, 1)$. In general, these roots can not be analytically expressed via the parameters, confining us to a numerical solution. It is clear, however, that $E[L]$ can always be seen as a function of ρ and ξ (without loss of generality, we can choose time units in such a way that $\lambda = 1$).

3.2. Matrix-geometric approach

Following [5, section 6.3], we construct a quasi birth-and-death process with generator $\widetilde{Q}$ given by

$$\widetilde{Q} = \begin{bmatrix} A_{0,0} & A_{0,1} & 0 & \dots & & & & & \\ A_{1,0} & A_{1,1} & A_{1,2} & \dots & & & & & \\ & & & \dots & & & & & \\ & & & & A_{c-1,0} & A_{c-1,1} & A_{c-1,2} & & \\ & & & & & A_2 & A_1 & A_0 & \\ & & & & & & A_2 & A_1 & \dots \\ & & & & & & & \dots & \end{bmatrix}.$$

The square blocks of dimension $(c+1)\times(c+1)$ are defined as

$$(A_{0,0})_{i,h}=\begin{cases}-\lambda-(c-i)\gamma, & 0\leqslant i=h\leqslant c,\\ (c-i)\gamma, & 0\leqslant i=h-1\leqslant c-1,\\ 0, & \text{otherwise;}\end{cases}$$

$$A_{0,1}=A_{1,2}=\cdots=A_{c-1,2}=A_0=\operatorname{diag}\{\lambda,\lambda,\ldots,\lambda\};$$

$$(A_{n,0})_{i,h}=\begin{cases}\mu\min(i,n), & 0\leqslant h=i-1\leqslant c-1,\\ 0, & \text{otherwise,}\end{cases}\qquad \text{for } 1\leqslant n\leqslant c;$$

$$(A_{n,1})_{i,h}=\begin{cases}-\lambda-(c-i)\gamma-\mu\min(i,n), & 0\leqslant i=h\leqslant c,\\ (c-i)\gamma, & 0\leqslant i=h-1\leqslant c-1,\\ 0, & \text{otherwise,}\end{cases}\qquad \text{for } 1\leqslant n\leqslant c;$$

$$A_2=A_{c,0};$$

$$A_1=A_{c,1}.$$

Here $(A)_{i,h}$ for $0\leqslant i,h\leqslant c$ is the element of the ith row and hth column of the matrix A.

The matrix $Q=A_0+A_1+A_2$ is the generator of the classical machine repair model with μ as a breakdown rate:

$$\begin{bmatrix}
-c\gamma & c\gamma & 0 & 0 & \ldots & 0 & 0 & 0\\
\mu & -(c-1)\gamma-\mu & (c-1)\gamma & 0 & \ldots & 0 & 0 & 0\\
0 & 2\mu & -(c-2)\gamma-2\mu & (c-2)\gamma & \ldots & 0 & 0 & 0\\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots\\
0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0\\
0 & 0 & 0 & 0 & \ldots & (c-1)\mu & -\gamma-(c-1)\mu & \gamma\\
0 & 0 & 0 & 0 & \ldots & 0 & c\mu & -c\mu
\end{bmatrix}.$$

In such a machine repair model each server is considered, independently, as alternating between two phases: service (mean $1/\mu$) and breakdown, or vacation (mean $1/\gamma$). Let $\pi=(\pi_0,\pi_1,\ldots,\pi_c)$ be a stationary vector of the matrix Q, i.e. $\pi Q=\mathbf{0}$, where $\mathbf{0}=(0,0,\ldots,0)$. In that machine repair model π_j is the stationary probability that there are j operative servers. In the case that vacations start after each service completion, π_j can be interpreted as a stationary probability that there are j operative servers given that there are c or more jobs in the system. It is readily seen that

$$\pi_j=\binom{c}{j}\left(\frac{\gamma}{\gamma+\mu}\right)^j\left(\frac{\mu}{\gamma+\mu}\right)^{c-j}.$$

Substituting this expression in the stability condition from [5, p. 83]

$$\pi A_2 \mathbf{e} > \pi A_0 \mathbf{e},$$

where $\mathbf{e} = (1, 1, \ldots, 1)^{\mathrm{T}}$, we again yield (1) since

$$\pi A_0 \mathbf{e} = \lambda \sum_{j=0}^{c} \pi_j = \lambda \quad \text{and} \quad \pi A_2 \mathbf{e} = \sum_{j=0}^{c} \mu j \pi_j = \frac{c\mu\gamma}{\gamma + \mu}.$$

Now let $\mathbf{p}_n = (p_{0,n}, p_{1,n}, \ldots, p_{c,n})$. Then

$$\mathbf{p}_n = \mathbf{p}_{c-1} R^{n-c+1}, \quad n \geqslant c-1.$$

Here R is a minimal solution of the matrix quadratic equation $R^2 A_0 + R A_1 + A_2 = 0$. There are various comutational procedures for finding the matrix R (see [5, section 1.9]).

The vectors $\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_{c-1}$ can be found from the equations

$$\begin{aligned}
\mathbf{p}_0 A_{0,0} + \mathbf{p}_1 A_{1,0} &= 0, \\
\mathbf{p}_0 A_{0,1} + \mathbf{p}_1 A_{1,1} + \mathbf{p}_2 A_{2,0} &= 0, \\
\mathbf{p}_{n-1} A_{n-1,2} + \mathbf{p}_n A_{n,1} + \mathbf{p}_{n+1} A_{n+1,0} &= 0, \quad 2 \leqslant n \leqslant c-2, \\
\mathbf{p}_{c-2} A_{c-2,2} + \mathbf{p}_{c-1}(A_{c-1,1} + R A_{c,0}) &= 0, \\
\sum_{n=0}^{c-2} \mathbf{p}_n \mathbf{1} + \mathbf{p}_{c-1}[I - R]^{-1} \mathbf{1} &= 1,
\end{aligned}$$

where I is an identity matrix. The vector $\mathbf{n} = (n_0, \ldots, n_c)$ is now determined from

$$\mathbf{n} = \sum_{n=0}^{\infty} n \mathbf{p}_n = \sum_{n=0}^{c-2} n \mathbf{p}_n + (c-2)\mathbf{p}_{c-1}[I - R]^{-1} + \mathbf{p}_{c-1}[I - R]^{-2},$$

and we can apply formula (13) to find $E[L]$.

Computationally, the matrix-analytic approach is more powerful than the solution via generating functions, and it enables one to effectively solve the model even for large values of c.

4. Performance evaluation

In this section we compare the performance of the two routing procedures: Wait and No-Wait. The arrival flow is Poisson with intensity λ, and the service times are exponential with parameter μ. As a comparing performance measure we choose the mean queue length in steady-state.

Using the results of section 3, the mean queue length in the Wait strategy can be calculated in two ways, following sections 3.1 or 3.2, respectively. For $c = 2, 3$ we have implemented the algorithm from sections 3.1 utilizing the approach via generating functions. For larger values of c, the matrix-geometric approach from section 3.2 provides more efficient computational procedures. However, computational aspects are

not the main concern in this paper. Our ultimate goal is to compare the two routing policies in order to see whether the use of the delayed information can help improve the performance of the system. Therefore, instead of diving into detailed calculations, we use a direct approximation method. The comparison with explicit results for $c = 2, 3$ in table 1 below proves that the approximations provide a very good accuracy to serve our purpose. Moreover, the approximations can be easily generalized for non-exponential delays. For instant, in table 2 we give results for the case when the delays are deterministic.

As mentioned earlier, our model where every server takes vacation after each service completion can be interpreted as an $M/G/c$ queue with a generalized service time S composed of two consecutive phases: actual service B and vacation V. The *queue length* in such a system is the same as in a system where the service itself is distributed as $B + V$. Thus, in order to calculate $E[L]$ we use a two-moment approximation (see [7, p. 297]):

$$E\big[L^{\text{app}}\big] = \big(1 - c_S^2\big)E\big[L(\text{det})\big] + c_S^2 E\big[L(\text{exp})\big], \tag{15}$$

where $c_S^2 = Var[S]/E^2[S]$ is the variation coefficient of the generalized service time $S = B + V$; $E[L(\text{exp})]$ is the mean queue size in a $M/M/c$ queue with arrival rate λ and mean service time $E[S]$, and $E[L(\text{det})]$ is the corresponding mean for an $M/D/c$ queue for which we use Cosmetatos approximation

$$E\big[L(\text{det})\big] \approx E\big[L^{\text{app}}(\text{det})\big] = \frac{1}{2}\beta E\big[L(\text{exp})\big].$$

Here

$$\beta \equiv 1 + (1-\rho)(c-1)\frac{\sqrt{4+5c}-2}{16\rho c}$$

and $\rho = \lambda E[S]/c$. Further, for the $M/M/c$ queue with mean service time $E[S]$ we have

$$E\big[L(\text{exp})\big] = \frac{\rho(c\rho)^c}{c!(1-\rho)^2}p_0, \tag{16}$$

where

$$p_0 = \left[\sum_{k=0}^{c-1}\frac{(c\rho)^k}{k!} + \frac{(c\rho)^c}{c!(1-\rho)}\right]^{-1}.$$

For exponential service times and exponential vacation durations, where $E[S] = 1/\mu + 1/\gamma$ and $Var[S] = 1/\mu^2 + 1/\gamma^2$, we get $c_S^2 = (\gamma^2 + \mu^2)/(\gamma + \mu)^2$. This enables us to write the right-hand side of equation (15) as a function of ρ and the ratio $E[V]/E[B] = (1/\gamma)/(1/\mu) = \xi$:

$$\begin{aligned} E\big[L^{\text{app}}\big] &= E\big[L^{\text{app}}(\text{exponential delay})\big] \\ &= (1+\xi)^{-2}\big[\xi^2 + \beta\xi + 1\big]E\big[L(\text{exp})\big]. \end{aligned} \tag{17}$$

Another approximation for the mean queue size in $M/G/c$ system is given by Nozaki and Ross [6] as follows:

$$E\big[L^{\text{app}}\big(M/G(S)/c\big)\big] \approx \frac{1+c_S^2}{2} E\big[L\big(M/M(S)/c\big)\big] = \frac{1+c_S^2}{2} E\big[L(\exp)\big].$$

Clearly, for $E[V] = 1/\gamma = 0$ we readily obtain, for both approximations, the classical $M/M/c$ queue with arrival rate λ, mean service time $E[S] = 1/\mu$ and utilization factor $\rho_0 = \lambda/(c\mu)$. Note that if $c_S^2 < 1$, then the Cosmetatos approximation gives higher $E[L^{\text{app}}]$ values than the Nozaki–Ross approximation, and vice versa. This follows since $\beta > 1$. Note also that, when $c = 1$, $\beta = 1$, making the two approximations equal.

To ensure stability, condition (1) requires that the ratio $\xi = \mu/\gamma$ must be smaller than a critical value

$$\xi^{\text{crit}} = \frac{1-\rho_0}{\rho_0}.$$

Moreover, for fixed μ, $E[L^{\text{app}}]$ increases with ξ.

Let us now compare the mean queue length $E[L]$ in the Wait system, where the controller assigns a newly arrived job to a server only (and immediately) after the former obtains the (delayed) information that the server has completed servicing an earlier job, with the mean queue length $E[L(\text{cyclic})]$ in No-Wait system, where the controller assigns jobs to servers "blindly," as soon as they arrive, following the round-robin procedure. In the latter case there are c separate subsystems, each being an $E_c/M/1$ queue with mean arrival rate λ/c, mean service time $1/\mu$ and $\rho_0 = (\lambda/c)/\mu$. The mean queueing time is given by (see [8])

$$E\big[W(E_c/M/1)\big] = \frac{\alpha}{\mu(1-\alpha)},$$

where α is the unique root in $(0, 1)$ of the equation

$$z = \left(\frac{\lambda}{\mu(1-z)+\lambda}\right)^c = \left(\frac{c\rho_0}{1-z+c\rho_0)}\right)^c.$$

Thus, the mean queue length for each individual queue is

$$E\big[L(E_c/M/1)\big] = \frac{\lambda}{c} E\big[W(E_c/M/1)\big] = \frac{\rho_0\alpha}{1-\alpha}.$$

It follows that the total mean queue size among all c separate queues is

$$E\big[L(\text{cyclic})\big] = \frac{c\rho_0\alpha}{1-\alpha}.$$

Since, for any $c > 1$, the value $E[L(\text{cyclic})]$ is greater than $E[L(M/M/c)]$ with the same ρ_0 (see [8]), and, for fixed ρ_0, the value $E[L]$ as well as $E[L^{\text{app}}]$ is an (increasing) function of ξ, there exists a threshold value $\xi^* \in (0, \xi^{\text{crit}})$ such that $E[L] \approx E[L^{\text{app}}]$ becomes equal to $E[L(\text{cyclic})]$. Thus, as long as $\xi < \xi^*$, it is better to keep all arriving

Table 1
Threshold values ξ^* when delays are exponential.

		$c=1$	$c=2$	$c=3$	$c=5$	$c=10$
$\rho_0 = 0.2$	$E[L(M/M/c)]$	0.05	0.0167	0.0062	0.00096	0.00001
$\xi^{\text{crit}} = 4$	$E[L(\text{cyclic})]$	0.05	0.0414	0.0376	0.0353	0.0376
	ξ^* approximate	0	0.4263	0.6781	1.0332	1.5723
	ξ^* explicit	0	0.4303	0.6806		
$\rho_0 = 0.5$	$E[L(M/M/c)]$	0.5	0.3333	0.2368	0.1304	0.0361
$\xi^{\text{crit}} = 1$	$E[L(\text{cyclic})]$	0.5	0.6180	0.7406	0.9905	1.6232
	ξ^* approximate	0	0.2190	0.3373	0.4769	0.6411
	ξ^* explicit	0	0.2193	0.3371		
$\rho_0 = 0.8$	$E[L(M/M/c)]$	3.2	2.8444	2.5888	2.2170	1.6367
$\xi^{\text{crit}} = 0.25$	$E[L(\text{cyclic})]$	3.2	4.5564	5.9026	8.6090	15.3782
	ξ^* approximate	0	0.0744	0.1128	0.1534	0.1934
	ξ^* explicit	0	0.0744	0.1128		
$\rho_0 = 0.9$	$E[L(M/M/c)]$	8.1	7.6737	7.3535	6.8624	6.0186
$\xi^{\text{crit}} = 0.1111$	$E[L(\text{cyclic})]$	8.1	11.8589	15.6188	23.1394	41.9425
	ξ^* approximate	0	0.0352	0.0531	0.0715	0.0888
	ξ^* explicit	0	0.0352	0.0531		

jobs in a common buffer and operate the system by using the delayed information on service completions, rather than to assign arriving jobs "blindly," immediately upon arrival, to the various servers, even if it is done in a cyclic manner. However, if $\xi > \xi^*$, then the delays are too long, and it is better to have a separate buffer for each server and assign new jobs to the various servers following the round-robin procedure, without waiting for the delayed information to arrive.

In table 1 we give the values of ξ^*, for some different values of ρ_0 and c. Since $c_S^2 < 1$ when the delay is exponential (or deterministic), we use the two-moment approximation (15) for calculations. The results show that, for example, if $\rho_0 = 0.8$ then, for stability, we must have $\xi < \xi^{\text{crit}} = 0.25$, while, for $c = 10$, as long as $\xi < \xi^* = 0.1934$, it is preferred to operate the system by using the delayed information. It is seen that, for a given value of ρ_0, ξ^* increases with growing numbers of servers and that, for fixed c, ξ^* decreases when ρ_0 increases.

In fact, the above calculations can be extended to the case where the delay V is deterministic with $V = E[V] = 1/\gamma$. Here the approximations are especially valuable since the exact solution is not available. In such a case $E[S] = 1/\mu + 1/\gamma$, as before, but $Var[S] = 1/\mu^2$. This implies that $c_S^2 = (1+\xi)^{-2}$ and $1 - c_S^2 = (1+\xi)^{-2}(2\xi + \xi^2)$, where again $\xi = \mu/\gamma$. Thus, equation (15) leads to

$$E\big[L^{\text{app}}(\text{deterministic delay})\big] = (1+\xi)^{-2}\left[\left(\xi + \frac{\xi^2}{2}\right)\beta + 1\right]E\big[L(\text{exp})\big].$$

Table 2
Threshold values ξ^* when delays are deterministic.

		$c=1$	$c=2$	$c=3$	$c=5$	$c=10$
$\rho_0=0.2$ $\xi^{\text{crit}}=4$	ξ^*	0	0.4532	0.7199	1.0877	1.6341
$\rho_0=0.5$ $\xi^{\text{crit}}=1$	ξ^*	0	0.2256	0.3487	0.4924	0.6581
$\rho_0=0.8$ $\xi^{\text{crit}}=0.25$	ξ^*	0	0.0747	0.1134	0.1541	0.1942
$\rho_0=0.9$ $\xi^{\text{crit}}=0.1111$	ξ^*	0	0.0353	0.0532	0.0716	0.0889

It follows that (see (17))

$$E\big[L^{\text{app}}(\text{exponential delay})\big]-E\big[L^{\text{app}}(\text{deterministic delay})\big]$$
$$=(1+\xi)^{-2}\left[\xi^2\left(1-\frac{\beta}{2}\right)\right]E\big[L(\text{exp})\big].$$

Table 2 gives, similarly to table 1, values of ξ^* for various combinations of ρ_0 and c. It is seen that ξ^* in table 2 is slightly bigger than in table 1, but the difference decreases when ρ_0 increases.

5. Delayed information on arrivals

In this section we briefly discuss general queueing systems where information on arrivals, rather than on service completions, is delayed. We consider an arbitrary structure of arrival flow and arbitrary service discipline. Furthermore, we do not specify the number of servers or the distribution of the service times. It is assumed that the delays are independent and are all distributed as some random variable K. The control policy is such that a job is routed to a server only when the controller knows for sure that the common buffer, which holds all arriving messages, is not empty.

Altman et al. [1] considered a queueing system satisfying the conditions above. Specifically, they assumed that the delay $K \in \{0, 1, \ldots\}$ is a constant, and they studied a queue length process $\{X_n^K\}$, $n=-K,-K+1,\ldots,-1,0,1,\ldots$, in a discrete-time single-server queue with a stationary process $\{Y_n\}$ of batch arrivals (Y_n is the number of jobs arriving at time slot n) and arbitrary service times. For such a system, they proved that

$$X_n^K \overset{d}{=} X_n^0+\sum_{i=1}^{K} Y_{n-i+K}, \tag{18}$$

where $\{X_n^0\}$, $n=0,1,\ldots$, is the queue length process corresponding to no delay on arrival information ($K=0$), and $X_{-K}^K \overset{d}{=} X_0^0$. To explain formula (18) on an intuitive

level, the authors pointed out that the delays of information on arrivals may be looked upon as extra server's vacations, since the server may stay idle even if there are jobs present in the buffer. This observation is still true for more general systems with delayed information on arrivals. Thus, once again, there exists a strong connection between models with server's vacations and models with delayed information.

However, for delayed information on arrivals, the interpretation via vacation model is not that natural as for delayed information on service completions. For example, for the system studied in [1], the description of the analogous vacation model may read as follows: "*The server takes a vacation at time n if the buffer became empty at time* $n - K$. *The server becomes available again at time* $n + l$, *if the first arrival after time* $n - K$ *occurred at time* $n - K + l$." Within such a vacation model formula (18) expresses a direct decomposition, where the first term of the right-hand side is the queue length in the beginning of an arbitrary time slot in the system without vacations, and the second term is the queue length at the beginning of an arbitrary non-serving slot. Yet, this formula is still not very intuitive, and formally it requires a proof, which is of the same difficulty as the direct proof for the system with delayed information.

Using the general model, we propose another outlook of delayed information on arrivals: when a job arrives to the buffer, the controller does not know about its existence during some random delay. When the delay is over, the controller recognizes the job, and operates as if this job has just arrived. From the point of view of the controller, this system differs from the system without delayed information only by the characteristics of the arrival flow. For example, if the delay K is a constant, then the controller observes the same arrival flow, shifted by K time units. Actually, the controller may have no idea about the delays. However, in reality, the delays dictate that new arrivals have to waste additional time, hanging in the system, causing an increased queue length.

This situation can be better visualized with the aid of figure 1. A new job first arrives to a "preliminary" queue with infinite number of servers and immediately gets served, where its service time is distributed as K. Completing this delay, the job immediately proceeds to the main queue which is the same as our original system, but without delays (and, of course, with a modified structure of the arrival flow). The queue length in the original system with delayed information equals the number of jobs in the first queue in figure 1 plus the buffer content of the second queue. In general, such a tandem queue-

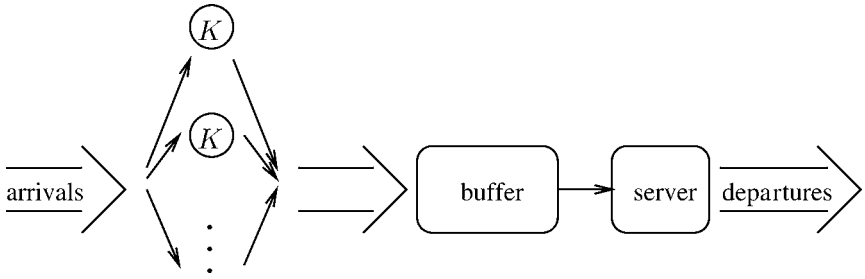

Figure 1. A delayed information on arrivals can be seen as a "preliminary" service.

ing system may not allow an explicit probabilistic analysis. Nevertheless, result (18) follows immediately from the above interpretation. One just has to note that, at the beginning of a time slot n, the number of jobs in the first queue is $\sum_{i=1}^{K} Y_{n-1-K+i}$, and the arrival flow to the second queue is the same as the original one, shifted by K slots. Moreover, the tandem-queue interpretation immediately yields proposition 5.1 bellow which is more general than (18).

Proposition 5.1. Let $(A(t), t \geqslant 0)$ be an arrival process of some queuing system. Let $(X^K(t), t \geqslant 0)$, where $K = \text{const} \geqslant 0$, be a queue length process in this system when information on arrivals is delayed by K time units. If $X^K(0) \stackrel{d}{=} X^0(0)$, then for any $t \geqslant 0$,

$$X^K(t) \stackrel{d}{=} X^0\big((t-K) \vee 0\big) + A(t) - A\big((t-K) \vee 0\big). \tag{19}$$

Using the tandem-queue interpretation, one can also derive some results when the delay K is random. For example, the following proposition is based on the well-know fact that in a steady-state, the number of jobs in $M(\lambda)/G/\infty$ queue has a Poisson distribution with parameter [λ(mean service time)], and the departure flow from such a queue is Poisson(λ).

Proposition 5.2. Let X^K be distributed as a steady-state queue length in some queueing system where arrival process is Poisson(λ) and information on arrivals is obtained with random delay K. Then

$$X^K \stackrel{d}{=} X^0 + Y,$$

where Y is a Poisson($\lambda E[K]$) random variable independent of X^0.

To conclude, we note that there is a considerable difference between delays of information on arrivals and delays on service completions. First of all, the delays of information on arrivals do not play any role in stability conditions, in contrast to delayed information on service completions. Second, the delayed information on arrivals increases waiting times merely by the length of the delay, and, in general, changes the original structure of the arrival flow. However, the delays of information on service completions affect the system via the extended "generalized" service which increases queue lengths and waiting times significantly when the load is high.

Acknowledgements

We would like to acknowledge Daniel Kofman for discussions that motivated this work. We are grateful to an anonymous referee whose constructive comments helped improving the presentation of the paper.

References

[1] E. Altman, D. Kofman and U. Yechiali, Discrete time queues with delayed information, Queueing Systems 19 (1995) 361–376.
[2] Y. Levy and U. Yechiali, An $M/M/s$ queue with server's vacations, INFOR 14 (1976) 153–163.
[3] I.L. Mitrany and B. Avi-Itzhak, A many server queue with service interruptions, Oper. Res. 16 (1968) 628–638.
[4] M. Mitzenmacher, How useful is old information, IEEE Trans. Parallel Distribution Systems 11 (2000) 6–20.
[5] M.F. Neuts, *Matrix-Geometric Solutions in Stochastic Models – An Algorithmic Approach* (Johns Hopkins Univ. Press, Baltimore, MD, 1981).
[6] S.A. Nozaki and S.M. Ross, Approximation in finite capacity multi-server queues with Poisson arrivals, J. Appl. Probab. 15 (1978) 826–834.
[7] H.C. Tijms, *Stochastic Models: An Algorithmic Approach* (Wiley, New York, 1994).
[8] U. Yechiali, On the relative waiting times in the $GI/M/s$ and the $GI/M/1$ queueing systems, Oper. Res. Quart. 28 (1977) 325–337.