Probability in the Engineering and Informational Sciences, **32**, 2018, 323–339.
doi:10.1017/S0269964817000286

MULTI-CLASS RESOURCE SHARING WITH PREEMPTIVE PRIORITIES

Isi Mitrani
School of Computing Science, Newcastle University,
Newcastle upon Tyne, UK
E-mail: isi.mitrani@ncl.ac.uk

Different virtual machines can share servers, subject to resource constraints. Incoming jobs whose resource requirements cannot be satisfied are queued and receive service according to a preemptive-resume scheduling policy. The problem is to evaluate a cost function, including holding and server costs, with a view to searching for the optimal number of servers. A model with two job types is analyzed exactly and the results are used to develop accurate approximations, which are then extended to more than two classes. Numerical examples and comparisons with simulations are presented.

Keywords: queueing theory, simulation, stochastic modeling

1. INTRODUCTION

This paper is concerned with the provision of several classes of service, with different patterns of demand and different resource requirements. To run a job of a given type, a virtual machine (VM) of that type is instantiated on one of the available servers. The resource capacity of a server is bounded, so that the ability of allocating a VM to it depends both on the type of the new job and on the numbers and types of the other jobs already running.

The quality of service offered to the different classes is regulated by a queueing policy based on preemptive priorities. There are costs associated, on the one hand, with holding jobs in the system, and on the other hand, with operating the servers. In the applications we have in mind, servers are hired from a cloud. Hence, the numbers involved are not considered to be large. They tend to be of the order of tens, rather than the thousands that are typically available in a service center. The number of job types is also, typically, not large. In the Amazon cloud, for example, the M3 family of VM instances has four types: medium, large, extra-large and extra-extra-large (https://aws.amazon.com/ec2/instance-types/).

In order to evaluate the server allocation trade-offs, it is necessary to analyze a multi-class queueing model with multiple shared servers. The purpose of that analysis is to provide exact and approximate procedures for computing the cost function for a given set of parameters. Those procedures would then be used to search for the optimal number of servers, whenever an allocation decision is to be made.

 0269-9648/17 $25.00

The assumption is that the traffic parameters (arrival and service rates) remain fixed for periods that are long enough for the system to be treated as having reached steady state. In practice, the resource provisioning policy would have to be supplemented by some monitoring and parameter estimation mechanism that would estimate those parameters and detect when they change. At the beginning of each such period, one would decide how many servers to allocate by applying a solution algorithm and searching for the optimum.

A model with two job types, formulated as a two-dimensional Markov process infinite in both directions, is solved exactly using a combination of one-dimensional and two-dimensional generating functions. The analysis also suggests an accurate approximation which is less complex, more numerically stable and more scalable than the exact solution. The solutions are then generalized recursively to more than two job types, by means of aggregations based not on total offered load but on previously computed average queue sizes.

The novel feature of our models is that, because VMs with different resource requirements share a server, an incoming higher priority job may interrupt and displace more than one lower priority jobs simultaneously. Similarly, the completion of a higher priority job may allow several lower priority jobs to enter service simultaneously. Therefore, although the methodology of transforming balance equations into generating functions has been applied before, the present analysis is different. The approximate solution and the generalization to more than two job types are also new.

The special case when each server can accommodate only one VM at a time, that is, the multi-class M/M/n preemptive priority queue, has been studied quite extensively. Harchol-Balter et al. [6] used phase-type distributions to approximate various busy periods and recursively reduce the dimensionality of the model to one. The resulting QBD process is solved by matrix-analytic methods. The numerical complexity of that approach increases quite rapidly with the number of servers, the number of job types and the number of phases in the PH distribution. In Mitrani and King [12] and Gail, Hantler and Taylor [5], the two-class case was solved by means of generating functions. A similar approach was taken by Kao and Narayanan [8]. A variant of the preemptive-priority policy involving queue size thresholds was examined by Feng, Kawada, and Adach [3].

The multi-class M/M/n model with non-preemptive priorities is also interesting. The two-class case was analyzed by Gail, Hantler, and Taylor [4] and Kao and Wilson [7], and (in the threshold variant) by Feng, Kawada, and Adach [2]. Kella and Yechiali [10] considered the special case where the average service times for all job types are the same. Some of those approaches could possibly be adapted to our system, assuming that VMs are allocated to servers according to a non-preemptive priority policy. However, that analysis would be rather complex and should be deferred to a separate undertaking.

Other, more distantly related works have sought to circumvent the difficulties of multi-dimensional processes by restricting the state space, so that only a finite number of jobs of certain types are allowed in the system. This was done by Kao and Narayanan [8,9] for both the preemptive and non-preemptive cases.

The optimization problem with respect to the number of servers has not received much attention, except in comparing the performance of the n-server system against that of a single server with an equivalent total service capacity (Harchol-Balter et al. [6], Wierman et al. [14]). The maximization of profit in a multi-class system where jobs that cannot enter service on arrival are rejected, was examined by Ezhilchelvan and Mitrani [1].

The exact analysis of the two-class model is presented in Section 2, while Section 3 describes the approximations and generalizations to more than two classes. Section 4 contains some numerical and simulation results aimed at evaluating the accuracy of the approximations.

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

2. TWO JOB TYPES: EXACT RESULTS

We start by assuming that a server may be shared by VMs of two different types, 1 and 2, and that type 1 jobs have preemptive-resume priority over type 2. Instantiations, interruptions and resumptions of service are instantaneous (in practice, of course, they take some time, but those overheads are assumed sufficiently small to be neglected; see Voorsluys et al. [13]). Jobs of types 1 and 2 arrive in independent Poisson streams with rates λ_1 and λ_2, respectively; their service times are independent random variables distributed exponentially with means $1/\mu_1$ and $1/\mu_2$, respectively.

The computational resources of a server may be expressed in terms of "virtual processors" (or vCPU), of which the total number available is V. The resource requirements of the two job types are v_1 and v_2, respectively. Hence, i jobs of type 1 and j jobs of type 2 can share the server without interference, provided that $iv_1 + jv_2 \leq V$. Typically $v_1 > v_2$, since higher priority is usually given to the jobs that bring in bigger revenues, and those tend to be the more demanding ones.

Thus, one type 1 job uses as much resource as k type 2 jobs, where $k = v_1/v_2$. Assume, to begin with, that that number is an integer. The maximum possible number of type 1 jobs in service is $s_1 = V/v_1$, also assumed to be an integer. The corresponding number for type 2 is $s_2 = s_1 k$. In the Amazon example mentioned in the "Introduction", if the two job types are, say, "extra-extra-large" and "large", then their requirements are $v_1 = 8$ and $v_2 = 2$, respectively. Hence, for a server with 16 vCPU, $s_1 = 2$, $k = 4$ and $s_2 = 8$. That is, such a server can be shared by a maximum of two type 1 and zero type 2 jobs, or one type 1 and four type 2 jobs, or zero type 1 and eight type 2 jobs.

In a system containing n-identical servers, the total amount of vCPU available is nV and the maximum number of type 1 jobs in service is $m = ns_1$. Moreover, if i type 1 jobs are being served, then the maximum number of type 2 jobs in service is $k(m - i)$. Any job that cannot be admitted into service joins an unbounded queue of its type.

The two numbers, m and k, together with the job arrival rates and average service times, are the parameters of our model. Note that in this formulation, the nature of the physical resource requirements, be they processors, memory, bandwidth, etc., is not important. It is enough to know the bound m on the number of type 1 jobs in service and the equivalence of 1-to-k between the two types. These two quantities will be referred to as the "type 1 service capacity" and the "type 2 equivalence", respectively. The product mk is the type 2 service capacity.

Suppose that each type 1 (type 2) job incurs a holding cost of c_1 (c_2) per unit time spent in the system. Each server incurs cost c_3 per unit time. The total long-term average cost incurred per unit time in an n-server system is then

$$C = c_1 L_1 + c_2 L_2 + c_3 n, \tag{1}$$

where L_1 and L_2 are the steady-state average numbers of type 1 and 2 jobs present in the system. The objective of the analysis is to provide algorithms for computing the right-hand side of (1), so that C can be minimized with respect to the number of servers, n.

We have no formal proof that the cost function (1) has a single minimum in terms of n, but that is invariably observed to be the case. Intuitively, if the cost of adding an extra server exceeds the benefit derived from it, then adding even more servers is not going to help. Therefore, the search for the best n can stop as soon as C starts increasing.

Since type 1 jobs have preemptive priority, they are not affected in any way by the existence of type 2. When there are i type 1 jobs present, their instantaneous completion rate is $i\mu_1$ if $i < m$ and $m\mu_1$ if $i \geq m$. Hence, the type 1 queue behaves like an M/M/m queue

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

with offered load $\rho_1 = \lambda_1/\mu_1$; steady-state exists when $\rho_1 < m$ and there is a well-known closed-form expression for L_1 (see, e.g., Mitrani [11]).

The difficulty is in computing L_2. For that, it is necessary to consider the joint equilibrium distribution of the type 1 and 2 jobs in the system. Intuitively, both queues are stable if the total offered load, expressed in terms of type 2 job equivalents, is lower than the type 2 service capacity:

$$\rho_1 k + \rho_2 < mk, \tag{2}$$

where $\rho_2 = \lambda_2/\mu_2$. This inequality implies $\rho_1 < m$, which is the stability condition for an isolated queue 1.

We shall establish later that (2) is indeed the ergodicity condition for the two-dimensional Markov process.

Denote by $p_{i,j}$ the steady-state probability that there are i type 1 and j type 2 jobs present $(i, j = 0, 1, \ldots)$. When $i < m$, these probabilities satisfy the following balance equations:

$$\begin{aligned}[\lambda_1 + \lambda_2 + i\mu_1 + \mu_2(i,j)]p_{i,j} = {} & \lambda_1 p_{i-1,j} + \lambda_2 p_{i,j-1} + (i+1)\mu_1 p_{i+1,j} \\ & + \mu_2(i,j+1)p_{i,j+1}; \quad j = 0, 1, \ldots,\end{aligned} \tag{3}$$

where a probability with a negative index is 0 by definition, and

$$\mu_2(i,j) = \begin{cases} j\mu_2 & \text{if } j < (m-i)k \\ (m-i)k\mu_2 & \text{if } j \geq (m-i)k. \end{cases} \tag{4}$$

When $i \geq m$, the servers are fully occupied by type 1 jobs and the service rate for type 2 is 0. The balance equations become

$$[\lambda_1 + \lambda_2 + m\mu_1]p_{i,j} = \lambda_1 p_{i-1,j} + \lambda_2 p_{i,j-1} + m\mu_1 p_{i+1,j}; \quad j = 0, 1, \ldots. \tag{5}$$

To determine $p_{i,j}$, introduce the generating functions

$$g_i(z) = \sum_{j=0}^{\infty} p_{i,j} z^j; \quad i = 0, 1, \ldots. \tag{6}$$

We shall also need the bi-variate generating function corresponding to the states where $i \geq m$:

$$g(y,z) = \sum_{i=m}^{\infty} g_i(z) y^{i-m}. \tag{7}$$

Consider first the region $i \geq m$. Multiplying (5) by z^j and summing over all j, those equations are transformed into

$$[\lambda_1 + \lambda_2(1-z) + m\mu_1]g_i(z) = \lambda_1 g_{i-1}(z) + m\mu_1 g_{i+1}(z); \quad i = m, m+1, \ldots. \tag{8}$$

Now, multiplying (8) by y^{i-m} and summing over $i \geq m$, yields after a little manipulation,

$$a(y,z)g(y,z) = \lambda_1 y g_{m-1}(z) - m\mu_1 g_m(z) \tag{9}$$

where

$$a(y,z) = \lambda_1 y(1-y) + \lambda_2 y(1-z) + m\mu_1(y-1). \tag{10}$$

Note that, for every z in the interval [0,1), the quadratic $a(y,z)$ is negative at $y = 0$, positive at $y = 1$ and negative at $y = \infty$. Therefore, for each such value of z, it has exactly

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

two real zeros, $0 < y_1(z) < 1$ and $1 < y_2(z)$. The smaller of these is given by

$$y_1(z) = \frac{d - \sqrt{d^2 - 4\lambda_1 m\mu_1}}{2\lambda_1}, \tag{11}$$

where $d = \lambda_1 + \lambda_2(1 - z) + m\mu_1$. At $z = 1$, $y_1(1) = 1$.

Since $g(y, z)$ is the generating function of part of a stationary distribution, it is finite at $y = z = 1$. Therefore, the right-hand side of (9) must vanish at points $[z, y_1(z)]$, for all $0 \le z \le 1$. This gives a relation between $g_m(z)$ and $g_{m-1}(z)$:

$$m\mu_1 g_m(z) = \lambda_1 y_1(z) g_{m-1}(z). \tag{12}$$

Substituting (12) back into (9), and remembering that $a(y, z)$ can be written as $\lambda_1(y - y_1(z))(y_2(z) - y)$, we obtain a simple expression for $g(y, z)$ in terms of $g_{m-1}(z)$:

$$g(y, z) = \frac{g_{m-1}(z)}{y_2(z) - y}. \tag{13}$$

When $i < m$, there is service capacity available to type 2 jobs. Moreover, the dependency of $\mu_2(i, j)$ on j, in states where $j < (m - i)k$, means that the probabilities of those states remain explicit after the balance equations are transformed using the generating functions $g_i(z)$. For each $i = 0, 1, \ldots, m - 1$, multiply (3) by z^j and sum over all $j \ge 0$. This yields the following set of equations:

$$a_i(z)g_i(z) = \lambda_1 z g_{i-1}(z) + (i + 1)\mu_1 z g_{i+1}(z) + q_i(z); \quad i = 0, 1, \ldots, m - 1, \tag{14}$$

where $g_{-1}(z) = 0$ by definition,

$$a_i(z) = (\lambda_1 + i\mu_1)z + \lambda_2 z(1 - z) + (m - i)k\mu_2(z - 1); \quad i = 0, 1, \ldots, m - 1, \tag{15}$$

and

$$q_i(z) = \mu_2(z - 1) \sum_{j=0}^{(m-i)k-1} [(m - i)k - j]p_{i,j} z^j. \tag{16}$$

In the last of Eq. (14), when $i = m - 1$, $g_m(z)$ can be eliminated by means of (12), leading to

$$b_{m-1}(z)g_{m-1}(z) = \lambda_1 z g_{m-2}(z) + q_{m-1}(z), \tag{17}$$

where

$$b_{m-1}(z) = \lambda_1 z(1 - y_1(z)) + (m - 1)\mu_1 z + \lambda_2 z(1 - z) + k\mu_2(z - 1), \tag{18}$$

with $y_1(z)$ given by (11).

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

This set of simultaneous equations can be written in matrix and vector form as

$$A(z)\mathbf{g}(z) = \mathbf{q}(z), \tag{19}$$

where $\mathbf{g}(z) = [g_0(z), g_1(z), \ldots, g_{m-1}(z)]$, $\mathbf{q}(z) = [q_0(z), q_1(z), \ldots, q_{m-1}(z)]$ (both are column vectors), and $A(z)$ is the tri-diagonal matrix

$$A(z) = \begin{bmatrix} a_0(z) & -\mu_1 z & & & & \\ -\lambda_1 z & a_1(z) & -2\mu_1 z & & & \\ & -\lambda_1 z & a_2(z) & -3\mu_1 z & & \\ & & & \ddots & & \\ & & & -\lambda_1 z & a_{m-2}(z) & -(m-1)\mu_1 z \\ & & & & -\lambda_1 z & b_{m-1}(z) \end{bmatrix}.$$

The solution of (19) is given by

$$g_i(z) = \frac{D_i(z)}{D(z)}; \quad i = 0, 1, \ldots, m-1, \tag{20}$$

where $D(z)$ is the determinant of $A(z)$ and $D_i(z)$ is the determinant of the matrix obtained from $A(z)$ by replacing its $(i+1)$st column with the column vector $\mathbf{q}(z)$.

In the right-hand side of (20), there are $km(m+1)/2$ unknown constants, the probabilities $p_{i,j}$ for $i = 0, 1, \ldots, m-1$ and $j = 0, 1, \ldots, k(m-i)-1$. However, in order to determine them, it is necessary to include among the unknowns all probabilities $p_{i,j}$ for $i = 0, 1, \ldots, m$ and $j = 0, 1, \ldots, km-i-1$. This is a larger set when $k > 1$. It contains $(2k-1)m(m+1)/2$ unknowns.

The $(2km-m-1)m/2$ balance Eq. (3), for $i = 0, 1, \ldots, m-1$ and $j = 0, 1, \ldots, km-i-2$ involve these unknowns. In addition, relation (12) between $g_m(z)$ and $g_{m-1}(z)$ yields $(km-m)$ equations expressing $p_{m,j}$ $(j = 0, 1, \ldots, km-m-1)$ in terms of $p_{m-1,j}$. Specifically,

$$p_{m,j} = \frac{\lambda_1}{m\mu_1} \sum_{s=0}^{j} v_s p_{m-1,j-s}; \quad j = 0, 1, \ldots, km-m-1, \tag{21}$$

where $v_s = y_1^{(s)}(0)/s!$ are the Maclaurin expansion coefficients for $y_1(z)$. These are easily calculated by means of recurrences. Expression (11) gives

$$v_0 = \frac{u - \sqrt{u^2 - 4\lambda_1 m\mu_1}}{2\lambda_1},$$

with $u = \lambda_1 + \lambda_2 + m\mu_1$. Then, equating (10) to 0 and taking derivatives at $z = 0$, we obtain

$$v_j(u - 2\lambda_1 v_0) = \lambda_2 v_{j-1} + \lambda_1 \sum_{s=1}^{j-1} v_{j-s} v_s; \quad j \geq 1. \tag{22}$$

There is now a shortfall of m equations with respect to the number of unknowns. One more equation is provided by the normalization condition: all probabilities $p_{i,j}$ must add up to 1:

$$\sum_{i=0}^{m-1} g_i(1) + g(1,1) = 1. \tag{23}$$

An equivalent way of formulating this equation is by remembering that the marginal distribution, $p_{i,\cdot}$, of type 1 jobs in the system is that of an M/M/m queue. Hence, the value

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

of $g_0(1)$, say, is given by the known expression for the probability that the M/M/m queue is empty. Equating that expression to the right-hand side of (20), for $i = 0$ and $z = 1$, and performing manipulations in order to avoid an indeterminacy and to simplify the result, produces a rather intuitive equation for the steady-state average number of type 2 service vacancies, in terms of the two offered loads:

$$\sum_{i=0}^{m-1} \sum_{j=0}^{(m-i)k-1} ((m-i)k-j)p_{i,j} = mk - \rho_1 k - \rho_2. \tag{24}$$

The derivation of (24) is shown in more detail in the Appendix.

Equation (24) also establishes the necessity of condition (2) for the stability of the queueing process. Clearly, if a normalizable steady-state distribution $p_{i,j}$ exists, then the right-hand side of (24) must be positive.

The sufficiency of (2) is implied by the following result.

LEMMA 2.1: *If condition (2) holds, then $D(z)$ has exactly $m-1$ real and distinct zeros in the interval (0,1), in addition to the zero at $z = 1$.*

The proof of the lemma is in the Appendix.

Consider one of the generating functions, say $g_0(z)$. It is finite on the interval (0,1), so the numerator $D_0(z)$ must vanish at the $m-1$ zeros of $D(z)$ on that interval. Those $m-1$ equations, together with the others described earlier, provide a set of $(2k-1)m(m+1)/2$ equations, which determines the $(2k-1)m(m+1)/2$ unknown constants.

Using other generating functions would not provide new independent equations because of the relations between them.

Since (2) enables the determination of a normalizable stationary distribution, it is a sufficient condition for the stability of the process.

Having computed the unknown probabilities, the average number of type 2 jobs present in the system is obtained from

$$L_2 = \sum_{i=0}^{m-1} g_i'(1) + \frac{\partial}{\partial z} g(1,1). \tag{25}$$

The derivatives $g_i'(1)$ can be computed either by applying the rule for differentiating a determinant, or more simply by using the definition of a derivative:

$$g_i'(1) \approx \frac{g_i(1) - g_i(\delta)}{1-\delta}, \tag{26}$$

for some value of δ suitably close to 1. The last term in the right-hand side of (25) involves the larger zero, $y_2(z)$, of $a(y,z)$, defined by (10). That function can be shown to satisfy

$$y_2(1) = \frac{m\mu_1}{\lambda_1}; \quad y_2'(1) = -\frac{\lambda_2 m\mu_1}{\lambda_1(m\mu_1 - \lambda_1)}. \tag{27}$$

The cost function (1) can now be evaluated for different values of n, in order to find the optimal number of servers to hire.

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

3. APPROXIMATIONS AND GENERALIZATIONS

The bulk of the computational effort expended in the evaluation of (1) goes into finding the $m-1$ zeros of $D(z)$ in the interval (0,1), and solving the set of $(2k-1)m(m+1)/2$ linear equations. That effort grows quite steeply with m and k; its numerical complexity is on the order of $O(k^3m^6)$. Moreover, the exact solution may have problems with instability. As m increases, some of the zeros of $D(z)$ tend to bunch together, causing the matrix of the set of linear equations to become ill-conditioned. It is therefore highly desirable to develop an approximate solution that is faster and more stable. This is what we now propose.

The idea is to make an accurate guess for the probabilities that appear in the right-hand side of (20). We shall assume that the values of $p_{i,j}$, for $i \leq m-1$ and $j \leq (m-i)k-1$, have the form

$$p_{i,j} = H\frac{\rho_1^i \rho_2^j}{i!j!}, \tag{28}$$

where H is an appropriately selected constant. These expressions satisfy some of the balance equations for $p_{i,j}$, namely all those where both $p_{i+1,j}$ and $p_{i,j+1}$ belong to the set of unknowns.

The normalization constant H is chosen so that (24) is satisfied:

$$H = (mk - \rho_1 k - \rho_2)\left[\sum_{i=0}^{m-1}\sum_{j=0}^{(m-i)k-1}((m-i)k-j)\frac{\rho_1^i\rho_2^j}{i!j!}\right]^{-1}. \tag{29}$$

From this point on, the computation proceeds as in the exact solution. Equations (20) and (26) are used to compute $g_i'(1)$, and (25) provides the value of L_2. It turns out that the choice of estimates (28), together with the exact normalization (29), produces a very good approximation. Its accuracy will be illustrated in Section 4.

3.1. More than Two Job Types

Consider now an n-server system with three preemptive priority job types and unbounded queues, under Markovian assumptions. The arrival rates are λ_1, λ_2, λ_3, and the average service times are $1/\mu_1$, $1/\mu_2$, $1/\mu_3$, respectively.

As before, assume that each server can accommodate a maximum of s_1 jobs of type 1, that is, up to $m = s_1 n$ such jobs can be in service at any one time. Each type 1 job is equivalent, in terms of resources required, to k_1 jobs of type 2, and each type 2 job is equivalent to k_2 jobs of type 3. Hence, the product mk_1k_2 is the service capacity available to type 3.

All three queues are stable when the total offered load, expressed in terms of type 3 job equivalents, is lower than the type 3 service capacity:

$$\rho_1 k_1 k_2 + \rho_2 k_2 + \rho_3 < mk_1k_2. \tag{30}$$

This condition implies the stability of queue 1 isolated from queues 2 and 3 ($\rho_1 < m$), and also of queues 1 and 2 isolated from queue 3 ($\rho_1 k_1 + \rho_2 < mk_1$).

The exact solution of the above model is currently intractable. There is no known methodology for tackling a three-dimensional Markov process where all three dimensions are infinite. We therefore propose an approximation based on aggregating types 1 and 2 into a single higher priority job type, called type h, with exponentially distributed service times. One would then apply either the exact or the approximate solution to the system consisting of the two priority types, h and 3.

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

The arrival rate for jobs of type h is clearly equal to $\lambda_h = \lambda_1 + \lambda_2$. However, it is far from obvious how best to set the other parameters associated with type h: the average service time, $1/\mu_h$, the type h service capacity, m_h, and the type 3 equivalence, k_h, of one type h job.

These parameters should, ideally, satisfy the following conditions:

(a) $m_h k_h = m k_1 k_2$. The type 3 service capacity can be expressed in terms of either equivalent type h jobs or equivalent type 1 and 2 jobs.

(b) $\rho_h k_h = \rho_1 k_1 k_2 + \rho_2 k_2$ (where $\rho_h = \lambda_h/\mu_h$ is the offered load of type h). The type 3 service capacity occupied by higher priority jobs can be expressed in terms of either type h or types 1 and 2.

(c) The average number of jobs, L_h, in the M/M/m_h queue with offered load ρ_h, is equal to $L_1 + L_2$, where the latter is obtained by solving the model with the two priority types 1 and 2.

Unfortunately, these three conditions are not, in general, compatible. For example, if $k_1 = k_2 = 1$, then $k_h = 1$ and (b) suggests the standard aggregation $\rho_h = \rho_1 + \rho_2$. This leads to a value for L_h which tends to underestimate the known total $L_1 + L_2$ (because a hyperexponential service time distribution is replaced by an exponential one with a lower variance). Moreover, the requirement that both m_h and k_h should be integers is difficult to satisfy.

The proposed approximation lays emphasis on (c), using that condition to calibrate the behavior of type h. The reduction in the service time variance resulting from the aggregation of types 1 and 2 is compensated by an appropriate choice of the type h service capacity and offered load.

Eliminating k_h from (a) and (b) yields

$$m_h = \frac{m k_1}{\rho_1 k_1 + \rho_2} \rho_h. \tag{31}$$

Note that the value of of m_h obtained from (31) is always larger than ρ_h, because of (30).

Find ρ_h, and m^* such that (a) m^* is given by the right-hand side of (31), and (b) the average number of jobs in the M/M/m^* queue with offered load ρ_h is equal to $L_1 + L_2$. Note that this search does not require integer values for m^*. The M/M/m^* queue is simply a Birth-and-Death process where the instantaneous departure rate in state j is $j\mu$ if $j < m^*$ and $m^*\mu$ if $j \geq m^*$.

The service capacity of type h is chosen as the integer part of m^*: $m_h = \lfloor m^* \rfloor$. The offered load ρ_h gives the service rate parameter for type h, $\mu_h = \lambda_h/\rho_h$, while the equivalence parameter k_h is set according to (a):

$$k_h = \frac{m k_1 k_2}{m_h}. \tag{32}$$

This is not necessarily an integer. However, a review of the two-queue solution in the previous section shows that it does not have to be. The definition of $\mu_2(i,j)$ in (4) holds for non-integer values of k, as do those of $a_i(z)$ and $b_{m-1}(z)$ in (15) and (18), respectively. The unknown probabilities that appear in (16) are those for which $j < \lceil (m-i)k \rceil$, where $\lceil x \rceil$ is the lowest integer greater than or equal to x. The application of the approximate solution based on (28) is equally straightforward.

Thus, the approximate solution of the three-queue model proceeds in two steps. First, the model consisting of types 1 and 2 is solved, in order to compute L_1 and L_2. These two

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

types are then aggregated into type h, as described above, and the model consisting of types h and 3 is solved in order to compute L_3.

More generally, if there are T job types in the system, with type 1 service capacity m and each type t job equivalent to k_t jobs of type $t+1$ ($t = 1, 2, \ldots, T-1$), then the condition for stability is

$$\sum_{t=1}^{T} \rho_t K_t < mK_1, \tag{33}$$

where $K_t = k_t k_{t+1}, \ldots, k_{T-1}$ is the equivalence of a type t job in terms of type T jobs; $K_T = 1$.

The approximate solution has $T-1$ steps. Step 1 yields L_1 and L_2. Step 2 aggregates types 1 and 2 and computes L_3, as described above. In step t ($t \geq 3$), the previously aggregated job types $1, 2, \ldots, t-1$ represent type 1, type t is type 2 and type $t+1$ is type 3. The above procedure then produces a single high priority type, h, comprising types $1, 2, \ldots, t$. The 2-type solution is applied to types h and $t+1$, in order to compute L_{t+1}. In the last step, $T-1$, L_T is computed after aggregating all the other job types $1, 2, \ldots, T-1$ into a single type.

4. ACCURACY OF THE APPROXIMATIONS

Our purpose in this section is to evaluate, numerically or by simulation, the accuracy of the various approximate solutions that have been proposed.

The first experiment compares the exact and approximate solutions of the 2-type model, for increasing offered load of type 2. In this example, the type 2 equivalence is $k = 4$, that is, a type 1 job requires as much resource as four jobs of type 2. The maximum number of type 1 jobs that can be in service is $m = 6$. This could represent a six-server system with $s_1 = 1$, or a three-server system with $s_1 = 2$, or a two-server system with $s_1 = 3$, or a one-server system with $s_1 = 6$. In all cases, the total type 2 service capacity is 24.

The fixed traffic parameters are $\lambda_1 = 3$, $\mu_1 = 1$ and $\mu_2 = 2$. The system is stable as long as $\rho_2 < 24 - 4\rho_1$, or $\lambda_2 < 24$. The exact and approximate values of L_2 are computed for 11 different type 2 arrival rates, ranging from $\lambda_2 = 2$ to $\lambda_2 = 22$. That is, the total utilization of the system ranges from just over 50% to more than 90%.

The results are illustrated in Figure 1. It is notable that both the absolute and the relative errors of the approximation are very small. That is true over the entire range of offered loads. In fact, the relative errors decrease, from about 5% at the lower loads, to <2% at the higher ones.

It can also be observed that the approximate solution is slightly pessimistic: it consistently overestimates the average type 2 queue sizes, although by small amounts. We have no intuitive explanation for this. It is somehow a consequence of the expressions (28) for the probabilities $p_{i,j}$.

The next example compares the exact and approximate values of the costs incurred when the number of servers increases. The parameters are as in Figure 1, except that the type 2 arrival rate is now fixed at $\lambda_2 = 4$, and $s_1 = 1$. That is, the maximum number of type 1 jobs in service is equal to the number of servers, $m = n$. The type 1 holding costs are twice as large as those for type 2: $c_1 = 2$, $c_2 = 1$. The cost of a server is $c_3 = 1$.

The number of servers in Figure 2 varies between $n = 4$, which is the smallest number that can cope with the offered load of $4\rho_1 + \rho_2$, and $n = 9$. At the start of that range the total cost incurred is high because the value of L_2 is large and the holding cost dominates. For large n, the increasing server cost dominates. The approximate costs are

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

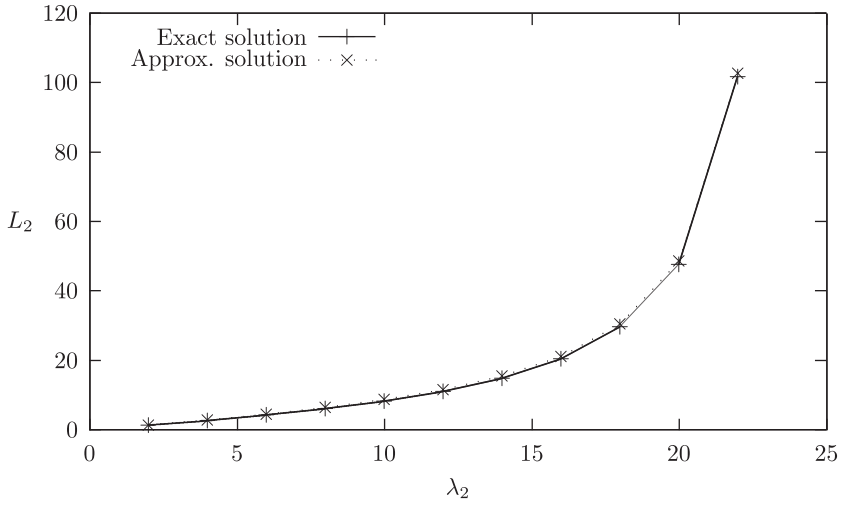

FIGURE **1.** Two job types; increasing λ_2. $\lambda_1 = 3$, $\mu_1 = 1$, $\mu_2 = 2$, $m = 6$, $k = 4$.

almost indistinguishable from the exact ones and correctly predict the optimal number of servers, $n = 6$.

It is worth pointing out the limitations of the exact solution with respect to the value of m. In the above example, if the system with $\lambda_2 = 4$ and $m = 6$ is scaled up by a factor of 5, to $\lambda_2 = 20$ and $m = 30$, the solution still works. However, if it is scaled up by a factor of 10, to $\lambda_2 = 40$ and $m = 60$, then the computation of the unknown probabilities fails. The matrix of the set of simultaneous linear equations, whose dimensions are then 5490×5490, becomes ill-conditioned. It is for that reason that the approximate solution is important. It copes easily with the scaled system.

Next, we examine systems with three job types. Since there is no exact solution for these models, the approximation proposed in the previous section will be compared against simulations. Moreover, when evaluating the intermediate results for two job types that form

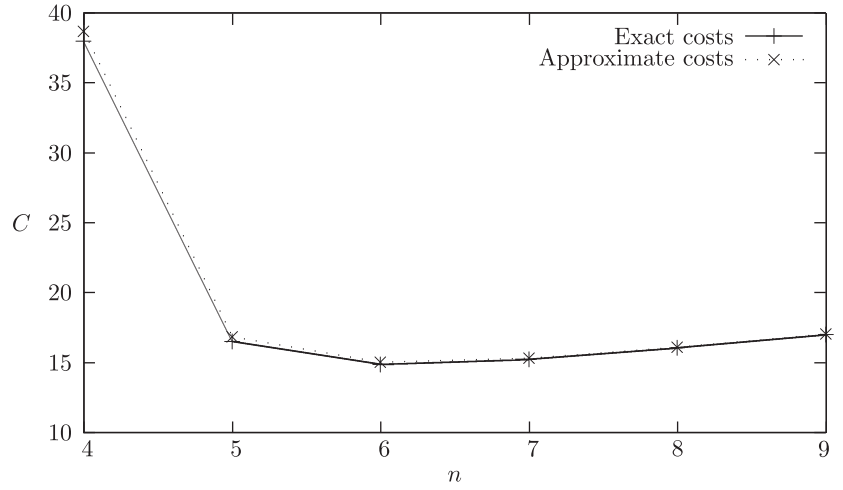

FIGURE **2.** Two job types; increasing number of servers. $\lambda_1 = 3$, $\lambda_2 = 4$, $\mu_1 = 1$, $\mu_2 = 2$, $m = n$, $k = 4$, $c = (2, 1, 1)$.

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

part of the three-type computation, we apply the approximate two-type solution. The exact solution would make very little difference.

Again we use examples where a job of type 1 requires as much resource as four jobs of type 2 ($k_1 = 4$), but now a job of type 2 is equivalent to two jobs of type 3 ($k_2 = 2$). Thus, if the offered loads of types 1, 2, and 3 are ρ_1, ρ_2, and ρ_3, respectively, then the total type three-equivalent offered load is $8\rho_1 + 2\rho_2 + \rho_3$. For stability, that should be lower than the type 3 service capacity, $mk_1k_2 = 8m$.

Figure 3 compares the estimated and simulated values of the average type 3 queue size, L_3, for increasing type 3 arrival rates. The other traffic parameters are: $\lambda_1 = 3$, $\lambda_2 = 6$, $\mu_1 = 1$, $\mu_2 = 2$, $\mu_3 = 3$. The maximum number of type 1 jobs in service is $m = 6$. That could represent, among others, a six-server system where $s_1 = 1$. Consequently, the type 3 service capacity is 48, and types 1 and 2 utilize more than 60% of it. For the chosen values of λ_3, the total system utilization ranges between 65 and 85%.

Each simulated point corresponds to a run where about a million jobs of all types go through the system. That run is divided into 10 portions of equal size for the purpose of obtaining a sample of observations and computing 95% confidence intervals.

The figure shows a close agreement between approximation and simulation. Indeed, all approximated points lie within the confidence intervals of the corresponding simulated ones. The slight underestimation of L_3 by the model at low to medium values of λ_3 is probably due to the fact that an aggregation of job types reduces the variability of the process. In fact, we have observed in other examples that the underestimation becomes more pronounced when the higher priority job types, 1 and 2, consume a bigger fraction of the type 3 service capacity. There is no obvious explanation for the apparent crossover into an overestimation at high values of λ_3; this may be due to the particular way of constructing the aggregation.

Figure 4 compares the estimated and simulated costs of the three-type system for increasing number of servers. There is a maximum of one type 1 job per server, that is, $m = n$. The equivalence numbers are again $k_1 = 4$, $k_2 = 2$. The three arrival rates are $\lambda_1 = 2$, $\lambda_2 = 6$ and $\lambda_3 = 12$; the service rates are kept as before, $\mu_1 = 1$, $\mu_2 = 2$ and $\mu_3 = 3$. Hence, the smallest number of servers that can cope with the offered load of $8\rho_1 + 2\rho_2 + \rho_3$ is $n = 4$; the utilization of those four servers would then be more than 81%, and about 70%

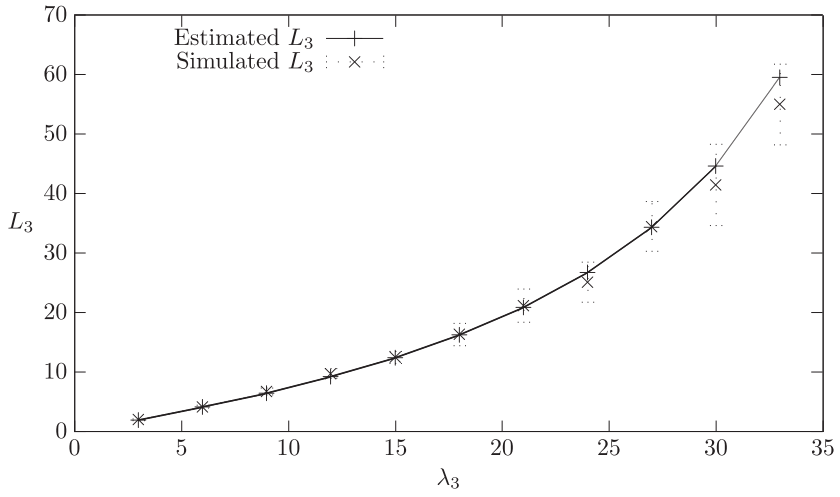

FIGURE **3.** Three job types; increasing λ_3. $(\lambda_1, \lambda_2) = (3, 6)$, $\mu = (1, 2, 3)$, $(m, k_1, k_2) = (6, 4, 2)$.

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

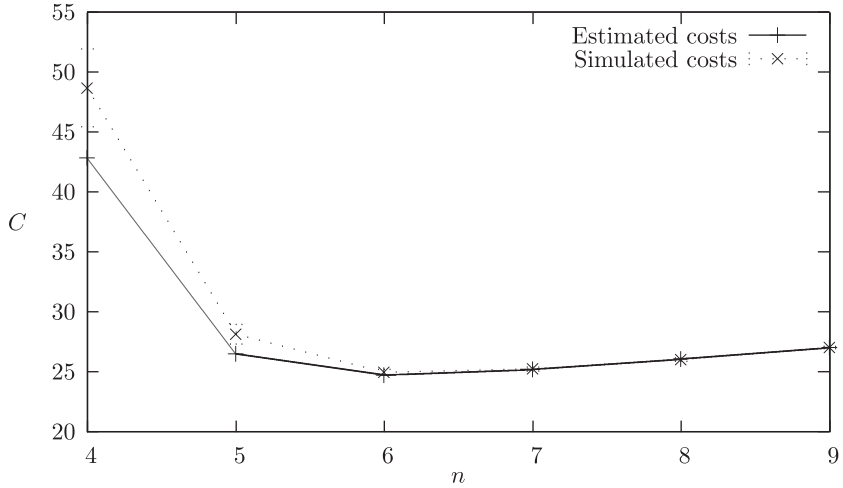

FIGURE **4.** Estimated and simulated costs; increasing n. $\lambda = (2, 6, 12)$, $\mu = (1, 2, 3)$, $(m, k_1, k_2) = [n, 4, 2]$, $c = [4, 2, 1, 1]$.

would be used by types 1 and 2. The holding costs are $c_1 = 4$, $c_2 = 2$, and $c_3 = 1$, while the server cost is $c_4 = 1$.

The results of the approximation are now less accurate at the heavily loaded end of the range. The first two estimated points are outside the 95% confidence intervals of the corresponding simulated ones. However, the agreement between the two plots is still sufficiently close for the optimal number of servers to be correctly predicted by the approximation.

The last experiment concerns a system with four job types. In Figure 5, the approximated values of L_4 are compared with the simulated ones, for increasing type 4 arrival rates. The system parameters are specified in the caption. In this example, the type 4 service capacity is 144, of which about 68% is consumed by types 1, 2, and 3. The type 4 traffic makes the total utilization range from 69 to 78%.

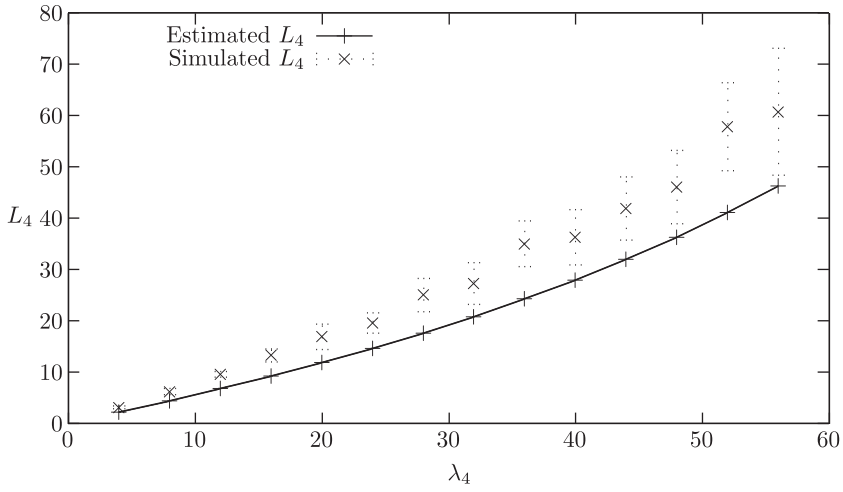

FIGURE **5.** Four job types; increasing λ_4. $(\lambda_1, \lambda_2, \lambda_3) = (3, 6, 12)$, $\mu = (1, 2, 3, 4)$, $(m, k_1, k_2, k_3) = [6, 4, 3, 2]$.

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

We observe that the model consistently underestimates the values of L_4. As the load increases, the variability of the process increases considerably, as shown by the large confidence intervals. Much of that variability is eliminated by the aggregation procedure. Hence, the accuracy of the approximation has decreased. The relative errors are now on the order of 10%. The absolute ones increase with the load.

It is clear that, in heavily loaded systems, the quality of the approximation would deteriorate further with the addition of more job classes. However, at light to moderate loads, the approximations we have proposed are adequate.

5. CONCLUSION

The contribution of this paper is threefold: first, it presents an exact solution for a model where several servers are shared unequally between two preemptive priority job types with different resource requirements. Second, an efficient and accurate approximation for the same model is developed. This eliminates certain defects of the exact solution, such as high numerical complexity and possible instability. Third, the generalization to more than two job types is handled by an approximate solution, which relies on a non-standard aggregation of higher priority job types and consecutive applications of two-type solutions. That approximation produces accurate estimates, as long as the offered loads are not too high.

Both the exact and the approximate solutions could be easily adapted to models where higher priority jobs have lower resource requirements. That is, a type 2 job could be equivalent to k type 1 jobs. The equations for the generating functions $g_i(z)$ would then be different, but the solution methodology would still apply.

One of the assumptions of the model was that V/v_1 is an integer. Otherwise, there would still be some resource capacity available in the server when the maximum number, $s_1 = \lfloor V/v_1 \rfloor$, of type 1 jobs are being served ($\lfloor x \rfloor$ is the integer part of x). If that leftover capacity is not enough to run a type 2 job, then the analysis would remain the same. However, if there is a non-zero service rate for type 2 when the maximum number of type 1 jobs are in service, then the treatment of the bi-variate generating function would be considerably more complicated. That would be a topic for future research.

If, instead of a single resource type, there are R resource types, with a server capacity V_r and a type 1 requirement $v_{1,r}$ for resource of type r, then the maximum number of type 1 jobs that can share a server would be

$$s_1 = \min_r \left\{ \left\lfloor \frac{V_r}{v_{1,r}} \right\rfloor \right\},$$

and there would be an appropriate type 2 equivalence parameter k. As explained above, our analysis would remain valid provided that, when there are s_1 jobs of type 1 in the server, the type 2 service rate is 0.

Another worthy topic for further research would be to solve this kind of model with non-preemptive priorities. Again, the analysis would be more complicated because jobs of type 2 may be in service regardless of how many type 1 jobs are present. However, it may be possible to adapt the approach of Gail, Hantler, and Taylor [4], and to develop approximations of the sort presented here.

References

1. Ezhilchelvan, P. & Mitrani, I. (2017). Optimal provisioning of servers for hosting services of multiple types. *Simulation Modelling Practice and Theory* 75: 17–28.

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

2. Feng, W., Kawada, M., & Adachi, K. (2000). Analysis of a multi-server queue with two priority classes and (M,N)-threshold service schedule I: Non-preemptive priority. *International Trans. in Operational Research* 7: 653–671.
3. Feng, W., Kawada, M., & Adachi, K. (2001). Analysis of a multiserver queue with two priority classes and (M,N)-threshold service schedule. II. preemptive priority. *Asia-Pacific Journal of Operations Research* 18: 101–124.
4. Gail, H., Hantler, S., & Taylor, B. (1988). Analysis of a non-preemptive priority multiserver queue. *Advances in Applied Probability* 20: 852–879.
5. Gail, H., Hantler, S., & Taylor, B. (1992). On a preemptive Markovian queues with multiple servers and two priority classes. *Mathematics of Operations Research* 17: 365–391.
6. Harchol-Balter, M., Osogami, T., Scheller-Wolf, A., & Wierman, A. (2005). Multi-server queueing systems with multiple priority classes. *Queueing Systems Theory and Applications* 51(3): 331–360.
7. Kao, E. & Wilson, S. (1999). Analysis of nonpreemptive priority queues with multiple servers and two priority classes. *European Journal of Operational Research* 118: 181–193.
8. Kao, E. & Narayanan, K. (1991). Modeling a multiprocessor system with preemptive priorities. *Management Science* 2: 185–197.
9. Kao, E. & Narayanan, K.S. (1990). Computing steady-state probabilities of a nonpreeptive priority multiserver queue. *Journal on Computing* 2(3): 211–218.
10. Kella, O. & Yechiali, U. (1985). Waiting times in the non-preemptive priority M/M/c queue. *Stochastic Models* 1: 257–262.
11. Mitrani, I. (1998). *Probabilistic modelling*. Cambridge, UK: Cambridge University Press.
12. Mitrani, I. & King, P. (1981). Multiprocessor systems with preemptive priorities. *Performance Evaluation* 1: 118–125.
13. Voorsluys, W., Broberg, J., Venugopal, S., & Buyya, R. (2009). Cost of virtual machine live migration in clouds: A performance evaluation. *LNCS* 5931: 254–265.
14. Wierman, A., Osogami, T., Harchol-Balter, M., & Scheller-Wolf, A. (2006). How many servers are best in a dual-priority M / PH / k system?. *Performance Evaluation* 63(12): 1253–1272.

APPENDIX

Appendix A: Derivation of Eq. (24)

Consider first the numerator in the right-hand side of (20), for $i = 0$. According to (16), we can write

$$D_0(z) = \mu_2(z-1)d_0(z), \tag{A.1}$$

where $d_0(z)$ is obtained from $D_0(z)$ by replacing the first column vector, $\mathbf{q}(z)$, with the vector whose elements are

$$s_i(z) = \sum_{j=0}^{(m-i)k-1} [(m-i)k-j]p_{i,j}z^j.$$

Adding all rows of $d_0(z)$ to the last row and setting $z = 1$ yields

$$d_0(1) = \begin{vmatrix} s_0(1) & -\mu_1 & & & & & \\ s_1(1) & a_1(1) & -2\mu_1 & & & & \\ s_2(1) & -\lambda_1 & a_2(1) & -3\mu_1 & & & \\ & & & \ddots & & & \\ s_{m-2}(1) & & & -\lambda_1 & a_{m-2}(1) & -(m-1)\mu_1 \\ s & 0 & 0 & \cdots & 0 & 0 \end{vmatrix}, \tag{A.2}$$

where $|M|$ is the determinant of matrix M and s is the left-hand side of (24):

$$s = \sum_{i=0}^{m-1} s_i(1) = \sum_{i=0}^{m-1} \sum_{j=0}^{(m-i)k-1} [(m-i)k-j]p_{i,j}.$$

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

Since the last row of $D_0(1)$ contains a single non-zero element and the corresponding minor is triangular, we have

$$d_0(1) = s\mu_1^{m-1}(m-1)!. \tag{A.3}$$

The denominator in the right-hand side of (20) also has $\mu_2(z-1)$ as a factor and can be written as

$$D(z) = \mu_2(z-1)d(z). \tag{A.4}$$

This can be seen by adding all the rows of $D(z)$ to the last one. The determinant $d(z)$ has the form

$$d(z) = \begin{vmatrix} a_0(z) & -\mu_1 z & & & & & \\ -\lambda_1 z & a_1(z) & -2\mu_1 z & & & & \\ & -\lambda_1 z & a_2(z) & -3\mu_1 z & & & \\ & & & \ddots & & & \\ & & & -\lambda_1 z & a_{m-2}(z) & -(m-1)\mu_1 z \\ b_0(z) & b_1(z) & b_2(z) & \cdots & b_{m-2}(z) & b(z) \end{vmatrix}, \tag{A.5}$$

where

$$b_i(z) = (m-i)k - \rho_2 z; \quad i = 0, 1, \ldots, m-2,$$

and

$$b(z) = \frac{mk - k\rho_1 - m\rho_2 z - \rho_1\rho_2 z[1 - y_1(z)]}{m - \rho_1}.$$

In writing this last expression, we have used the fact that $y_1(z)$ satisfies

$$a(y_1(z), z) = 0,$$

with $a(y, z)$ given by (10).

Setting $z = 1$ in (A.5) and expanding the determinant along the elements of its last row, we obtain, after some work,

$$d(1) = a\mu_1^{m-1}(m-1)! \left[\sum_{j=0}^{m-1} \frac{\rho_1^j}{j!} + \frac{m\rho_1^m}{(m-\rho_1)m!} \right], \tag{A.6}$$

where $a = mk - k\rho_1 - \rho_2$ is the right-hand side of (24).

Thus, the marginal probability that the type 1 queue is empty is given by

$$p_{0,\cdot} = \frac{s}{a} \left[\sum_{j=0}^{m-1} \frac{\rho_1^j}{j!} + \frac{m\rho_1^m}{(m-\rho_1)m!} \right]^{-1}. \tag{A.7}$$

On the other hand, the reciprocal of the square bracket in the right-hand side of (A.7) is equal to $p_{0,\cdot}$, according to the M/M/m result. Therefore, we must have $s = a$, which establishes (24).

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press

Appendix B: Proof of Lemma

We start by constructing a sequence of polynomials based on the principal diagonal minors of the determinant $D(z)$. Define

$$Q_0(z) \equiv 1$$
$$Q_1(z) = a_0(z)$$
$$Q_i(z) = a_{i-1}(z)Q_{i-1}(z) - (i-1)\mu_1\lambda_1 z^2 Q_{i-2}(z); \quad i = 2, 3, \ldots, m-1.$$

$Q_i(z)$ is a polynomial of degree $2i$, so it has $2i$ zeros. The following properties hold:

1. $sign[Q_i(0)] = (-1)^i$, $i = 1, 2, \ldots, m-1$.
2. $sign[Q_i(1)] = 1$, $i = 1, 2, \ldots, m-1$.
3. $sign[Q_i(\infty)] = (-1)^i$, $i = 1, 2, \ldots, m-1$.
4. If $Q_{i-1}(z) = 0$ for some z, then $Q_i(z)$ and $Q_{i-2}(z)$ have opposite signs at that point, $i = 2, 3, \ldots, m-1$.

Properties 1–3 follow directly from the form of the relevant determinants. At $z = 0$ only the diagonal elements remain and they are negative. At $z = 1$, adding rows together also leads to products of diagonal elements, which are now positive. At $z = \infty$ the dominant term is a product of negative elements. Property 4 follows from the recurrence relations.

Starting with $Q_1(z)$, we note that according to properties 1-3, its two zeros, $z_{1,1}$ and $z_{1,2}$, are real and satisfy $0 < z_{1,1} < 1 < z_{1,2}$. At those two points, $Q_2(z)$ is negative, according to property 4. Hence, the four zeros of $Q_2(z)$, $z_{2,1}$, $z_{2,2}$, $z_{2,3}$, and $z_{2,4}$, are real and lie in the intervals $(0, z_{1,1})$, $(z_{1,1}, 1)$, $(1, z_{1,2})$, and $(z_{1,2}, \infty)$, respectively. Moreover, the sign of $Q_3(z)$ at point $z_{2,s}$ $(s = 1, 2, 3, 4)$, is $(-1)^{3+s}$ for $s = 1, 2$, and $(-1)^{s-2}$ for $s = 3, 4$.

Continuing in this manner, we find that for $i = 3, 4, \ldots, m-1$, the $2i$ zeros of $Q_i(z)$, $z_{i,1}$, $z_{i,2}$, $\ldots$, $z_{i,2i}$, are real and distinct; the first i of them lie in the consecutive intervals between points 0, $z_{i-1,1}$, $z_{i-1,2}$, $\ldots$, $z_{i-1,i-1}$, 1; the second i are in the intervals between points 1, $z_{i-1,i}$, $z_{i-1,i+1}$, $\ldots$, $z_{i-1,2(i-1)}$, ∞. Moreover, the sign of $Q_{i+1}(z)$ at point $z_{i,s}$ is $(-1)^{i+s+1}$ for $s = 1, 2, \ldots, i$, and $(-1)^{s-i}$ for $s = i+1, i+2, \ldots, 2(i-1)$.

The determinant $D(z)$ is given by

$$D(z) = b_{m-1}(z)Q_{m-1}(z) - (m-1)\mu_1\lambda_1 z^2 Q_{m-2}(z).$$

We have already seen that $D(1) = 0$. Direct evaluation and the above observations show that $\text{sign}[D(0)] = (-1)^m$, and $\text{sign}[D(z_{m-1,s})] = (-1)^{m+s}$ for $s = 1, 2, \ldots, m-1$. Therefore, $D(z)$ has a zero in each of the $m-1$ intervals $(0, z_{m-1,1})$, $(z_{m-1,1}, z_{m-1,2})$, $\ldots$, $(z_{m-1,m-2}, z_{m-1,m-1})$. Moreover, $D(z)$ is negative at $z_{m-1,m-1}$.

Whether there is another zero in the interval $(z_{m-1,m-1}, 1)$ depends on the value of the derivative $D'(1)$. That quantity can be obtained in closed form by adding all rows of $D(z)$ to the last one, dividing that row by $z-1$, setting $z = 1$ and expanding the resulting determinant along the elements of the last row. This yields, as we have already seen in the derivation of (24),

$$D'(1) = \mu_1^{m-1}(m-1)!\mu_2(km - k\rho_1 - \rho_2)\left[\sum_{j=0}^{m-1}\frac{\rho_1^j}{j!} + \frac{m\rho_1^m}{(m-\rho_1)m!}\right]^{-1}. \qquad \textbf{(A.8)}$$

If $km - k\rho_1 - \rho_2 > 0$, then $D'(1) > 0$. Hence, for a sufficiently small ε, $D(1-\varepsilon) < 0$ and $D(1+\varepsilon) > 0$. In that case, $D(z)$ is negative on the interval $(z_{m-1,m-1}, 1)$, that is, there are no other zeros. The Lemma is established and a normalizable solution to the balance equations exists.

If $km - k\rho_1 - \rho_2 = 0$, then $D(z)$ has a double zero at $z = 1$. There is no normalizable solution and the queueing process is recurrent-null. If $km - k\rho_1 - \rho_2 < 0$, then $D(1-\varepsilon) > 0$ for some ε. In that case, $D(z)$ has an extra zero, in the interval $(z_{m-1,m-1}, 1)$. Again, there is no normalizable solution and the process is transient.

https://doi.org/10.1017/S0269964817000286 Published online by Cambridge University Press