Hyytiä, Esa; Aalto, Samuli

On Round-Robin routing with FCFS and LCFS scheduling

Published in:
Performance Evaluation

DOI:
10.1016/j.peva.2016.01.002

Published: 01/03/2016

Document Version
Publisher's PDF, also known as Version of record

Performance Evaluation 97 (2016) 83–103

Contents lists available at ScienceDirect

Performance Evaluation

journal homepage: www.elsevier.com/locate/peva

On Round-Robin routing with FCFS and LCFS scheduling

Esa Hyytiä [a,b,*], Samuli Aalto [b]

[a] Department of Computer Science, University of Iceland, Reykjavík, Iceland
[b] Department of Communications and Networking, Aalto University, Finland

ARTICLE INFO

Article history:
Available online 2 February 2016

Keywords:
Round-Robin
M/G/*m*-RR-system
Erl/G/1-queue
Task assignment
FCFS
LCFS

ABSTRACT

We study the Round-Robin (RR) routing to a system of parallel queues, which serve jobs according to FCFS or preemptive LCFS scheduling disciplines. The cost structure comprises two components: a service fee and a queueing delay related component. With Poisson arrivals, the inter-arrival time to each queue obeys the Erlang distribution. This allows us to study the mean and transient behavior of the queues separately. The service fee is independent of the scheduling and queueing, and we obtain the corresponding mean cost rate and value function in closed forms. With respect to queueing delay, we first derive integral expressions enabling efficient computation of the corresponding size-aware value functions. By decomposition, these yield also the value function for the whole system of m parallel queues fed by RR. Given the value function, one can carry out the first policy iteration step with arbitrary holding cost rates (e.g., delay, slowdown, etc.) yielding efficient size-, cost- and state-aware policies. Moreover, the mean waiting and sojourn times in the corresponding systems get resolved at the same time. The results are demonstrated in the numerical examples, where we compute near optimal job routing policies for sample systems.

1. Introduction

In task assignment (or routing) problems, one chooses a server out of m parallel servers for each new job immediately upon the arrival. The objective is to minimize the mean response time, slowdown, energy consumption, or some other performance quantity of interest. Within each queue, the *First-Come-First-Served* (FCFS) scheduling is usually assumed, but other scheduling disciplines can also be considered. The preemptive *Last-Come-First-Served* (LCFS) is one such candidate possessing several insensitivity properties [1]. Even though task assignment problems have been studied extensively in the literature, only a few optimality results are known, and these generally require homogeneous servers.

The *Join-the-Shortest-Queue* (JSQ) policy assigns a new job to the server with the fewest tasks. Assuming exponentially distributed inter-arrival times and job sizes, and homogeneous servers, Winston [2] showed that JSQ, followed by FCFS, minimizes the mean delay. Since then the optimality of JSQ has been shown in many other settings [3–9]. Similarly, *Round-Robin* (RR), followed by FCFS, has been shown to be the optimal policy when it is only known that the queues were initially in the same state, and the routing history is available [4,10,11]. For RR combined with the *Shortest-Remaining-Process-Time* (SRPT) scheduling, see [12]. With a Poisson arrival process and RR, the inter-arrival times to each queue obey the Erl(m, λ) distribution, where m denotes the number of servers and λ the arrival rate [13]. This enables the analysis of the RR system by studying a single queue at a time [14]. The *Least-Work-Left* (LWL) policy assigns a new job to the queue

* Corresponding author at: Department of Communications and Networking, Aalto University, Finland.
E-mail addresses: esa@hi.is (E. Hyytiä), samuli.aalto@aalto.fi (S. Aalto).

http://dx.doi.org/10.1016/j.peva.2016.01.002

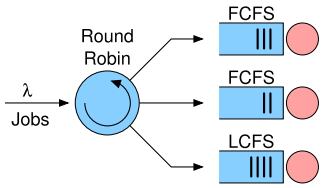


Fig. 1. Round-Robin routing to m FCFS and/or LCFS servers.

with the least amount of unfinished work. LWL is equivalent to M/G/m with a shared queue [15,13], and thus it makes sure no server is idle when there are jobs in the queue. Interestingly, with constant service times and FCFS scheduling, LWL, JSQ and RR make equivalent decisions as in M/D/m (with a shared queue), which can also be shown to be optimal with respect to the mean delay.

The *Myopic* task assignment policy chooses the server that minimizes the objective function assuming that no other jobs arrive later on. With identical LCFS servers, JSQ minimizes the immediate additional delay caused to the present jobs, and thus it is the myopic policy with respect to sojourn time. Similarly, LWL minimizes the sojourn time of the new job in a system of identical FCFS servers, making it the myopic policy in this sense. In some cases, the myopic policy is optimal, but in general this is not the case. Instead, the optimal policy has to take into account also the anticipated new arrivals.

The M/G/m system (with FCFS) is non-trivial to analyze. More results are available for M/D/m. The first analytical result for the distribution of waiting time in M/D/m is by Franx [16]. Numerically much more stable expressions are given by Crommelin [17], who has also analyzed its transient behavior in [18]. In particular, [17] gives the waiting time distribution as a function of the queue length distribution, allowing also the determination of the mean waiting and sojourn times. For extensive surveys on the topic we refer to [19,20].

Value functions for single server queues with *Poisson arrivals* have been derived in [21–24], which form the basis also for value functions for task assignment systems operating under a state-independent policy such as the Bernoulli-split. With state-dependent policies, such as RR and LWL, the queues are coupled and the analysis gets more complicated. In this paper, we derive a set of integral equations that enable efficient computation of the value function with respect to arbitrary holding cost based cost structure in a size-aware task assignment system subject to RR routing policy, and the FCFS or LCFS scheduling disciplines. Jobs arrive according to a Poisson process with rate λ and the servers are assumed to be identical.[1] As an interesting and useful side product, these expressions also provide a new approach to determine the mean waiting time (queueing delay) in the system of parallel FCFS or LCFS servers fed by RR. In particular, with identical FCFS servers, one obtains an M/G/m-RR system. We note that the results given in [17] yield the mean waiting time for M/D/m-RR, but require the determination of the queue length distribution as an intermediate result.

The value functions enable the policy improvement step, yielding a new routing policy. Past work has successfully utilized the Bernoulli-split as the starting policy, whereas our new results enable the same for RR, which is generally a better routing policy. Somewhat surprisingly, the performance of the routing policies based on Bernoulli and RR can be quite similar (see the example cases in Section 5).

The rest of the paper is organized as follows. First, in Section 2, we introduce our notation and analyze the service fees. In Section 3, we analyze a single Erl(m, λ)/G/1-FCFS queue and derive both differential and integral expressions for the value function, whereas Section 4 provides the corresponding results for Erl(m, λ)/G/1-LCFS. Due to the decomposition, the value functions for the systems of parallel FCFS and LCFS queues fed by RR, including M/G/m-RR, are also obtained. Section 5 discusses the task assignment problem and policy iteration, and gives some numerical examples. Section 6 sheds light on possible more elaborate applications, and Section 7 concludes the paper.

2. Preliminaries

In this section, we first describe the model and derive some preliminary results for the Round-Robin routing policy illustrated in Fig. 1. With Poisson arrivals, each queue is an Erl(m, λ)/G/1 system, which facilitates the analysis.

2.1. Model and cost structure

We consider a system of m parallel servers to which jobs arrive according to a Poisson process with rate λ. The service time of job j is denoted by X_j and they are i.i.d., $X_j \sim X$. For *stability*, we require that

$$\lambda\, \mathrm{E}[X] < m.$$

[1] In fact, the analysis does not depend on this and the results hold with very minor modifications also for heterogeneous systems. However, the Round-Robin policy is not an ideal candidate if the service rates are highly asymmetric.

The cost structure comprises two components. First, each job pays a server-specific *service fee* of S_k upon entering a queue k, $k = 1, \ldots, m$. Second, job j incurs costs at a *job-specific holding cost rate* H_j while in the system, $H_j \sim H$. Hence, e.g., the mean cost per job is

$$\sum_k p_k S_k + \mathrm{E}[HT],$$

where p_k is the fraction of jobs routed to server k, and T denotes the sojourn time.

With Round-Robin (RR), the corresponding system is M/G/m-RR as illustrated in Fig. 1. With a Poisson arrival process, each queue behaves according to an Erl(m, λ)/G/1 queue, where the inter-arrival time in each queue is a sum of m independent and exponentially distributed time-intervals, i.e., phases, with the mean durations of $1/\lambda$. Jobs arrive at the end of phase m, after which a new phase 1 starts. The evolution of the queues is naturally coupled as they see the same Poisson process, while each of them is in a different phase $(1, \ldots, m)$.

As a remark, we note that one can also assume that the service time X, the service fee S and the holding cost rate H all may depend on both the job and also on the queue the job is assigned to. In this case, job j is defined by triples $(X_{j,k}, H_{j,k}, S_{j,k})$, where j corresponds to the job and k the queue. The varying service times could be due to different service rates, $X_{j,k} = \nu_k X_j$. The service fee could be, e.g., a function of service time, $S_{j,k} = g_k(X_{j,k})$ (e.g., per bit charging in mobile networks, energy consumption, etc.). Also in this case, it is convenient to assume that jobs are still i.i.d.,

$$(\mathbf{X}_j, \mathbf{H}_j, \mathbf{S}_j) \sim (\mathbf{X}, \mathbf{H}, \mathbf{S}),$$

while the different variables of a single job may depend on each other (e.g., the service fee can be equal to the service time). The stability condition depends on the internal structure of the service times. For RR, the necessary and sufficient stability condition is $\lambda < m/\mathrm{E}[X_{j,k}]$ for all k. Similarly, the necessary stability condition for any *size-unaware* routing policy is $\lambda < \sum_k (\mathrm{E}[X_{j,k}])^{-1}$.

We note that most of the results given in this paper hold immediately or with a minor additional notation also for this more general cost structure. However, RR is a natural choice only with identical servers, and thus in the rest of this paper we assume the basic cost structure with job-specific holding costs, server-specific service fees and equally fast servers. As RR assigns the jobs sequentially, independently of their holding cost rates and service fees, this cost structure is already unnecessarily complicated. Later in Section 5, we consider also routing policies that take into account the job- and server-specific characteristics, and hence the notation.

2.2. Service fees with Round-Robin

Let us first consider the service fees each Erl(m, λ)/G/1 queue incurs. Note that this is only relevant to the routing problem if some server(s) have unequal service fees. In addition, the scheduling discipline does not play any role here. Let $S_j \sim S$ denote the service fee of the jth job assigned to a given queue. Then let i denote the current phase in the arrival process, $i = 1, \ldots, m$, such that at the end of phase m a job arrives and a cost of S_j is incurred.

A sufficient state description with respect to service fees is the current phase of the arrival process. Consequently, the corresponding *value function* depends also only on the current phase, and it is defined as the expected difference between a system initially in phase i and a system initially in equilibrium,

$$v_i \triangleq \lim_{t\to\infty} \mathrm{E}[V_i(t) - r_s t], \tag{1}$$

where $V_i(t)$ denotes the service fees incurred during $(0, t)$ when initially in phase i, and r_s is the mean rate at which service fees are incurred in each queue,

$$r_s = \frac{\lambda\,\mathrm{E}[S]}{m}.$$

Proposition 1. *The value function with respect to service fees for an* Erl(m, λ)/G/1 *queue is*

$$v_i = \frac{2i - m - 1}{2m}\mathrm{E}[S] \tag{2}$$

where i denotes the current phase of the arrival process, $i = 1, \ldots, m$, and $\mathrm{E}[S]$ *is the mean service fee.*

Proof. The value function, as defined in (1), measures the expected difference in the cumulative costs from the given initial phase i to the mean cost rate. For an arbitrary phase i, the so-called Howard's equation is [25]

$$v_i = \frac{m - i + 1}{\lambda}(0 - r_s) + \mathrm{E}[S] + v_1.$$

The first term corresponds to the time interval before the next arrival. During this time no service fees are collected and the difference to the mean cost rate is $0 - r_s$. The factor $(m - i + 1)/\lambda$ corresponds to the mean time duration.

The second term is the mean immediate cost due to the following arrival, after which the arrival process enters to phase 1. The mean difference in costs between phase 1 and the mean cost rate is v_1 by definition, i.e., v_1 takes care of the future costs from that point onwards (recall the Markov property). Consequently,

$$v_i - v_1 = \frac{i-1}{m}\mathrm{E}[S]. \tag{3}$$

As each phase is equally likely, we have $\sum_i v_i = 0$. Taking a sum of (3) over i gives v_1, which in turn yields (2). □

Consider next the whole system of m parallel queues and let $S^{(k)}$ denote the service fee in Queue k. As only the phase matters, a sufficient state description is

$$\mathbf{z} = (q_1, \ldots, q_m),$$

where q_i is the index of the queue currently in phase i. Due to the decomposition, we have the following result:

Corollary 1. *The total value function w.r.t. service fees for the* M/G/m*-RR system in state* $\mathbf{z} = (q_1, \ldots, q_m)$ *is*

$$v(\mathbf{z}) = \frac{1}{2m}\sum_{i=1}^{m}(2i - m - 1)\mathrm{E}[S^{(q_i)}]. \tag{4}$$

Note that the value function is insensitive to the arrival rate λ and depends only on the phases (i.e., the RR sequence) and the mean service fees. In fact, the constant offset in the value functions is irrelevant for our primary application, task assignment (see Section 5), and we can equally use

$$v(\mathbf{z}) = \frac{1}{m}\sum_{i=1}^{m} i\,\mathrm{E}[S^{(q_i)}], \tag{5}$$

from which it is obvious that the Round-Robin sequence assigning the jobs (initially) in the increasing order of the mean service fee so that $\mathrm{E}[S^{(q_i)}] \geq \mathrm{E}[S^{(q_{i+1})}]$ incurs the least costs, as expected.

3. FCFS and Round-Robin

In this section, we focus on the *First-Come-First-Served* scheduling discipline. We are interested in the (remaining) sojourn times. In particular, we assume that each job will incur costs at a certain job-specific holding cost rate, and derive expressions for the value functions, the mean sojourn time and the mean cost rate. We start by considering the so-called *virtual waiting time*, i.e., the waiting time an arriving customer would see. With Poisson arrivals, this quantity is directly related to the actual waiting and sojourn times (Section 3.2).

3.1. Virtual waiting time

With FCFS, the virtual waiting time in the system is equal to the backlog of the queue receiving the next customer. Therefore, in this section, we define the cost rate of the system to be equal to the backlog of the queue in phase m. Considering an individual Erl(m, λ)/G/1-FCFS queue, it thus incurs costs at the rate equal to the backlog only during phase m, at the end of which a new job arrives. This is illustrated in Fig. 2. Due to the PASTA property, the virtual waiting time is directly related to the waiting time the jobs arriving to the M/G/m-RR system see. Let $\tilde{r}$ denote the mean cost rate in a single queue and r in the whole system, $r = \mathrm{E}[W]$. With identical servers, we have the elementary relationship

$$\tilde{r} = r/m.$$

In general, the service times may be heterogeneous across servers. Let $W^{(k)}$ denote the waiting time in server k. Then

$$\begin{cases} r = \mathrm{E}[W] = (1/m)\sum_k \mathrm{E}[W^{(k)}] \\ r = \sum_k \tilde{r}^{(k)} \end{cases} \Rightarrow \mathrm{E}[W^{(k)}] = m\,\tilde{r}^{(k)}.$$

3.1.1. *Single* Erl(m, λ)/G/1-*FCFS Queue*

Consider next a single Erl(m, λ)/G/1-FCFS queue, and let $F(x)$ denote the cdf of the service time X. Let $I_t(i)$ and $U_t(i, u)$ denote the phase of the arrival process and the backlog in the queue at time t, where (i, u) denotes the initial phase and backlog, $I_0(i) = i$ and $U_0(i, u) = u$. In this RR-specific cost structure, the queue incurs costs at rate

$$C_t(i, u) \triangleq 1(I_t(i) = m) \cdot U_t(i, u).$$

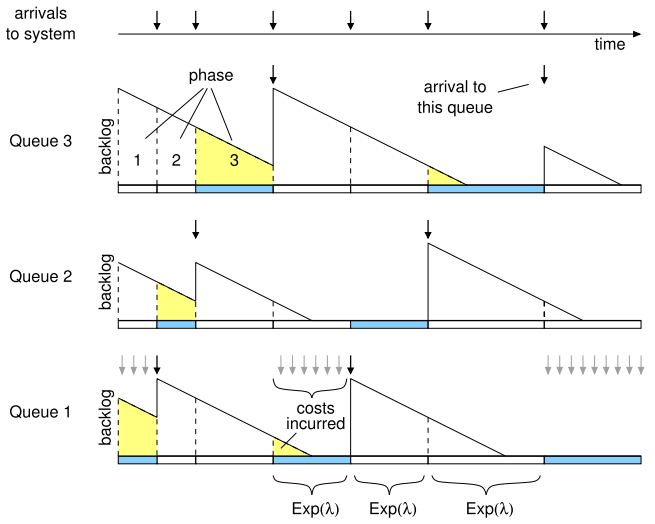


Fig. 2. Sample path with $m = 3$ queues.

Let $v_i(u)$ denote the value function, where i is the initial phase and u the initial backlog. Formally,

$$v_i(u) \triangleq \lim_{t\to\infty} \mathrm{E}[V_i(u,t) - \tilde{r}\,t],$$

where $V_i(u,t)$ denotes the costs a queue initially in state (i,u) incurs during time t,

$$V_i(u,t) \triangleq \int_0^t C_s(i,u)\,ds.$$

Proposition 2. *The size-aware value function of an* Erl(m,λ)/G/1*-FCFS queue with respect to the virtual waiting time satisfies the following system of integro-differential equations,*

$$\begin{aligned} v_i'(u) &= -\tilde{r} + \lambda(v_{i+1}(u) - v_i(u)), \qquad i = 1,\ldots,m-1 \\ v_m'(u) &= u - \tilde{r} + \lambda\left(\mathrm{E}[v_1(u+X)] - v_m(u)\right), \end{aligned} \tag{6}$$

with

$$\begin{aligned} \tilde{r} &= \lambda(v_{i+1}(0) - v_i(0)), \qquad i = 1,\ldots,m-1, \\ \tilde{r} &= \lambda\left(\mathrm{E}[v_1(X)] - v_m(0)\right). \end{aligned} \tag{7}$$

Proof. For $u > 0$, small $\delta > 0$, and for phases $i = 1,\ldots,m-1$

$$v_i(u) = (0 - \tilde{r})\delta + (1-\lambda\delta)\,v_i(u-\delta) + \lambda\delta\,v_{i+1}(u-\delta) + o(\delta).$$

The first term corresponds to the difference between the current cost rate (zero for phases $i \neq m$) and the mean cost rate $\tilde{r}$ multiplied by the time-interval δ. With probability $(1-\lambda\delta)$, the phase remains the same and $v_i(u-\delta)$ gives the future contribution, and with probability $\lambda\delta$, the arrival process moves to the next phase $i+1$, the value of which is given by $v_{i+1}(u-\delta)$. In contrast, at the end of last phase m a new job arrives, and thus for $v_m(u)$ we have

$$v_m(u) = (u - \tilde{r})\delta + (1-\lambda\delta)\,v_m(u-\delta) + \lambda\delta\,\mathrm{E}[v_1(u+X)] + o(\delta),$$

where $\mathrm{E}[v_1(u+X)] = \int_0^\infty v_1(u-\delta+s)\,dF(s)$. By dividing by δ and letting $\delta \to 0$, the above yields (6). Similarly, considering an empty system with $u = 0$ gives (7). □

Note that (6) does not have a unique solution, but a constant term can be added to every $v_i(u)$ without affecting (6). However, the constant term is (typically) immaterial and neglected. Apart from that, the solution is unique. Combining (6) and (7) gives,

$$\begin{aligned} v_i'(0) &= 0, \quad i = 1,\ldots,m, \\ v_i''(0) &= 0, \quad i = 1,\ldots,m-1. \end{aligned} \tag{8}$$

Adding the Eqs. (7) together gives

$$m\tilde{r} = \lambda(\mathrm{E}[v_1(X)] - v_1(0)). \tag{9}$$

Similarly, we have for all $i = 1, \ldots, m-1$

$$v_{i+1}(0) - v_1(0) = \frac{i\tilde{r}}{\lambda}. \tag{10}$$

That is, initially the value functions $v_i(u)$ at $u = 0$ differ by a constant amount of $\tilde{r}/\lambda$.

Note that both (6) and (7) are insensitive to a constant term in the $v_i(u)$. The constant offset in the value functions indeed is generally superfluous and we can set, e.g., $v_1(0) = 0$. Unfortunately, (6) and (7) are difficult to solve even numerically. If the mean cost rate $\tilde{r}$ were available in a closed form, (7) would give the initial values $v_i(0)$ also for $i = 2, \ldots, m$. Moreover, even if the initial values were available, $v_m'(u)$ still depends on the $v_1(u+s)$, $s > 0$, and therefore the standard Runge–Kutta method could not be applied to compute the $v_i(u)$ for $u > 0$. However, one could solve the $v_i(u)$ numerically backwards for $u < u^*$ given the $v_i(u^*)$ were available for some $u^* > 0$. We describe an elegant approach to solve for $v_i(u)$ and $\tilde{r}$ in Section 3.1.5.

Finally, we note that with $m = 1$ the Erl(m, λ)/G/1-FCFS queue reduces to M/G/1-FCFS, for which the exact size-aware value function is available [24],

$$v(u) - v(0) = \frac{u^2}{2(1-\rho)}. \tag{11}$$

It is easy to see that (11) satisfies (6), and applying (7) gives, as expected, the Pollaczek–Khinchine formula for the mean waiting time.

3.1.2. Constant service time Δ

The constant service time Δ is an interesting special case for which the above results have somewhat more simple forms. In this case, the latter equation in (6) reads

$$v_m'(u) = u - \tilde{r} + \lambda(v_1(u+\Delta) - v_m(u)),$$

and we have a first-order system of differential equations. Similarly, the latter equation in (7) reads

$$\tilde{r} = \lambda(v_1(\Delta) - v_m(0)).$$

Finally, (9) reduces to

$$r = m\tilde{r} = \lambda(v_1(\Delta) - v_1(0)).$$

3.1.3. M/G/m-RR/FCFS system

Consider next the whole M/G/m-RR/FCFS system. If all servers are identical then the state of the Round-Robin system with FCFS for jobs arriving later can be described by an m-tuple,

$$\mathbf{z} = (u_1, \ldots, u_m),$$

where u_i denotes the backlog in the queue currently in phase i. In general, the state of the system can be described by

$$\mathbf{z} = ((q_1, u_1), \ldots, (q_m, u_m)),$$

where (q_i, u_i) denotes the server that is currently in phase i and its backlog. Therefore, $(q_1, \ldots, q_m)$ is some permutation of $(1, \ldots, m)$, whereas $u_i \geq 0$ for all i. In the general case, the servers may be heterogeneous and the value functions have to be determined separately for each of them. Let $v_{k,i}(u)$ denote the value function of Queue k currently in phase i. Due to the decomposition to m parallel Erl(m, λ)/G/1-FCFS queues, we again have:

Corollary 2. *The value function w.r.t. virtual waiting time for* M/G/m-RR *in state* $\mathbf{z} = ((q_1, u_1), \ldots, (q_m, u_m))$ *is*

$$v(\mathbf{z}) = v_{q_1,1}(u_1) + \cdots + v_{q_m,m}(u_m). \tag{12}$$

If the service times $X_{j,k}$ are identical for every queue k, then

$$v(\mathbf{z}) = v_1(u_1) + \cdots + v_m(u_m).$$

3.1.4. Asymptotic behavior

Let us next argue that, with relatively broad assumptions, the asymptotic behavior of the value function of the G/G/1-FCFS queue is quadratic. For large u, the backlog decreases with an average rate of $1 - \rho'$, where ρ' denotes the queue specific

offered load. The virtual waiting time incurred during the remaining busy period corresponds to a triangle with initial height u and base (= duration) $u/(1-\rho')$. Therefore,

$$v(u) \approx \frac{u^2}{2(1-\rho')} - \frac{u}{1-\rho'} \cdot r + v(0),$$

where the first term corresponds to the costs incurred during the remaining busy period, the second term to the mean cost rate during the same time-interval, and the third term corresponds to what happens after that. For large u, the first quadratic term dominates.

With Erl(m, λ)/G/1, in the context of RR, the costs are accrued only in the final phase m as explained above (see also Fig. 2). Let ρ denote the offered load to the whole system, $\rho = \lambda\,\mathrm{E}[X]$, so that $\rho' = \rho/m$. The costs accrued during the remaining busy period are roughly $1/m$ of the "full triangle", i.e., for $u \gg 1$ we have,

$$v_i(u) \approx \frac{u^2}{2(m-\rho)}, \quad \text{and} \quad v_i'(u) \approx \frac{u}{m-\rho}. \tag{13}$$

3.1.5. Numerical solution for $v_i(u)$

The Eqs. (6) that the value functions $v_i(u)$ must satisfy can be written in an integral form that is suitable for numerical computations. Without loss of generality, we assume here that $v_1(0) = 0$.

Proposition 3. *For the* Erl(m, λ)/G/1*-FCFS queue, the value function $v_i(u)$ with respect to the virtual waiting time satisfies the following system of integral equations,*

$$\begin{aligned} v_i(u) &= \left(e^{-\lambda u} - \frac{1}{i}\right) v_{i+1}(0) + \lambda \int_0^u e^{-\lambda(u-s)} v_{i+1}(s)\, ds, \quad i = 1, \ldots, m-1, \\ v_m(u) &= \left(e^{-\lambda u} - \frac{1}{m}\right) \int_0^\infty v_1(s)\, dF(s) + \frac{e^{-\lambda u} + \lambda u - 1}{\lambda^2} + \lambda \int_0^u e^{-\lambda(u-s)} \int_0^\infty v_1(s+\ell)\, dF(\ell)\, ds. \end{aligned} \tag{14}$$

Proof. For $i = 1, \ldots, m-1$, multiplying both sides of (6) with $e^{\lambda u}$ gives

$$e^{\lambda u} v_i'(u) + \lambda e^{\lambda u} v_i(u) = e^{\lambda u}(-\tilde{r} + \lambda v_{i+1}(u)).$$

The left-hand side is equal to $(d/du)\, e^{\lambda u} v_i(u)$, yielding

$$\begin{aligned} e^{\lambda u} v_i(u) &= \int_0^u e^{\lambda s}(-\tilde{r} + \lambda v_{i+1}(s))\, ds + v_i(0), \\ v_i(u) &= \frac{\tilde{r}}{\lambda}(e^{-\lambda u} - 1) + e^{-\lambda u} v_i(0) + \lambda \int_0^u e^{-\lambda(u-s)} v_{i+1}(s)\, ds. \end{aligned}$$

Similarly, for phase m one obtains

$$v_m(u) = \frac{\tilde{r}}{\lambda}(e^{-\lambda u} - 1) + e^{-\lambda u} v_m(0) + \frac{e^{-\lambda u} + \lambda u - 1}{\lambda^2} + \lambda \int_0^u e^{-\lambda(u-s)} \int_0^\infty v_1(s+\ell)\, dF(\ell)\, ds.$$

As we are generally interested in the differences between the relative values, we can set $v_1(0) = 0$ so that by (10)

$$v_i(0) = \frac{(i-1)\tilde{r}}{\lambda}, \quad \text{for } i = 1, \ldots, m,$$

and

$$\frac{\tilde{r}}{\lambda} = \begin{cases} \dfrac{v_{i+1}(0)}{i}, & \text{for } i = 1, \ldots, m-1, \\ \dfrac{1}{m} \displaystyle\int_0^\infty v_1(s)\, dF(s). \end{cases}$$

Substituting these into the above gives (14). □

Corollary 3. *For a constant service time Δ, the latter equation in (14) reads*

$$v_m(u) = \left(e^{-\lambda u} - \frac{1}{m}\right) v_1(\Delta) + \frac{e^{-\lambda u} + \lambda u - 1}{\lambda^2} + \lambda \int_0^u e^{-\lambda(u-s)} v_1(s+\Delta)\, ds. \tag{15}$$

Note that (14) expresses $v_i(u)$ as a function of $v_{i+1}(u)$ for $i = 1, \ldots, m-1$, and $v_m(u)$ as a function of $v_1(u)$. Given an initial guess for any $v_i(u)$, provided, e.g., by (13), the Eqs. (14) can be iterated until they converge.

Algorithm 1 shows the computation of the value functions for the Erl(m, λ)/D/1-FCFS queue with respect to the virtual waiting time at a total of k possible backlog levels, $u_j = j\Delta_u$, $j = 0, \ldots, k-1$, where $\Delta_u = x/n$ is the constant discretization interval, x the fixed job size and n some positive integer number. That is, the value function is computed at $m \times k$ points, $v_i(u_j) = v_{i,j}$, where $i = 1, \ldots, m$ and $j = 0, \ldots, k-1$. We use (13) for the initial values of $v_{i,j}$, and then iterate (14) until the $v_{i,j}$ converge. Note that it is important to use quadratic extrapolation for $v_i(u)$ when $u > (k-1)\Delta_u$. In our numerical experiments, this iteration converged relatively fast making it computationally appealing.

Data: k = points of discretization, Δ_u = discretization interval, $x = n \cdot \Delta_u$ = service time, and $\rho = \lambda x$
Result: Value function $v_{i,j} = v_i(j \cdot \Delta_u)$ for Erl(m, λ)/D/1-FCFS, where $i = 1, \ldots, m$ and $j = 0, \ldots, k-1$. ($u_j = j \cdot \Delta_u$)
for $j \leftarrow 0$ **to** $k-1$ **do**
  $v_{1,j} = (j \cdot \Delta_u)^2/(2(m-\rho))$; // Initial values for phase $i = 1$ using (13)
repeat
  for $i \leftarrow m$ **downto** 1 **do**
    if $i = m$ **then**
      $i' \leftarrow 1$ and $n' \leftarrow n$
    else
      $i' \leftarrow i+1$ and $n' \leftarrow 0$
    $c \leftarrow v_{i',n'}$ and $y_1 \leftarrow c$ and $I \leftarrow 0$;
    $v_{i,0} \leftarrow c(i-1)/i$;
    for $j = 1$ **to** $k-1$ **do**
      $u \leftarrow j \cdot \Delta_u$;
      $y_0 \leftarrow y_1$;
      $y_1 \leftarrow e^{\lambda u} v_{i',n'+j}$; // Use quadratic extrapolation for $v_{i,j}$ when $n'+j \geq k$
      $I \leftarrow I + \frac{y_0 + y_1}{2} \cdot \Delta_u$; // Trapezoidal rule for one interval
      $v_{i,j} \leftarrow e^{-\lambda u}(\lambda I + c) - c/i$;
      if $i = m$ **then**
        $v_{i,j} \leftarrow v_{i,j} + (e^{-\lambda u} + \lambda u - 1)/\lambda^2$
until *the* $v_{i,j}$ **converge**;

Algorithm 1: Numerical computation of the value function for the Erl(m, λ)/D/1-FCFS queue.

As a convenient side product of being able to determine the value functions efficiently using (14), also the *mean waiting time*, $r = \mathrm{E}[W]$, is obtained directly from (7):

Corollary 4. *The value functions give also the mean waiting time* $\mathrm{E}[W]$ *in the* Erl(m, λ)/G/1*-FCFS queue,*

$$\mathrm{E}[W] = \lambda\, m\, (v_2(0) - v_1(0)). \tag{16}$$

In order to compute $\mathrm{E}[W]$, we do not need to find the waiting time distribution first (which itself is non-trivial, [17]).

For $m = 1$, the insensitive solution (11) can be shown to satisfy (14). That is, for the M/G/1 queue we have,

$$v(u) = \left(e^{-\lambda u} - 1\right) \int_0^\infty v(s)\, dF(s) + \frac{e^{-\lambda u} + \lambda u - 1}{\lambda^2} + \lambda \int_0^u e^{-\lambda(u-s)} \int_0^\infty v(s+\ell)\, dF(\ell)\, ds,$$

which the "trial"

$$v(u) = \frac{u^2}{2(1-\rho)}$$

satisfies independently of the service time distribution.

3.1.6. Generalized Round-Robin (GRR)

RR is a special case of the *Generalized Round-Robin* policies defined by typically periodic sequences [26–28]. RR is optimal under certain assumptions when the servers are identical. However, if some servers have different service rates, then the even split of tasks that RR carries out may no longer make sense. In [26], Hajek proves the intuitive result that among a very large class of arrival process, the one with constant inter-arrival times is optimal for a single server queue with exponentially distributed service times. Then, in [27], he shows that the so-called most-regular-sequence is optimal for two, not necessarily identical, servers when jobs again are exponentially distributed.

Suppose that the (external) sequence defining the task assignments is periodic with M denoting the length of the period. Without lack of generality, we can consider Queue 1. With respect to service fees, both the mean rate and value function

are straightforward to deduce. We omit these for brevity. For the virtual waiting time, let $v_i(u)$ again denote its value function with respect to the backlog, and where a_i is an indicator for whether an arrival will be sent to queue 1 during phase i. (Note that now we can have multiple phases with arrivals in the same period.) For notational convenience, we define $v_{i+M}(u) \triangleq v_i(u)$. As before, we have a system of differential equations,

$$v_i'(u) = \begin{cases} -\tilde{r} + \lambda(v_{i+1}(u) - v_i(u)), & \text{if } a_i \neq 1, \\ u - \tilde{r} + \lambda \int (v_{i+1}(u+t) - v_i(u))\, dF(t), & \text{if } a_i = 1, \end{cases}$$

where the initial values are coupled,

$$v_i'(0) = \begin{cases} -\dfrac{\tilde{r}}{\lambda} + v_{i+1}(0), & \text{if } a_i \neq 1, \\ -\dfrac{\tilde{r}}{\lambda} + \displaystyle\int v_{i+1}(t)\, dF(t), & \text{if } a_i = 1. \end{cases}$$

That is, the generalized (periodic) RR can be analyzed essentially the same way as RR. Also probabilistic variants, where subsequences are chosen with certain probabilities (for load-balancing reasons), are amenable to the same approach.

3.2. Waiting and sojourn time

The backlog based cost rate $c(\mathbf{z}) = u_m$ can be seen as a (linear) penalty for a long queue length. However, typically one is interested in the actual *waiting or sojourn time*, possibly weighted with arbitrary job-specific holding costs. Consequently, next we consider M/G/m-RR/FCFS with identical servers and job-specific holding costs (i.e., the holding costs do not depend on the queue).

Let $U(t) = U_m(t)$ denote the virtual waiting time at time t. With FCFS, this is the waiting time an arriving customer sees, $W \sim U$. The mean cost rate w.r.t. waiting time is

$$r_W = \lambda \cdot r = \lambda \cdot \mathrm{E}[W], \tag{17}$$

i.e., the rate at which the system incurs waiting time. For the sojourn time, $r_T = \lambda \cdot (\mathrm{E}[W] + \mathrm{E}[X]) = \mathrm{E}[N]$. The state of a queue does not affect the service time of a job, and this is the same for all servers, so the (relative) value function with respect to waiting and sojourn times are the same:

Proposition 4. *The size-aware value function for an* M/G/m*-RR/FCFS system with respect to waiting or sojourn time is*

$$\tilde{v}(\mathbf{z}) - \tilde{v}(0) = \lambda\ (v(\mathbf{z}) - v(0))\,, \tag{18}$$

where $v(\mathbf{z})$ is the value function w.r.t. virtual waiting time.

Proof. Let $W_1, W_2, \ldots$ denote the waiting times for future arrivals. Service times of the jobs arriving in the future are independent of the state and thus do not show up in the value function. One can associate the costs in two equivalent ways for these arrivals until the end of the current busy period (renewal point),

$$\begin{aligned} \tilde{c}_1 &\triangleq \lambda\, \mathrm{E}\left[\int_0^{B_{\mathbf{z}}} U_m(t)\, dt\right], \\ \tilde{c}_2 &\triangleq \mathrm{E}[W_1 + \cdots + W_{N_{\mathbf{z}}}], \end{aligned} \tag{19}$$

where $B_{\mathbf{z}}$ denotes the duration of the (remaining) busy period (having a finite mean), and $N_{\mathbf{z}}$ the number of jobs arriving during it. Due to the PASTA property, $\tilde{c}_1 = \tilde{c}_2$. The first equation corresponds to the virtual waiting time based holding cost $c(\mathbf{z}) = u_m$ multiplied by the arrival rate λ, and the latter to the actually incurred waiting time. Then,

$$\tilde{v}(\mathbf{z}) - \tilde{v}(0) = \mathrm{E}[W_1 + \cdots + W_{N_{\mathbf{z}}}] - r_W\, \mathrm{E}[B_{\mathbf{z}}].$$

According to (19),

$$\mathrm{E}[W_1 + \cdots + W_{N_{\mathbf{z}}}] = \lambda\, \mathrm{E}\left[\int_0^{B_{\mathbf{z}}} U_m(t)\right],$$

and similarly, with (17), we obtain

$$r_W\, \mathrm{E}[B_{\mathbf{z}}] = \lambda\, r\ \mathrm{E}\left[\int_0^{B_{\mathbf{z}}} 1\, dt\right] = \lambda\, \mathrm{E}\left[\int_0^{B_{\mathbf{z}}} r\, dt\right].$$

Therefore,

$$\tilde{v}(\mathbf{z}) - \tilde{v}(0) = \lambda\, \mathrm{E}\left[\int_0^{B_{\mathbf{z}}} (U_m(t) - r)\, dt\right].$$

By definition, $v(\mathbf{z}) = \mathrm{E}[\int_0^{B_{\mathbf{z}}} (U_m(t) - r)\, dt] + v(0)$, yielding (18). □

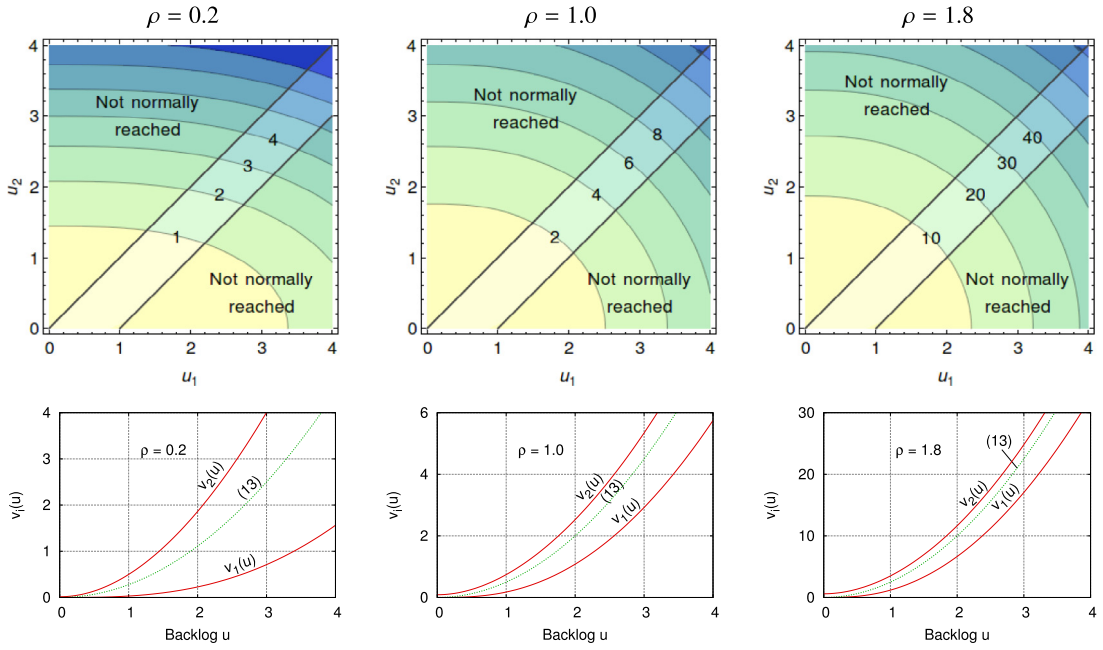


Fig. 3. Value function for M/D/2-RR/FCFS with respect to the virtual waiting time. The dotted curve in the lower graphs corresponds to (13).

3.3. General job-specific holding costs

Consider next the case with arbitrary *job-specific holding costs*. Let (X_j, H_j) denote the size and the holding cost of job j, which are assumed to be i.i.d., $(X_j, H_j) \sim (X, H)$, while each X_j and H_j may still depend on each other. For example, the slowdown metric, defined as the ratio of the delay to the service time, is obtained with $H_j = 1/X_j$ [29]. The difference in the cumulative costs is incurred by later arriving jobs during their waiting time. Therefore:

Corollary 5. *The size-aware value function w.r.t. general holding costs (associated with the waiting or sojourn time) for* M/G/m*-RR/FCFS in state* $\mathbf{z} = (u_1, \ldots, u_m)$ *is*

$$\tilde{v}(\mathbf{z}) = \lambda \sum_i v_i(u_i)\, \mathrm{E}[H] = \lambda\, \mathrm{E}[H]\; v(\mathbf{z}). \tag{20}$$

3.4. Examples

Next we give some numerical examples with the virtual waiting time. The service fee is included in the cost later in Section 5.

3.4.1. M/D/2 w.r.t. virtual waiting time

Consider next the virtual waiting time in M/D/2-RR. Fig. 3 depicts the equivalue contours of the resulting value function $v(\mathbf{z}) = v_1(u_1) + v_2(u_2)$ for the whole system (upper row) and its components (lower row) for $\rho = 0.2,\ 1.0,\ 1.8$. For the upper row, we have included also states with $u_2 > u_1$ and $u_1 > u_2 + 1$, even though the normal state space[2] of an initially empty M/D/2-RR queue is the narrow strip constrained by $0 \le u_2 \le u_1 \le u_2 + 1$. That is, we allow an arbitrary initial state. The value function becomes more symmetric as ρ increases. From the lower row, we observe that initially, at $u = 0$, the slope of each $v_i(u)$ is zero in accordance with (8). The dotted line in the lower row corresponds to the approximation of $v_i(u)$ given in Eq. (13).

3.4.2. Other distributions

Let $D(x_1, x_2)$ denote the two point distribution on $x_1 < 1 < x_2$ with probabilities such that the mean is 1. Similarly, $D(x)$ denotes the deterministic distribution with one outcome x and $U(x_1, x_2)$ the uniform distribution on interval (x_1, x_2). Fig. 4 illustrates the resulting value functions with $D(0.5, 10)$, $D(0.5, 5)$, $U(0, 2)$ and $D(1)$. In each case, the mean service time is 1

[2] Later, in Section 5, the so-called FPI policy may deviate from RR and any state is in principle possible.

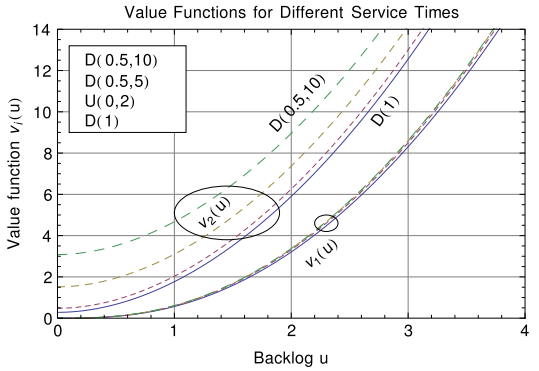


Fig. 4. Value function $v_i(u)$ for $m = 2$ phases ($i = 1, 2$) when service times obey $D(0.5, 10)$, $D(0.5, 5)$, $U(0, 2)$ and $D(1)$ with unit mean and $\lambda = 1.6$. For both $v_1(u)$ and $v_2(u)$, service time distribution $D(0.5, 10)$ corresponds to the highest curve and $D(1)$ to the lowest (in the order of variability).

and arrival rate $\lambda = 1.6$. We can observe that the higher the variance in the service times is, the higher the initial difference $v_2(0) - v_1(0)$ is. The $v_1(u)$ behave rather similarly (note the initial choice $v_1(0) = 0$). Consequently, we have the following result:

Corollary 6. *The size-aware value function for an* Erl(m, λ)/G/1*-FCFS queue w.r.t. waiting time (or delay) is not insensitive to the job size distribution (for* $m > 1$*).*

We note that this is in contrast to the M/G/1-FCFS queue, for which the size-aware value function is insensitive to the job size distribution [24]. This implies that the value function of the corresponding Round-Robin system, due to the decomposition, is also sensitive to the job size distribution, which again is not the case if the routing is by any state-independent (static) policy such as the random Bernoulli-split.

4. LCFS with Round-Robin

In this section, we focus on the *Last-Come-First-Served* (LCFS) scheduling discipline, where preemption is assumed. LCFS is a robust scheduling discipline and well understood in the context of M/G/1 queues. In particular, the mean sojourn time in M/G/1-LCFS, as in M/G/1-PS, is insensitive to the service time distribution. Moreover, the value function of the size-aware M/G/1-LCFS queue is insensitive [1,24].

Here we first consider a single Erl(m, λ)/G/1-LCFS queue, and then apply those results to an M/G/m-RR/LCFS multi-server system. We note that with LCFS the current state of the system does not affect the sojourn times of the jobs arriving in the future, whereas with FCFS the situation is quite the opposite as the jobs arriving in future do not affect the present jobs. In particular, this means that the value function of an LCFS system with respect to holding costs depends solely on the expected remaining sojourn times of the present jobs, which in turn depend on the current phase of the arrival process. We start by considering the busy period in an arbitrary work-conserving Erl(m, λ)/G/1 queue.

4.1. Busy period in an Erl(m, λ)/G/1 queue

We are interested in the expected remaining busy period, $b_i(u)$, in an arbitrary work-conserving Erl(m, λ)/G/1 queue when the arrival process is initially in phase i and the backlog in the queue is u. By its definition, for $\lambda\, \mathrm{E}[X] < m$,

$$\lim_{u \to 0} b_i(u) = 0, \quad \forall i. \tag{21}$$

For an arbitrary $u > 0$ we have the following result:

Proposition 5. *The expected remaining busy period* $b_i(u)$ *in an* Erl(m, λ)/G/1 *queue in phase* i *with backlog* u *satisfies the following set of differential equations:*

$$\begin{aligned} b_i'(u) &= 1 + \lambda(b_{i+1}(u) - b_i(u)), && i = 1, \ldots, (m-1), \\ b_m'(u) &= 1 + \lambda(\mathrm{E}[b_1(u + X)] - b_m(u)), && i = m. \end{aligned} \tag{22}$$

Proof. For $i < m$, $u > 0$ and small time-interval δ, we can write

$$b_i(u) = \delta + \lambda\delta \cdot b_{i+1}(u - \delta) + (1 - \lambda\delta) \cdot b_i(u - \delta) + o(\delta),$$

which in the limit as $\delta \to 0$ gives

$$b_i'(u) = 1 + \lambda(b_{i+1}(u) - b_i(u)).$$

Similarly, for $i = m$ and $u > 0$ we have

$$b_m(u) = \delta + \lambda\delta \cdot \mathrm{E}[b_1(u - \delta + X)] + (1 - \lambda\delta) \cdot b_i(u - \delta) + o(\delta),$$

where X denotes the service time of the job arriving next (by conditioning). This gives in the limit as $\delta \to 0$

$$b'_m(u) = 1 + \lambda(\mathrm{E}[b_1(u + X)] - b_m(u)),$$

which completes the proof. □

With $m = 1$, the system reduces to an M/G/1 queue and $b(u) = u/(1 - \rho)$, which satisfies (21) and (22). The solution for $m > 1$ is unfortunately more complex:

Corollary 7. *The expected remaining busy periods $b_i(u)$ in an* Erl(m, λ)/G/1 *queue with $m > 1$ are nonlinear functions of the initial backlog u.*

Proof. We prove this by contradiction. Suppose that for some $i < m$ and $\epsilon > 0$,

$$b_i(u) = a_i u, \quad u \in [0, \epsilon].$$

Substituting this into (22) gives

$$b'_i(u) = 1 + \lambda(b_{i+1}(u) - b_i(u)) \Rightarrow a_i = 1 + \lambda(b_{i+1}(u) - a_i u).$$

First the above means that also $b_{i+1}(u), \ldots, b_m(u)$ are all linear with the same constant factor a_i. Second, in the limit as $u \to 0$, we have $b_i(u) \to 0$ and therefore $a_i = 1$. In particular, $b_m(u) = u$ for $u \in [0, \epsilon]$. On the other hand, for $u \in [0, \epsilon]$ we also have, by (22),

$$b'_m(u) = 1 + \lambda(\mathrm{E}[b_1(u + X)] - b_m(u)) \Rightarrow 1 = 1 + \lambda(\mathrm{E}[b_1(u + X)] - u),$$

and again as $u \to 0$, we obtain

$$1 = 1 + \lambda\,\mathrm{E}[b_1(X)] \Rightarrow \mathrm{E}[b_1(X)] = 0,$$

which is a contradiction. Therefore no $b_i(u)$ with $i < m$ can be linear in $[0, \epsilon]$. The case $i = m$ follows directly from the latter part. □

Corollary 8. *The expected remaining busy periods $b_i(u)$ in an* Erl(m, λ)/G/1 *queue are sensitive to the shape of the service time distribution.*

Proof. We prove by contradiction. Consider two service time distributions with the same mean x: (i) with a fixed value x, and (ii) with two values $x - \Delta_1$ and $x + \Delta_2$ obtained with probabilities p_1 and p_2. Then we utilize (22),

$$b'_m(u) = 1 + \lambda(\mathrm{E}[b_1(u + X)] - b_m(u)),$$

in the limit as $u \to 0$, which gives

$$b'_m(0) = 1 + \lambda\,\mathrm{E}[b_1(X)].$$

If $b_i^{(1)}(u)$ and $b_i^{(2)}(u)$ were insensitive, then the above would hold for both distributions and

$$b_1(x) = p_1\, b_1(x - \Delta_1) + p_2\, b_1(x + \Delta_2).$$

For the same mean, $p_1 = \Delta_2/(\Delta_1 + \Delta_2)$ and $p_2 = \Delta_1/(\Delta_1 + \Delta_2)$, and we have

$$b_1(x) = \frac{\Delta_2}{\Delta_1 + \Delta_2}\, b_1(x - \Delta_1) + \frac{\Delta_1}{\Delta_1 + \Delta_2}\, b_1(x + \Delta_2).$$

The above holds for every choice of (Δ_1, Δ_2) only if $b_1(u)$ is linear. However, in the proof of Corollary 7 we showed that the $b_i(u)$ are nonlinear for $u \in [0, \epsilon]$, which leads to a contradiction. □

As with FCFS, the $b_i(u)$ can be solved numerically by transforming (22) to an integral form:

Corollary 9. *The expected remaining busy periods $b_i(u)$ in an* Erl(m, λ)/G/1 *queue initially in phase i with backlog u satisfy the following set of integral equations:*

$$\begin{aligned} b_i(u) &= \frac{1 - e^{-\lambda u}}{\lambda} + \lambda \int_0^u e^{-\lambda(u-s)}\, b_{i+1}(s)\, ds, \qquad i = 1, \ldots, (m-1) \\ b_m(u) &= \frac{1 - e^{-\lambda u}}{\lambda} + \lambda \int_0^u e^{-\lambda(u-s)}\, \mathrm{E}[b_1(s + X)]\, ds. \end{aligned} \tag{23}$$

Proof. Multiplying both sides of (22) by $e^{\lambda u}$ gives

$$\begin{aligned} e^{\lambda u} b_i'(u) + \lambda\, e^{\lambda u} b_i(u) &= e^{\lambda u} + \lambda e^{\lambda u} b_{i+1}(u), \qquad i = 1, \ldots, (m-1), \\ e^{\lambda u} b_m'(u) + \lambda\, e^{\lambda u} b_m(u) &= e^{\lambda u} + \lambda e^{\lambda u} \mathrm{E}[b_1(u+X)], \quad i = m. \end{aligned}$$

Hence,

$$\begin{aligned} \frac{\partial}{\partial u} e^{\lambda u} b_i(u) &= e^{\lambda u} + \lambda e^{\lambda u} b_{i+1}(u), \qquad i = 1, \ldots, (m-1), \\ \frac{\partial}{\partial u} e^{\lambda u} b_m(u) &= e^{\lambda u} + \lambda e^{\lambda u} \mathrm{E}[b_1(u+X)], \quad i = m. \end{aligned}$$

Integrating both sides from 0 to u (see Section 3.1.5) then gives (23). □

A numerical solution for $b_i(u)$ can be obtained by iterating (23) until the functions converge. An initial solution is provided, e.g., by the mean remaining busy period in an equivalent M/G/1 queue [30], giving

$$b_i^{(0)}(u) = \frac{u}{1 - \lambda\, \mathrm{E}[X]/m}.$$

Corollary 10. *For a constant service time Δ, the latter equation in (23) reads*

$$b_m(u) = \frac{1 - e^{-\lambda u}}{\lambda} + \lambda \int_0^u e^{-\lambda(u-s)}\, b_1(s+\Delta)\, ds. \tag{24}$$

Asymptotically, for $u \gg \mathrm{E}[X]$, it is easy to show that

$$b_i(u) \approx \frac{u}{1-\rho},$$

and moreover,

$$b_1(u) - b_i(u) \to \frac{(i-1)\,\mathrm{E}[X]}{m(1-\rho)}.$$

4.2. *Single* Erl(m, λ)/G/1-*LCFS queue*

Next we will apply the previous results to the Erl(m, λ)/G/1-LCFS queue. With LCFS, the current state of the queue does not affect the sojourn time of the arriving job that will preempt a job currently receiving service, if any. This means that the arriving jobs see the system as if it were empty. Therefore, we have an elementary relationship between the sojourn times and busy periods:

Lemma 1. *The mean conditional sojourn time of a job with service time x in an* Erl(m, λ)/G/1-*LCFS queue is*

$$\mathrm{E}[T \mid x] = b_1(x), \tag{25}$$

and the mean sojourn time is $\mathrm{E}[T] = \mathrm{E}[b_1(X)]$.

Similarly, $b_i(u)$ gives the expected sojourn time of a job that sees a backlog of u in front (including the job's own service time) when the phase of the arrival process is i. Note that according to Little's result, the term $\lambda\, \mathrm{E}[b_1(X)]$, that appeared frequently in the previous section, corresponds to the mean number in the queue, $\mathrm{E}[N]$.

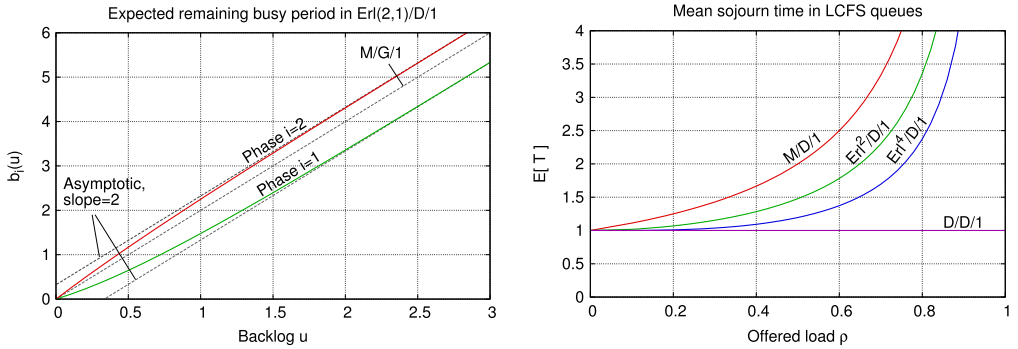
We can also write the size-aware value function for Erl(m, λ)/G/1-LCFS in terms of $b_i(u)$. Let z denote the state of the queue, $z = (i; (h_1, x_1), \ldots, (h_n, x_n))$, where i denotes the current phase of the arrival process, and (h_j, x_j) are the holding cost rate and the (remaining) service time of job j. Job 1 (if there are any jobs present) is currently receiving service (if any), and the last job n sees an effective backlog equal to $x_1 + \cdots + x_n$.

Corollary 11. *The size-aware value function of the* Erl(m, λ)/G/1-*LCFS queue in phase i with respect to job-specific holding costs h_j (associated with the waiting or sojourn time) satisfies*

$$v_i(z) - v_i(0) = \sum_{j=1}^{n} \left(h_j \cdot b_i \left(\sum_{k=1}^{j} x_k \right) \right),$$

where $v_i(0) = v_1(0) + (i-1)\mathrm{E}[N]/\lambda$ and $\mathrm{E}[N]$ is the mean queue length.

That is, the value function is equal to the costs the present n jobs will incur on average before they depart the system, i.e., the sum of their holding cost h_j multiplied by the expected remaining sojourn time $b_i(\cdot)$.



(a) Remaining busy period in Erl(m, λ)/D/1. (b) Mean sojourn time with LCFS.

Fig. 5. (a) The expected remaining busy period (or sojourn time) in an Erl(2, λ)/D/1-LCFS queue with $\lambda = 1$ and $\rho = 0.5$, and (b) a comparison of the mean sojourn time in different LCFS queues.

4.3. M/G/m-RR/LCFS multi-server system

Let us next consider the corresponding multi-server system, M/G/m-RR/LCFS queue. We recall that jobs arrive according to a Poisson process with rate λ, and are routed in Round-Robin fashion to m available servers so that each server receives jobs with Erl(m, λ) distributed time-intervals. The state of the system is defined by $\mathbf{z} = (z_1, \ldots, z_m)$, where $z_i = ((h_{i,1}, x_{i,1}), \ldots, (h_{i,n_i}, x_{i,n_i}))$ denotes the jobs in the server that is currently in phase i. Then, as with FCFS, the system decomposes (in terms of average behavior) and we have:

Corollary 12. *The size-aware value function of M/G/m-RR/LCFS system is*

$$v(\mathbf{z}) = \sum_{i=1}^{m} v_i(z_i). \tag{26}$$

4.3.1. Generalized Round-Robin (GRR)

As in Section 3.1.6, we can consider a generalized Round-Robin sequence with periodicity of M in the context of LCFS servers. Without loss of generality, we can focus on Server 1 and let a_i indicate if a job is assigned to Server 1 at the end of phase i, $i = 1, \ldots, M$. Then, for the expected remaining busy period $b_i(u)$, it holds that

$$b_i(u) = \begin{cases} \dfrac{1 - e^{-\lambda u}}{\lambda} + \lambda \displaystyle\int_0^u e^{-\lambda(u-s)}\, b_{i+1}(s)\, ds, & \text{if } a_i \neq 1, \\ \dfrac{1 - e^{-\lambda u}}{\lambda} + \lambda \displaystyle\int_0^u e^{-\lambda(u-s)}\, \mathrm{E}[b_1(s+X)]\, ds, & \text{if } a_i = 1, \end{cases}$$

where for the notational simplicity we have $b_{i+M}(u) = b_i(u)$.

4.4. Examples with LCFS

Next we will illustrate the analytical results obtained for LCFS. Fig. 5(a) depicts the expected remaining busy period $b_i(u)$ in an Erl(m, λ)/D/1 queue with $\lambda = 1$, $m = 2$, and constant service times, $X \equiv 1$, so that $\rho = 0.5$. We recall that the $b_i(u)$ also give the expected remaining sojourn times under LCFS.

The asymptotic linear behavior with slope $(1 - \rho)^{-1} = 2$ is self-evident already when the backlog is $u > 2$, as indicated with the dotted lines. Fig. 5(b) depicts the mean sojourn time in Erl(m, λ)/D/1-LCFS with $m = 1, 2, 4, \infty$, where the case $m = \infty$ corresponds to a D/D/1 queue without any queueing. We can see that a more regular arrival pattern decreases the mean sojourn time also with LCFS.

5. Task assignment problem

In this section, we consider the routing (task assignment) problem in the system of m parallel servers with job- and server-specific service fees and holding costs. As reference routing policies, we consider the following:

RR: Round-Robin assigns jobs using a predefined sequence of servers: $s_1, \ldots, s_m, s_1, \ldots$ where $s_i \neq s_j$ for $i \neq j$.
RND: Bernoulli-split assigns jobs randomly and independently using probabilities $p_1, \ldots, p_m$.
JSQ: Join-the-shortest-queue assigns a new job to queue with the least number of jobs.

LWL: Least-work-left assigns a new job to the queue with the shortest backlog.
Myopic chooses the queue that minimizes the costs assuming no other jobs arrive in future.

Ties are broken in favor of the queue with a smaller index.

5.1. Policy iteration

Policy iteration is a standard technique of the MDP framework to improve a given policy based on a value function [31,25,32]. In layman's terms, at every state, it chooses the action a for which the sum of the immediate cost and the change in the future cumulative costs for the given policy is the smallest. Here we carry out the first policy iteration round, and let FPI denote the resulting policy.

When the basic policy is a static policy such as RND, the action a simply defines the server for job j. However, if the basic policy is RR, then an action can define two things:

1. The queue for the new job.
2. The future RR sequence (the phases for queues).

The latter decision is fictitious and made for evaluation purposes only as the next job will be eventually assigned by FPI, not by RR.

Let us first consider the service fees, which are independent of the scheduling discipline. The value function for basic policy RR is given by (5), which depends on the phases of the queues. The total service fee cost of action a is

$$c_{\mathrm{S}}(a) = s_j^a + v_{\mathrm{S}}(\mathbf{z} \oplus a) - v_{\mathrm{S}}(\mathbf{z}),$$

where s_j^a is the service fee of the current job, $\mathbf{z}$ is the current state of the system, and $\mathbf{z} \oplus a$ the state after action a. Note that we have made it explicit that the service fee may depend on the chosen queue. In contrast, with any static (basic) policy such as RND, the state of the system does not affect how the system incurs service fees in the future, and the corresponding value function is a constant and can be omitted. In this case, one considers only the immediate cost, $c_{\mathrm{S}}(a) = s_j^a$.

With FCFS, no later arriving job affects the sojourn time of the present jobs, and the waiting time w_j^a of job j gets fixed at the task assignment by action a. For this reason, the value function (20) of the holding costs has been defined to consider only the jobs arriving in the future, whereas the holding cost of the current job j is taken care of by the immediate cost, $(w_j^a + x_j^a) \cdot h_j$. In this section, we assume that the service time is independent of the server, $x_j^a = x_j$, (i.e., the servers are equally fast) and thus $x_j \cdot h_j$ is a constant term for all actions and can be ignored. Utilizing (20) gives the expected increase in the holding costs due to action a is

$$c_{\mathrm{F}}(a) = w_j^a \cdot h_j + v_{\mathrm{F}}(\mathbf{z} \oplus a) - v_{\mathrm{F}}(\mathbf{z}),$$

where $\mathbf{z}$ is the current state of the system, $\mathbf{z} \oplus a$ the state after action a, and $v_{\mathrm{F}}(\mathbf{z}) = \tilde{v}(\mathbf{z})$ is the value function with respect to the waiting time (18).

With LCFS, the sojourn time of each job depends on the later arriving jobs and the value function (26) includes also the holding costs the present jobs incur. Therefore, there is no immediate holding cost and the expected increase in the holding costs due to action a is

$$c_{\mathrm{L}}(a) = v_{\mathrm{L}}(\mathbf{z} \oplus a) - v_{\mathrm{L}}(\mathbf{z}),$$

where $v_{\mathrm{L}}(\mathbf{z}) = v(\mathbf{z})$ given by (26).

The improved policy α' according to the policy iteration algorithm chooses the action that minimizes the expected increase in the total costs,

$$\alpha'(j, \mathbf{z}) = \operatorname*{argmin}_{a \in \mathcal{A}} \left(c_{\mathrm{S}}(a) + c_{\mathrm{W}}(a)\right) \tag{27}$$

where $\mathcal{A}$ denotes the set of possible actions and $c_{\mathrm{W}}(a)$ is either $c_{\mathrm{F}}(a)$ or $c_{\mathrm{L}}(a)$ depending on the scheduling discipline in the given queue. We will utilize (27) in the following numerical examples repeatedly to derive efficient state- and cost-aware FPI policies in different settings.

5.2. Numerical examples with FCFS

5.2.1. Two policy iteration steps for FCFS servers with constant service

As RR/LWL is optimal with respect to the mean delay for M/D/m, let us consider the same system but with arbitrary job-specific holding cost rates. Let Δ denote the constant service time of all job and h the holding cost rate of the new job. If $h > \mathrm{E}[H]$, the intuition suggests that the new job should be assigned to the shorter queue according to RR/LWL. However, if $h < \mathrm{E}[H]$, it may be beneficial to assign the new job to the longer queue, thus keeping the other queue shorter for later arriving, possibly more important, jobs. The potential pitfall is that no such "important" job arrives and one of the servers is unnecessarily idle (which never happens with RR/LWL). For policy improvement, it is more convenient to consider the waiting time based cost structure and (27).

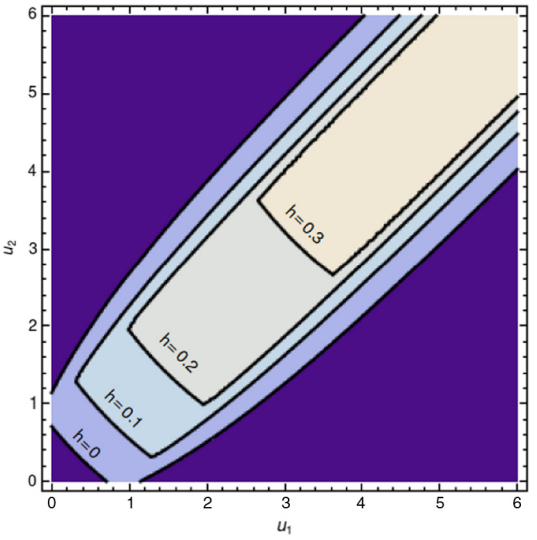


Fig. 6. States in the diagonal where SPI suggests assigning the new job with holding cost $h < \mathrm{E}[H] = 1$ to the longer queue.

Let us start with the static *Bernoulli-split policy* (RND). With two identical FCFS servers, this policy assigns the new job to Server 1 with probability of 0.5, and otherwise to Server 2. The value function of the whole system is [24]

$$\tilde{v}_{\mathrm{RND}}(u_1, u_2) = \frac{\lambda' \,\mathrm{E}[H]}{2(1-\rho')}(u_1^2 + u_2^2),$$

where λ' is the queue-specific arrival rate, $\lambda' = \lambda/2$, and ρ' the queue-specific load, $\rho' = \lambda/2 \cdot \Delta$. The mean difference in the expected costs between assigning the new job with holding cost h to Server 1 and Server 2 is

$$\begin{aligned}\Delta c &= h(u_1 - u_2) + \tilde{v}_{\mathrm{RND}}(u_1 + \Delta, u_2) - \tilde{v}_{\mathrm{RND}}(u_1, u_2 + \Delta) \\ &= \left(h + \frac{\lambda\Delta\,\mathrm{E}[H]}{2 - \lambda\Delta}\right) \cdot (u_1 - u_2),\end{aligned}$$

i.e., Δc is negative when $u_1 < u_2$, and vice versa. This means that the *first policy iteration* (FPI) step (27), choosing the action with the lowest expected overall cost if the consecutive decisions are according the basic policy (Bernoulli-split), yields LWL. Moreover, we recall that LWL was equivalent to RR with a constant service time Δ.

As the first policy iteration step yielded RR, the value function of which we can now compute, we can proceed further and carry out the *second policy iteration step* (SPI) for the M/D/m-RND/FCFS system,

$$\mathrm{RND} \overset{PI}{\Longrightarrow} \mathrm{LWL} \overset{PI}{\Longrightarrow} \mathrm{SPI}.$$

Fig. 6 illustrates the regions in the state space where SPI chooses the alternative action, i.e., assigns the new job with $h < \mathrm{E}[H] = 1$ to the longer queue. The arrival rate λ was chosen to be 0.5. We note that SPI changes the FPI policy (RR) only near the diagonal where both queues have roughly equal unfinished work. The closer the holding cost is to the mean, the higher the backlogs must be before the change, on average, pays off. Jobs with $h \geq \mathrm{E}[H]$ are categorically assigned to the shorter queue.

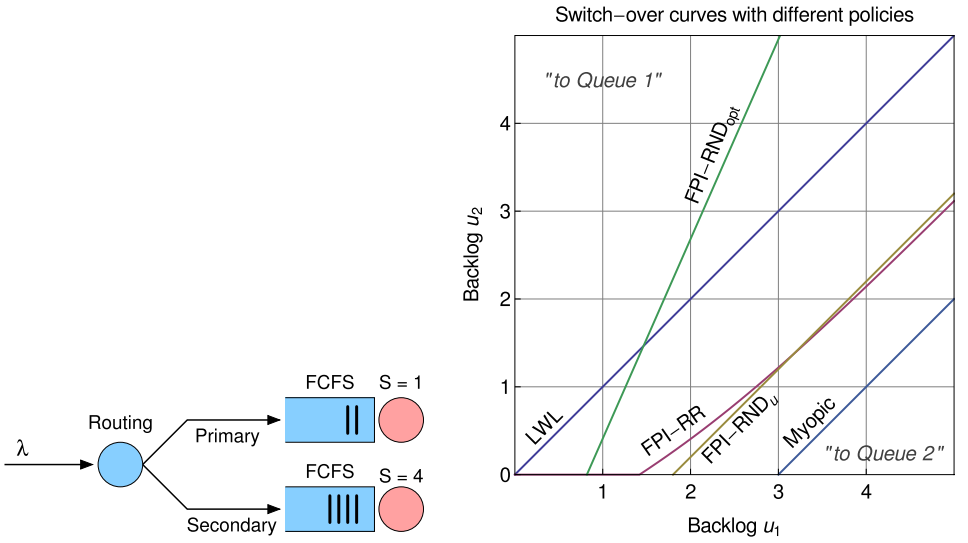
5.2.2. FCFS servers with constant service times and varying service fees

Let us next consider a server system with a primary and secondary server with fixed-size jobs illustrated in Fig. 7. The servers are equally fast, i.e., the service time of a job is the same in both queues. The cost structure is

$$H = 1, \quad \text{and} \quad S_1 = 1, \qquad S_2 = 4,$$

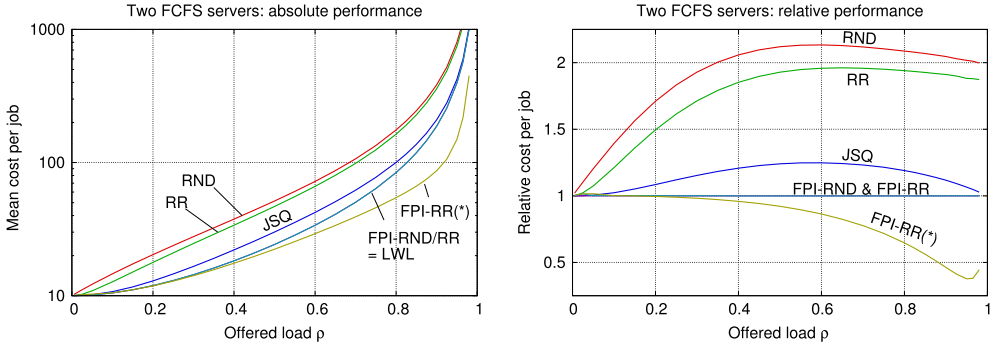
i.e., the secondary server has a four times higher service fee. Note that with identical service fees, LWL/RR/JSQ minimize the mean waiting and sojourn times.

With two servers, both LWL and Myopic belong to the class of the so-called switch-over policies, which are defined by a curve $f(u_1)$ such that a new job is routed to Queue 2 if $u_2 < f(u_1)$, and otherwise to Queue 1. It turns out that also the FPI policies based on RND and RR yield a switch-over policy. Fig. 7(b) illustrates the switch-over curves for $\lambda = 0.8$. RND_{opt} uses the optimal splitting probabilities, and RND_u splits the jobs equally, $p_1 = p_2 = 0.5$. We note that the curves for FPI-RND



(a) System with FCFS servers. (b) Routing policies.

Fig. 7. (a) Primary and secondary servers with unit holding cost $H = 1$ and service fees $S_1 = 1$ and $S_2 = 4$ processing jobs with constant service time. (b) Dynamic switch-over policies illustrated for a system with a constant service time, and two equally fast servers with service fees $S_1 = 1$ and $S_2 = 4$. Each policy assigns a new job to Queue 1 when the current state is above the corresponding curve, and otherwise to Queue 2.



(a) Absolute performance. (b) Relative performance.

Fig. 8. Mean holding costs in the elementary example setting with two identical servers. FPI-RR(*) achieves clearly the lowest cost rate.

policies are straight lines, while the FPI-RR switch-over curve is a slowly turning curve. Simulating the system gives the mean costs per job:

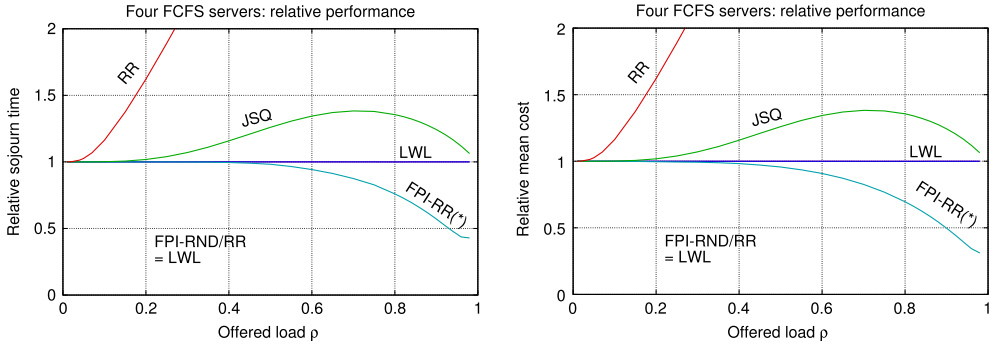
LWL:	2.20	FPI-RND$_{opt}$: 1.92	FPI-RR: 1.88
Myopic:	2.12	FPI-RND$_u$: 1.91	

We observe that FPI-RR achieves the lowest mean cost rate, closely followed by the other two FPI policies. Numerically experimenting one can see that when λ approaches 2 (the stability bound for this system), all FPI policies converge to LWL. Similarly, when $\lambda \to 0$, Myopic is optimal and all three FPI policies reduce to it.

5.2.3. *FCFS servers with random service times and holding costs*

Let us now consider an elementary system comprising two identical servers with random service times. Jobs arrive according to a Poisson process with rate λ. Job sizes and holding costs are i.i.d. random variables. The job size is 1 with probability of 0.9, and otherwise 91, so that $\mathrm{E}[X] = 10$ and $\mathrm{V}[X] = 629$. The holding costs are assumed to obey an exponential distribution with unit mean, $H \sim \mathrm{Exp}(1)$. With identical servers, the optimal RND policy splits the jobs equally, $p_k = 1/m$. LWL minimizes the holding cost of the current job, i.e., it makes the same greedy decision as selfish users. Moreover, both Myopic and FPI-RND reduce to LWL.

Simulation results are depicted in Fig. 8 for Bernoulli-split (RND), JSQ, RR, FPI-RND (i.e., LWL), FPI-RR, and FPI-RR(*) which we will describe later. The offered load ρ is on the x-axis, and the y-axis corresponds to the mean costs incurred per job. The left figure shows the absolute performance in logarithmic scale, and the right figure the performance relative to LWL,



(a) Sojourn time. (b) Holding costs.

Fig. 9. Mean holding costs in the elementary example setting with four identical servers. FPI-RR and FPI-RND are in practice identical to LWL, whereas the heuristic FPI-RR(*) yields a significantly lower mean sojourn time ($H \equiv 1$) and cost rate ($H \sim \text{Exp}(1)$).

$\text{E}[HT]/\text{E}[HT \mid \text{LWL}]$. The static RND and RR are the weakest routing policies. When $\rho \approx 0$, both LWL and JSQ work well. The task there is merely to avoid situations where two or more jobs are in one server while the other server is idle. As ρ increases beyond about 0.1, the performance of JSQ starts to degrade. As $\rho \to 1$, JSQ and LWL appear to converge to the same point.

FPI-RR is as good as LWL (and FPI-RND). This is due to the fact that both RND and RR are blind to holding costs, and the value of the state u is proportional to u^2. Therefore, the decision of LWL (i) minimizes the costs the current job incurs, and (ii) the expected increase in the costs incurred in the future. A closer inspection shows that FPI-RR is marginally better than LWL and FPI-RND.

However, all the aforementioned routing policies are non-optimal as the new heuristic, FPI-RR(*), clearly achieves a significantly lower mean cost rate. This heuristic policy was discovered while experimenting with the parameters of the value function computation. Instead of the correct value function, FPI-RR(*) uses the value functions obtained with (i) a too short interpolation interval ($u_{\max} = 90$) and (ii) linear extrapolation. As a consequence, a long backlog u is considered to be less harmful for future arrivals than it would be if the later arriving jobs were actually assigned according to RR. At the same time, FPI-RR(*), like FPI-RR, also makes a tentative decision as to where the next job will be assigned to. The performance of FPI-RR(*) is surprisingly strong and it achieves over 50% reduction in the costs under a high load, relative to LWL.

Fig. 9 depicts the corresponding results with four servers. The heuristic FPI-RR(*) is again very good, whereas FPI-RND and FPI-RR remain essentially equal to LWL. This suggests that FPI-RR(*) "understands" the situation better and the corresponding "value function" can be expected to be closer to the one of the optimal routing policy. Why this happens and how it can be utilized in this or other contexts is an interesting direction for future work.

5.3. Numerical examples with LCFS

Let us consider next systems of parallel LCFS servers, where the servers are again assumed to be equally fast. First we note that, as with FCFS, it is possible to carry out two policy iteration steps when service times are constants:

$$\text{RND} \overset{PI}{\Longrightarrow} \text{JSQ} \overset{(ties)}{\Longrightarrow} \text{RR} \overset{PI}{\Longrightarrow} \text{SPI}, \tag{28}$$

where SPI is the same as FPI-RR.

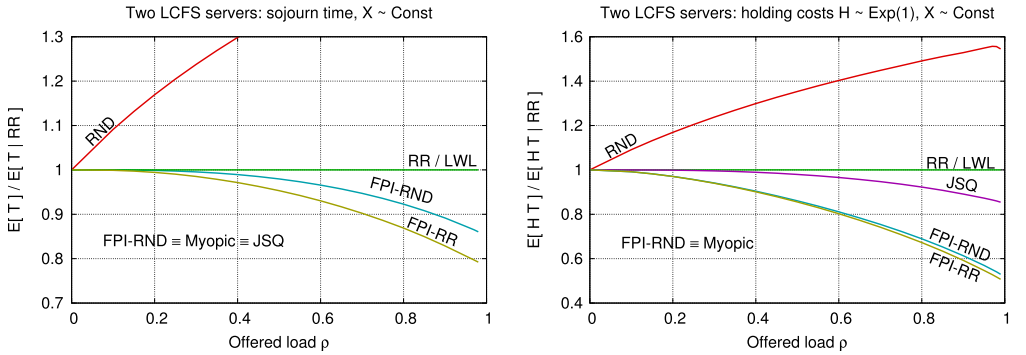
Let us consider next the performance of the different dispatching policies in a system of two parallel LCFS servers and jobs arriving according to the Poisson process with rate λ. We consider two objectives: (a) minimizing the mean sojourn time ($H \equiv 1$), and (b) minimizing the mean holding cost, when $H \sim \text{Exp}(1)$.

Suppose first that service times are constant. In this case, some of the routing policies become equivalent:

- RR and LWL, both ignoring the holding cost parameter, are equivalent, because the server that received the latest job has never a shorter backlog than the other server.
- Also FPI-RND reduces to JSQ when minimizing the sojourn time (see Eq. (28)). These policies are also equivalent to the Myopic routing policy that minimizes the cost under the condition that no other jobs arrive.

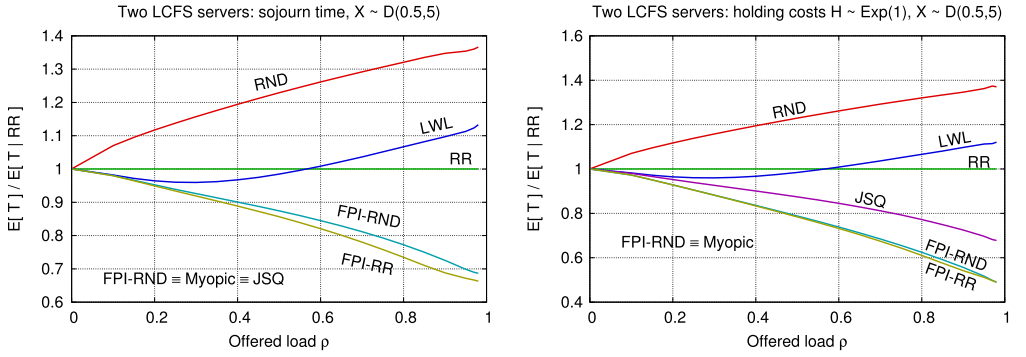
Recall that for FPI-RND, we consider all possible job assignments, which in this case is to Server 1 or Server 2. In contrast, for FPI-RR we consider all combinations of (i) possible job assignments and (ii) RR phases after the assignment ($m!$ combinations in general, two in our case). Hence, for each assignment, FPI-RR evaluates the value function (26) of the whole system in $2 \times 2 = 4$ states, and chooses the action that corresponds to a state with the lowest (relative) value. Note that the decision on phase essentially defines the server, where the job arriving next is (tentatively) assigned to.

Fig. 10(a) shows the mean sojourn time with different routing policies in comparison to RR. FPI-RR yields the lowest mean sojourn time, followed by FPI-RND and JSQ. Fig. 10(b) depicts the corresponding results with the mean holding cost



(a) Sojourn time. (b) Holding costs.

Fig. 10. Two identical LCFS servers and constant job size: (a) Relative mean sojourn time, and (b) relative mean holding costs.



(a) Sojourn time. (b) Holding costs.

Fig. 11. Two LCFS servers and service time distribution $D(0.5, 5.5)$: (a) Relative mean sojourn time, and (b) relative mean holding costs.

$H \sim \text{Exp}(1)$. In this case, FPI-RND is no longer identical to JSQ. Both FPI policies, being aware of the cost structure, incur significantly less costs than other policies. Among them, FPI-RR is marginally better.

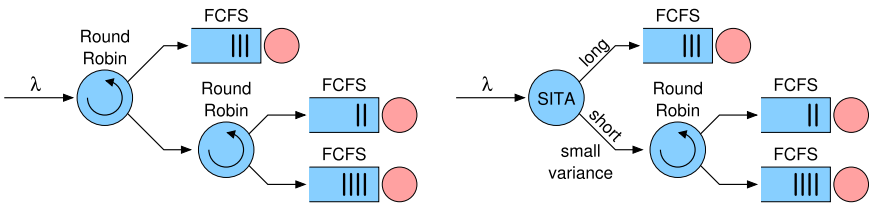
Consider next a non-constant service time distribution $D(0.5, 5.5)$, so that $\text{E}[X] = 1$ and $\text{V}[X] = 2$. Fig. 11(a) depicts the results when the objective is to minimize the mean sojourn time. Also in this case, FPI-RND and Myopic reduce to JSQ as long as the ties are resolved in the same way. On the other hand, LWL is no longer equivalent to RR and its performance in fact deteriorates substantially when ρ increases. Otherwise the results are similar as with the fixed service time. The dynamic JSQ/Myopic/FPI-RND policy does a rather good job, but FPI-RR yields an even lower mean sojourn time. Fig. 11(b) depicts the corresponding results with the exponential holding cost distribution, $H \sim \text{Exp}(1)$. The results are similar as before and, e.g., the performance of FPI-RND and FPI-RR are alike.

6. More advanced applications

The ability to determine a value function for $\text{Erl}(m, \lambda)/\text{G}/1$ queues enables the analysis of far more complex server systems than the plain Round-Robin system. One example is illustrated in Fig. 12(a), where a multi-layer RR is constructed: Queue 1 behaves according to $\text{Erl}(2, \lambda)/\text{G}/1$ and Queues 2 and 3 according to $\text{Erl}(4, \lambda)/\text{G}/1$. This type of arrangement can be advantageous when the service rates and/or operating costs are asymmetric.

The main strength in RR comes from the fact that it reduces the variability in the inter-arrival times (see Sections 3.1.6 and 4.3.1). On the other hand, a high variability in the job sizes can be equally harmful (due to the second moment in the Pollaczek–Khinchine formula for the mean waiting time). The so-called *Size-Interval-Task-Assignment* (SITA) policy [33,15,34,35] seeks to reduces the variability in the job sizes by assigning jobs with a similar size to the same queue. To this end, the support of the job sizes is divided into m non-overlapping intervals $[\xi_i, \xi_{i+1})$, $i = 1, \ldots, m$, and a job with size x is assigned to Server i iff $x \in [\xi_i, \xi_{i+1})$.

Fig. 12(b) illustrates a server system which combines the useful features of SITA (variance reduction in service times) and RR (which was the optimal policy w.r.t. delay for tasks with a constant service time). The arrival process to Queue 1 is a Poisson process as SITA is a state-independent policy. Queues 2 and 3 behave according to $\text{Erl}(2, \lambda')/\text{G}/1$, where the service time of tasks can have a significantly smaller variance thanks to SITA. Moreover, dedicated jobs arriving according to some other Poisson process can be directed to any point already receiving a Poisson process, such as Queue 1 and the second level



(a) Multi-layer RR. (b) Hybrid routing.

Fig. 12. (a) Multi-layer Round-Robin system feeds tasks to each queue with Erlang-distributed inter-arrival times. (b) In the hybrid system, Queue 1 behaves according to M/G/1 and Queues 2 and 3 according to Erl$(2, \lambda')$/G/1 with reduced variance in the service times.

RR dispatcher in Fig. 12(b). Thus, the analysis of systems that hierarchically combine dispatchers remains tractable, their value function can be determined, and the policy improvement step can be carried out.

7. Conclusions

We have analyzed the Round-Robin (RR) routing to a system of parallel queues. RR is a commonly used robust technique to balance the load by assigning tasks to different servers sequentially. It decreases the burstiness in the arrival process to each queue, which is important especially when the queues process the jobs in the FCFS order.

The availability of the value function for RR systems under FCFS and LCFS scheduling, via the corresponding value function of Erl(m, λ)/G/1 queues, provides new insight to this mechanism itself (and to G/G/1 queues). The value functions that we considered characterize the system state with respect to the service fees and (virtual) waiting time, and also enable the policy iteration step with respect to a very versatile cost structure defined by job- and server-specific service fees and holding costs, yielding robust cost- and state-aware routing policies. Even though the value functions for RR systems are computationally more demanding than the corresponding expressions for Bernoulli splitting, they can be evaluated numerically in online fashion, thus facilitating efficient routing policies. Moreover, as a useful side-product, we obtain the mean waiting time in the Round-Robin systems with FCFS and LCFS scheduling, which itself is a non-trivial result even for an M/D/m-RR queue.

Acknowledgments

This work was supported by the Academy of Finland in the TOP-Energy project (grant no. 268992) and FQ4BD project (grant no. 296206).

References

[1] E. Hyytiä, R. Righter, S. Aalto, Energy-aware job assignment in server farms with setup delays under LCFS and PS, in: 26th International Teletraffic Congress, ITC'26, Karlskrona, Sweden, 2014.
[2] W. Winston, Optimality of the shortest line discipline, J. Appl. Probab. 14 (1977) 181–189.
[3] R.R. Weber, On the optimal assignment of customers to parallel servers, J. Appl. Probab. 15 (2) (1978) 406–413.
[4] A. Ephremides, P. Varaiya, J. Walrand, A simple dynamic routing problem, IEEE Trans. Autom. Controlc 25 (4) (1980) 690–693.
[5] A. Hordijk, G. Koole, On the optimality of the generalised shortest queue policy, Probab. Engrg. Inform. Sci. 4 (1990) 477–487.
[6] D. Towsley, P. Sparaggis, C. Cassandras, Stochastic ordering properties and optimal routing control for a class of finite capacity queueing systems, in: Proc. of the 29th IEEE Conference on Decision and Control, 1990, pp. 658–663.
[7] P.D. Sparaggis, D. Towsley, Optimal Routing and Scheduling of Customers with Deadlines, Probab. Engrg. Inform. Sci. 8 (1) (1994) 33–49.
[8] G. Koole, P.D. Sparaggis, D. Towsley, Minimizing response times and queue lengths in systems of parallel queues, J. Appl. Probab. 36 (4) (1999) 1185–1193.
[9] O. Akgun, R. Righter, R. Wolff, Multiple server system with flexible arrivals, Adv. Appl. Probab. 43 (2011) 985–1004.
[10] Z. Liu, D. Towsley, Optimality of the round-robin routing policy, J. Appl. Probab. 31 (2) (1994) 466–475.
[11] Z. Liu, R. Righter, Optimal load balancing on distributed homogeneous unreliable processors, Oper. Res. 46 (4) (1998) 563–573.
[12] D. Down, R. Wu, Multi-layered round robin routing for parallel servers, Queueing Syst. 53 (4) (2006) 177–188.
[13] M. Harchol-Balter, Performance Modeling and Design of Computer Systems: Queueing Theory in Action, Cambridge University Press, 2013.
[14] E. Hyytiä, S. Aalto, Round-robin routing policy: value functions and mean performance with job- and server-specific costs, in: 7th International Conference on Performance Evaluation Methodologies and Tools, ValueTools, Torino, Italy, 2013.
[15] M. Harchol-Balter, M.E. Crovella, C.D. Murta, On choosing a task assignment policy for a distributed server system, J. Parallel Distrib. Comput. 59 (1999) 204–228.
[16] C.D. Crommelin, Delay probability formulas when the holding times are constant, Post Off. Electr. Eng. J. 25 (1932) 41–50.
[17] G.J. Franx, A simple solution for the M/D/c waiting time distribution, Oper. Res. Lett. 29 (5) (2001) 221–229.
[18] G.J. Franx, The transient M/D/c queueing system, 2002.
[19] H. Tijms, New and old results for the M/D/c queue, AEU-Int. J. Electron. Commun. 60 (2) (2006) 125–130.
[20] A.J.E.M. Janssen, J.S.H. Van Leeuwaarden, Back to the roots of the M/D/s queue and the works of Erlang, Crommelin and Pollaczek, Stat. Neerl. 62 (3) (2008) 299–313.
[21] K.R. Krishnan, Joining the right queue: a state-dependent decision rule, IEEE Trans. Autom. Control 35 (1) (1990) 104–108.
[22] P.S. Ansell, K.D. Glazebrook, C. Kirkbride, Generalised 'Join the Shortest Queue' policies for the dynamic routing of jobs to multi-class queues, J. Oper. Res. Soc. 54 (4) (2003) 379–389.
[23] S. Bhulai, On the value function of the M/Cox(r)/1 queue, J. Appl. Probab. 43 (2) (2006) 363–376.
[24] E. Hyytiä, A. Penttinen, S. Aalto, Size- and state-aware dispatching problem with queue-specific job sizes, European J. Oper. Res. 217 (2) (2012) 357–370.
[25] R.A. Howard, Dynamic Probabilistic Systems, Volume II: Semi-Markov and Decision Processes, Wiley Interscience, 1971.
[26] B. Hajek, The proof of a folk theorem on queuing delay with applications to routing in networks, J. ACM 30 (4) (1983) 834–851.

[27] B. Hajek, Extremal splittings of point processes, Math. Oper. Res. 10 (4) (1985) 543–556.
[28] Y. Arian, Y. Levy, Algorithms for generalized round robin routing, Oper. Res. Lett. 12 (5) (1992) 313–319.
[29] E. Hyytiä, S. Aalto, A. Penttinen, Minimizing slowdown in heterogeneous size-aware dispatching systems, ACM SIGMETRICS Perform. Eval. Rev. 40 (2012) 29–40. (ACM SIGMETRICS/Performance conference).
[30] L. Kleinrock, Queueing Systems, Volume I: Theory, Wiley Interscience, 1975.
[31] R. Bellman, Dynamic Programming, Princeton University Press, 1957.
[32] M.L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming, Wiley, 2005.
[33] M.E. Crovella, M. Harchol-Balter, C.D. Murta, Task assignment in a distributed system: Improving performance by unbalancing load, in: Proceedings of SIGMETRICS '98, Madison, Wisconsin, USA, 1998, pp. 268–269.
[34] H. Feng, V. Misra, D. Rubenstein, Optimal state-free, size-aware dispatching for heterogeneous M/G/-type systems, Perform. Eval. 62 (1–4) (2005) 475–492.
[35] M. Harchol-Balter, A. Scheller-Wolf, A.R. Young, Surprising results on task assignment in server farms with high-variability workloads, in: Proc. of SIGMETRICS, ACM, New York, NY, USA, 2009, pp. 287–298.

Esa Hyytiä received the M.Sc. (Tech.) degree in engineering physics and Dr.Sc. (Tech.) degree in electrical engineering from Helsinki University of Technology, in 1998 and 2004, respectively. In 2013, he was awarded a docentship in performance analysis of communication networks at the Aalto University School of Electrical Engineering. In 1997, he joined the Laboratory of Telecommunications of Helsinki University of Technology (TKK). From 2005 to 2006, he was with the Norwegian University of Science and Technology (NTNU), Norway as a postdoc researcher, from 2005 to 2009, with the Telecommunication Research Center Vienna (FTW), Austria, as a senior researcher, and from 2009 to 2015, with Aalto University, Finland, as a research fellow. Currently he is working at the Department of Computer Science of the University of Iceland as an assistant professor. His research interests include performance analysis, modelling and optimization of computer and communications systems.

Samuli Aalto received his M.Sc. and Ph.D. degrees in Mathematics from the University of Helsinki in 1984 and 1998, respectively. From 1984 to 1997, Dr. Aalto worked as a Research Scientist at VTT Technical Research Center of Finland. Since 1997, he has been with TKK Helsinki University of Technology, which is now part of Aalto University. Currently he acts as Senior Research Fellow leading the Teletraffic and Performance Analysis Group in the Department of Communications and Networking. Dr. Aalto's research interests include queueing theory, teletraffic theory, and performance analysis of modern communications systems and networks.