Energy-aware Job Assignment in Server Farms with Setup Delays under LCFS and PS

Esa Hyytiä
Department of Communications and Networking
Aalto University, Finland

Rhonda Righter
Department of Industrial Engineering and Operations Research
University of California Berkeley

Samuli Aalto
Department of Communications and Networking
Aalto University, Finland

***Abstract*—We consider the job (or task) assignment problem to heterogeneous parallel servers, where servers can be switched off to save energy. However, switching a server back on involves a constant server-specific delay. We will use one step of policy iteration from a starting policy such as Bernoulli splitting, in order to derive efficient job assignment (dispatching) policies that minimize the long-run average cost. In our earlier work, we assumed FCFS scheduling at the servers. In this paper, we focus on LCFS and PS, where the latter in particular serves as an important model, e.g., for server farms where jobs are processed concurrently. LCFS, on the other hand, is a robust policy that works especially well when service times are highly variable, and is easier to implement than the standard alternative policy for high variability, LAS (least attained service time). Our costs include energy related running costs, as well as performance related mean response times. The efficiency of the resulting operating policies is illustrated with numerical examples, where we also compare FCFS to LCFS and PS.**

***Index Terms*—setup delay, routing, insensitivity, LCFS, PS**

I. Introduction

Many Internet services are now provided by data centers and (web) server farms, and their performance and energy consumption are receiving increasing attention. A straightforward way to control the energy consumption is to switch off idle servers during the off peak periods when they are not needed. However, switching them back on typically takes time and thus it may not be cost effective to switch a server off whenever one becomes idle.

In this paper, we consider an elementary model for a server farm comprising k parallel servers. The scheduling discipline of jobs in each server is often assumed to be first-come-first-served (FCFS). In contrast, we consider (preemptive) last-come-first-served (LCFS) and processor sharing (PS) [1], which are insensitive to the service time distribution for standard M/G/1 queues. In particular, LCFS is a robust policy that works especially well when service times are highly variable, and is easier to implement than the standard alternative policy for high variability, least-attained-service-time (LAS).

Before considering the complete system of parallel queues, we first study a setup delay in single-server queues. Our aim is to derive the value functions with respect to the response time (sojourn time). This was done for M/G/1-FCFS in [2]. We focus first on M/G/1-LCFS, and then we move to the processor sharing (PS) scheduling discipline and study M/D/1-PS, because M/G/1-PS is beyond our means. Additionally, we show that the mean response time in an M/G/1-PS is no longer insensitive to the service time distribution when a setup delay is introduced, whereas with LCFS the insensitivity property is preserved. The insensitivity property is generally desired as it enhances robustness [3], [4].

We then utilize the single-server results to assign jobs in a server farm modelled as a system of parallel queues. The optimal decisions take into account both the (expected) response times as well as the energy consumption. Additional complications arise from the setup delays when some server has been switched off in order to preserve energy.

The main contributions of this work are as follows:

1) We derive the mean response time for M/G/1-LCFS and M/D/1-PS queues with random setup delays S, and consequently, we show that the mean response time with LCFS remains insensitive to the job size distribution, whereas with PS the setup delay breaks the widely celebrated insensitivity property of PS.
2) We derive value functions for the size-aware M/G/1-LCFS and M/D/1-PS queues subject to setup delays, which enable efficient job assignment policies.
3) We also derive the so-called Lookahead policy for LCFS with setup delays, which outperforms all reference policies in the numerical examples.

We also evaluate example server farms under different scheduling policies and experiment with the variability in the job sizes.

A. Related work

In the basic problem, jobs are processed in FCFS order, there is no setup delay and one is interested in minimizing the mean response time. Under different settings, *Join-the-Shortest-Queue* (JSQ) has been shown to be optimal, e.g., in [5], [6], *Least-Work-Left* (LWL) in [7]–[10], *Round-Robin* (RR) in [11]–[13], and *Size-Interval-Task-Assignment* (SITA) in [14]. Such simple policies are no longer optimal when there is a setup delay. The job assignment problem has been approached also within the Markov decision problem (MDP) framework, see, e.g., [15]–[17]. The idea is to start with an arbitrary static policy, and then carry out the first policy iteration (FPI) step.

Results with setup delays are scarce. Artalejo et. al [18] give steady-state results for an M/M/k, where at most one server can be in an exponentially distributed setup phase. Gandhi

978-0-9836283-9-2 © 2014 ITC

and Harchol-Balter [19] consider a similar M/G/k with an exponential setup delay. In [20], [21], Gandhi et. al consider an M/M/k with different power saving policies. Maccio and Down [22] study different switching on/off extensions to an M/G/1 and then apply random routing.

II. Preliminaries

A. Arrival process and model for the server farm

We assume that jobs arrive according to a Poisson process with rate λ. The job sizes are i.i.d. random variables, $X_j \sim X$. Jobs are processed by a system of k parallel servers. Each server has its own queue, and new jobs are assigned upon arrival. Server i processes the jobs at rate ν_i according to a local scheduling rule (e.g., LCFS, or PS).

We assume that Server i incurs energy costs at rate e_i whenever it is running, either processing jobs or in the setup phase. When the server is idle and switched off, the cost rate is zero. When a server becomes idle, it is switched off to preserve energy. However, it takes time s_i to switch it back on before it can start processing jobs. (In Section V-B we consider systems where some servers are never switched off).

Next we review some earlier results for M/G/1 queues that are necessary for our analysis in Sections III and IV. In the next few sections we consider only one queue so we drop the subscript i.

B. Busy periods in M/G/1

Our primary interest in this paper is server systems with a fixed known setup delay s. However, some results we give in more general form for arbitrary i.i.d. random setup delays.

Consider first an arbitrary M/G/1 queue with i.i.d. setup delays $S_i \sim S$. That is, the first arrival to an empty queue initiates a setup phase with a random duration S before the service starts. Evolution of such a system consists of three phases, (i) an exponentially distributed idle time with mean $1/\lambda$, (ii) a setup phase with length S, and (iii) the working phase of length B; the cycle of three phases repeats sequentially as illustrated in Fig. 1, which shows a sample path of the backlog process. An empty system corresponds to a renewal point, and B clearly depends on S. The mean duration of the whole cycle is

$$\mathrm{E}[\ell] := 1/\lambda + \mathrm{E}[S] + \mathrm{E}[B].$$

Moreover, the fraction of time a stable system works is equal to the offered load $\rho = \lambda \cdot \mathrm{E}[X]$, i.e.,

$$\frac{\mathrm{E}[B]}{1/\lambda + \mathrm{E}[S] + \mathrm{E}[B]} = \rho,$$

and solving for $\mathrm{E}[B]$ gives

$$\mathrm{E}[B] = \frac{\mathrm{E}[X]}{1-\rho} + \frac{\rho}{1-\rho} \cdot \mathrm{E}[S] = \frac{(1+\lambda \mathrm{E}[S])\mathrm{E}[X]}{1-\rho}, \quad (1)$$

and

$$\mathrm{E}[\ell] = \frac{1/\lambda + \mathrm{E}[S]}{1-\rho}. \quad (2)$$

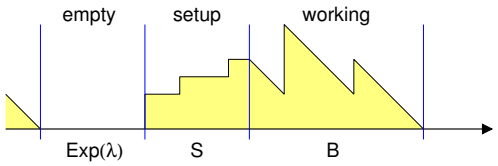

Fig. 1. *One busy cycle consists of three phases: an empty system, a setup phase of s, and a working phase during which jobs are actually processed.*

Note that the above holds for any work conserving scheduling discipline. The mean duration of the busy period including the initial setup phase is

$$\mathrm{E}[B^*] = \mathrm{E}[S] + \mathrm{E}[B] = \frac{\mathrm{E}[S] + \mathrm{E}[X]}{1-\rho},$$

whereas the mean number of jobs served in a busy period is

$$\mathrm{E}[N_B] = \frac{1 + \lambda \mathrm{E}[S]}{1-\rho}.$$

The probability that an arriving job finds the system empty and starts a setup period is $(1/\lambda)/\mathrm{E}[\ell]$ (PASTA), which gives

$$\mathrm{P}\{\text{empty}\} = \frac{1-\rho}{1+\lambda \mathrm{E}[S]} = \frac{1}{\mathrm{E}[N_B]},$$

where the latter is due to the fact that exactly one job per busy period sees the system empty.

C. State description

Let z be the current (size-aware) state of the system, $z = (\delta; x_1, \ldots, x_n)$, where δ is the remaining setup delay and the x_i the remaining service times of the n jobs. The ordering of the jobs in z depends on the scheduling discipline:

- With LCFS, Job 1 receives service first and Job n last.
- With PS, $x_{i+1} \le x_i$; Job n departs first and Job 1 last.

This choice is for convenience of notation. We let $z \oplus x$ denote the state after an arrival of a job with size x. The *virtual backlog*, denoted by u, is the time until the system becomes idle given no additional jobs arrive,

$$u = \delta + x_1 + \ldots + x_n,$$

i.e., u includes also the remaining setup delay, if any. Thus, the virtual backlog after an arrival is given by

$$u' = \begin{cases} u + x, & \text{if the server was running,} \quad (u > 0) \\ s + x, & \text{if the server was off.} \quad (u = 0) \end{cases}$$

D. Value functions and admission costs

The central notion in this paper is the value function,

$$v_z := \lim_{t \to \infty} \mathrm{E}[V_z(t) - r \cdot t], \quad (3)$$

where $V_z(t)$ denotes the costs incurred during $(0, t)$ when initially in state z, and r is the mean cost rate. We have implicitly assumed a stable and ergodic system ($\lambda \mathrm{E}[X] < 1$). Moreover, in this paper we consider size-aware state information. The important quantity is actually the difference, $v_{z'} - v_z$, which describes how much more or less it costs to start an ergodic system from state z' instead of state z.

E. First policy iteration and Lookahead

Policy improvement is a standard technique of Markov decision processes [23]. At every step, one first computes the value function for the current policy α_i, and the policy iteration step yields a new policy α_{i+1}. Under certain assumptions, α_i converges to the optimal policy.

In our case, actions correspond to job assignment decisions. We start from a static basic policy α_0, which assigns jobs independently of the state of the queues. With such policies, the system decomposes to k M/G/1 queues and the value function of the whole system is the sum of the queue-specific value functions,

$$v_z = \sum_i v^{(i)}_{z_i}.$$

The first policy iteration (FPI) step reduces to

$$\alpha_{\mathrm{FPI}}(z, x) = \arg\min_i \; a_i(z_i, x),$$

where $a_i(z_i, x)$ is the marginal *admission cost* to Queue i,

$$a_i(z_i, x) := v^{(i)}_{z_i \oplus x} - v^{(i)}_{z_i}. \tag{4}$$

In the following we often omit x and simply write $a(z)$. Note that v_z and $a(z)$ depend on the cost structure (response time, energy consumption, etc.).

With FPI, we deviate from the basic policy for a single action and estimate the future costs by assuming that the subsequent decisions are by the basic policy. With *Lookahead* (LH), we deviate for two consecutive actions; for the current job and (tentatively) for the next. Lookahead thus gives cost estimates for such decions as "this job to Queue 1 and the next (currently unknown) job to Queue 2". We return to LH in Section III-A. See also [24].

F. Running costs in M/G/1

The running costs depend only on the lengths of busy and idle periods, and not on the scheduling discipline as long as it is work conserving. The following results, given in [2] for a fixed setup delay s, hold for all the M/G/1 queues we study. Here we generalize the results for a random setup delay of S.

The mean running cost rate in a work conserving M/G/1 queue with setup delay S is

$$r_R = \frac{\rho + \lambda \,\mathrm{E}[S]}{1 + \lambda \,\mathrm{E}[S]} \cdot e. \tag{5}$$

The size-aware running cost value function for M/G/1 satisfies

$$v_R(u) - v_R(0) = \frac{u}{1 + \lambda \,\mathrm{E}[S]} \cdot e. \tag{6}$$

Using (4), the admission cost of a job with size x is

$$a_R(u, x) = \frac{x + 1(u = 0)\,\mathrm{E}[S]}{1 + \lambda \,\mathrm{E}[S]} \cdot e.$$

Note that even though here we assume the elementary cost structure of running costs only for energy consumption, we can include, e.g., switching costs and processing costs, along with our results on response times in the sequel, to develop more general energy-related cost structures for LCFS and PS models. We refer the interested reader to [2] for further details.

III. Preemptive LCFS with a Setup Delay

Let us next consider the response times for the preemptive LCFS scheduling discipline with a setup delay. First we suppose that the setup delays are i.i.d. random variables $S_i \sim S$.

Theorem 1 (Mean response time): The mean response time in an M/G/1 with preemptive LCFS is

$$\mathrm{E}[T] = \frac{\mathrm{E}[X]}{1 - \rho} + \frac{\mathrm{E}[S] + (\lambda/2)\,\mathrm{E}[S^2]}{1 + \lambda \,\mathrm{E}[S]}. \tag{7}$$

Proof: The conditional mean response time of an arbitrary job with size x arriving to a busy-idle-setup cycle with setup delay s is

$$\begin{aligned}\mathrm{E}[T|x, s] &= \frac{1/\lambda}{\mathrm{E}[\ell]} \cdot \frac{x + s}{1 - \rho} + \frac{s}{\mathrm{E}[\ell]} \int_0^s \frac{1}{s} \cdot \frac{x + \delta}{1 - \rho}\, d\delta + \frac{\mathrm{E}[B]}{\mathrm{E}[\ell]} \cdot \frac{x}{1 - \rho} \\ &= \frac{x}{1 - \rho} + \frac{s(2 + \lambda s)}{2(1 + \lambda s)},\end{aligned}$$

where ℓ is the busy-idle-setup cycle length. Thus $\mathrm{E}[T|s] = \mathrm{E}[\mathrm{E}[T|X, s]] = \frac{\mathrm{E}[X]}{1-\rho} + \frac{s(2+\lambda s)}{2(1+\lambda s)}$, and from PASTA,

$$\mathrm{E}[T] = \int \mathrm{E}[T|s]\, \pi(s)\, dF_S(s),$$

where F_S is the distribution function of S and $\pi(s)\, dF_S(s)$ is the proportion of time the system is in a busy-idle-setup cycle where the setup delay is s. From (2),

$$\pi(s) = \frac{\mathrm{E}[\ell|S = s]}{\mathrm{E}[\ell]} = \frac{1 + \lambda s}{1 + \lambda \,\mathrm{E}[S]},$$

which yields (7). ∎

The latter term in (7) corresponds to the penalty due to the setup delay, and is the same as the FCFS delay penalty (see, e.g., (8.3.64) in [25]), so we have:

Corollary 1 (Setup delay penalty): The penalty due to setup delay in LCFS is the same as with FCFS,

$$p_S := \frac{\mathrm{E}[S] + (\lambda/2)\,\mathrm{E}[S^2]}{1 + \lambda \,\mathrm{E}[S]}. \tag{8}$$

Corollary 2 (Insensitivity): The mean response time in M/G/1-LCFS with an i.i.d. setup delay is insensitive to the service time distribution.

The fact that the penalty due to a setup delay is the same for FCFS and LCFS makes one wonder if this is a general property that holds for an arbitrary scheduling discipline. Later, in Section IV, we will show that this is not the case. Note also that the setup delay penalty (8) varies between $\mathrm{E}[S^2]/(2\,\mathrm{E}[S])$ and $\mathrm{E}[S]$ as a function λ. With a fixed setup delay of s, the delay penalty for each job ranges from $s/2$ to s. In this case, regardless of λ, the increase in the mean response time is of order s. On the other hand, if S has high variance, the effect of the delay penalty *increases* with λ. Of course, as long as the variance is finite, the relative penalty (the cost rate due to setup delays) will still approach 0 as $\rho \to 1$ because of the long busy periods. In a system of parallel queues, we can minimize

the penalty by controlling the routing of jobs to modify the arrival process *to each queue*.[1]

Little's result then gives the mean number in the system, which will also be the mean cost rate with respect to the cumulative response time,

$$r := \mathrm{E}[N] = \lambda\,\mathrm{E}[T] = \frac{\rho}{1-\rho} + \frac{\lambda\,\mathrm{E}[S] + (\lambda^2/2)\,\mathrm{E}[S^2]}{1+\lambda\,\mathrm{E}[S]} =: r_0 + r_s, \tag{9}$$

where r_0 is the mean cost rate for an M/G/1 queue with no setup delay, and r_s is the additional mean cost rate due to the introduction of the setup delay S.

Next we limit ourselves to the case of deterministic setup delay $S \equiv s$. We let z denote the size-aware state, $z = (\delta; x_1, \ldots, x_n)$, where δ is the remaining setup delay and the x_i denote the remaining service times. Job 1 receives service first and Job n is at the end of the queue. Then

$$y_j := \delta + x_1 + \ldots + x_j$$

denotes the sum of job j's own (remaining) service time, the (remaining) service time of the jobs that have arrived after it and the remaining setup time, i.e., y_j is the (tentative) remaining response time of Job j. The mean remaining response time of Job j is $y_j/(1-\rho)$ and the virtual backlog is $u = y_n = \delta + x_1 + \ldots + x_n$. For the size-aware value function we have the following result:

Theorem 2 (Value function): The value function w.r.t. the response time in an M/G/1-LCFS with setup delay s is

$$v_z - v_0 = \sum_{i=1}^{n} \frac{y_i}{1-\rho} + \frac{\lambda\,\delta^2}{2(1-\rho)} - \frac{u\lambda s(2+\lambda s)}{2(1-\rho)(1+\lambda s)}. \tag{10}$$

Proof: The mean response-time cost rate is $r = \mathrm{E}[N] = r_0 + r_s$ given in (9). We assume response time costs are incurred continuously. Then

$$v_z = \mathrm{E}[V_z(R_z)] - (r_0 + r_s)\,\mathrm{E}[R_z] + v_0,$$

where R_z is the length of time until the system empties starting in state z, for which $\mathrm{E}[R_z] = u/(1-\rho)$, and $V_z(R_z)$ is the total cumulative response time cost incurred during R_z. For all jobs that arrive during R_z, once they enter service, they are unaffected by the setup delay, so their expected response times starting at time δ are the standard $M/G/1$ LCFS response times of $\mathrm{E}[X]/(1-\rho)$. That is,

$$\begin{aligned}\mathrm{E}[V_z(R_z)] &= \lambda\,\mathrm{E}[R_z]\,\mathrm{E}[X]/(1-\rho) + \mathrm{E}[V_z^0] + \mathrm{E}[V_z^\delta] \\ &= r_0\,\mathrm{E}[R_z] + \mathrm{E}[V_z^0] + \mathrm{E}[V_z^\delta]\end{aligned}$$

where V_z^0 is the total remaining response times for the n jobs initially present, and V_z^δ is the total *additional* response time that jobs arriving during δ experience *due to the setup delay*. The remaining mean response time for a job j that is initially present is $y_j/(1-\rho)$. Thus, $\mathrm{E}[V_z^0] = \sum_j y_j/(1-\rho)$. For a job that arrives at time t, $0 \le t \le \delta$, its expected response time increases by $(\delta - t)/(1-\rho)$, so

$$\mathrm{E}[V_z^\delta] = \int_0^\delta \frac{\lambda(\delta - t)}{1-\rho}\,dt = \frac{\lambda\delta^2}{2(1-\rho)}.$$

It follows that

$$\begin{aligned} v_z - v_0 &= \sum_{j=1}^{n} \frac{y_j}{1-\rho} + \frac{\lambda\delta^2}{2(1-\rho)} - r_s\,\mathrm{E}[R_z] \\ &= \sum_{j=1}^{n} \frac{y_j}{1-\rho} + \frac{\lambda\delta^2}{2(1-\rho)} - \frac{u\lambda s(2+\lambda s)}{(1-\rho)2(1+\lambda s)}. \end{aligned}$$

■

Note that alternatively (10) can be written using the x_i,

$$v_z - v_0 = \frac{1}{1-\rho}\left(\sum_{i=1}^{n} i\,x_{n-i+1} + n\delta + \frac{\lambda\,\delta^2}{2} - \frac{u\lambda s(2+\lambda s)}{2(1+\lambda s)}\right).$$

We note that the full setup delay s shows up in an additional term in (10). It is negative due to the fact that a large u means that the next setup period has been pushed far into the future.

Corollary 3: The (response time) admission cost of a job with size x to an LCFS queue is $v_{z\oplus x} - v_z$, which gives

$$a(x, n, \delta) = \frac{nx + \delta}{1-\rho} + \frac{(2 - (\lambda s)^2)x + 1(n{=}0)s(2+\lambda s)}{2(1-\rho)(1+\lambda s)}. \tag{11}$$

In particular, the admission cost depends only on the length of the remaining setup delay δ and the number in queue n, not on the backlog u. On one hand, this is a positive thing as the required state information is minimal, which improves robustness. But on the other hand, this is a negative feature as it means that the FPI policy does not utilize the available size information. For example, with identical servers having no setup delay, the FPI step gives JSQ, which may be optimal when only the queue occupation is available, but we assumed that the remaining service times are also known. We overcome this shortcoming in the next section by also considering explicitly the job arriving next.

A. Lookahead

In FPI, we deviate from the basic policy α_0 for one decision and assume that the consecutive decisions are all again by α_0. In our case, the basic policy is typically a static policy, i.e., its decisions do not depend on the current state of the system. This may obviously hinder the quality of the FPI policy. One approach to overcome this is Lookahead, where the next decision is also taken into account [2], [24].

1) Analysis for a single queue: We will define a lookahead cost for assigning a job with size X^* at time $T^* \sim \mathrm{Exp}(\lambda^*)$. We note that X^* may obey a different distribution than X, or it can be set explicitly to zero, which means that the given queue has an arrival free period of length T^* (i.e., the next job is assigned somewhere else). Similarly, λ^* can be different from λ and typically we have $\lambda^* > \lambda$ when the Lookahead cost is used to evaluate different dispatching decisions.

Definition 1:

$$L(z, \lambda^*, X^*) := \mathrm{E}[V_z(T^*) - r \cdot T^* + v_{Z^*\oplus X^*} - v_0],$$

[1]The effect of the routing policy can be seen in the following example. It is easy to see that under a suitable light traffic setting the Round-robin routing policy, though good for routing to queues without setups, can be the worst possible policy if setup delays are present and new jobs are assigned to servers that were just switched off.

where $V_z(T^*)$ denotes the costs incurred during time T^* and Z^* denotes the state of the queue at time T^* before the assignment of the job with size X^*.

Theorem 3: The lookahead cost in an M/G/1-LCFS with setup delay s when the next job with size X^* arrives at time $T^* \sim \mathrm{Exp}(\lambda^*)$ is

$$\begin{aligned} L(z,\lambda^*,X^*) &= \sum_{i=1}^{n} \frac{y_i}{1-\rho} + \frac{\left(n+1-\sum_{i=1}^{n} e^{-\lambda^* y_i}\right)(\rho^*-\rho)}{\lambda^*(1-\rho)} \\ &+ \frac{\lambda\delta^2}{2(1-\rho)} + \frac{(\lambda-\lambda^*)(1-\lambda^*\delta-e^{-\lambda^*\delta})}{(\lambda^*)^2(1-\rho)} \\ &+ \frac{\lambda(\rho-\rho^*-\lambda^* u)+(\lambda^*-\lambda)e^{-\lambda^* u}}{\lambda^*(1-\rho)} \cdot \frac{s(2+\lambda s)}{2(1+\lambda s)}, \end{aligned} \tag{12}$$

where u denotes the virtual backlog including the remaining setup delay δ, and $\rho^* = \lambda^* \mathrm{E}[X^*]$.

Proof: Consider first the difference in the expected cost incurred during $(0, T^*)$ starting in state z and the long-run average cost for the same interval, $A_0 = \mathrm{E}[V_z(T^*) - T^* \cdot r]$, where $r = \mathrm{E}[N]$, given by (9). Thus,

$$\begin{aligned} A_0 &= \sum_i \mathrm{E}[\min(y_i, T^*)] - \frac{r}{\lambda^*} = \sum_i \frac{1-e^{-\lambda^* y_i}}{\lambda^*} - \frac{r}{\lambda^*} \\ &= \frac{1}{\lambda^*}\left(n - \sum_i e^{-\lambda^* y_i} - \frac{\rho}{1-\rho} - \frac{\lambda s(2+\lambda s)}{2(1+\lambda s)}\right). \end{aligned}$$

Then at time T^* a job with size X^* arrives, and we need to compute the value function (10) for the resulting state (after the arrival). Let Δ denote the remaining setup delay at time T^*. For the first sum in (10) we have

$$A_1 = \frac{\mathrm{E}[\Delta] + \mathrm{E}[X^*]}{1-\rho} + \sum_i \frac{\mathrm{P}\{T^* < y_i\}\,\mathrm{E}[y_i - T^* + X^* | T^* < y_i]}{1-\rho},$$

where the first term corresponds to the arriving job with size X^*, and the summation to the earlier jobs possibly still present. This gives

$$\begin{aligned} A_1 = \frac{1}{1-\rho}\Bigg(&\delta - \frac{1-e^{-\lambda^*\delta}}{\lambda^*} + s e^{-\lambda^* u} + \mathrm{E}[X^*] \\ &+ \sum_i \left[y_i + (1-e^{-\lambda^* y_i})\left(\mathrm{E}[X^*] - \frac{1}{\lambda^*}\right)\right]\Bigg). \end{aligned}$$

The second term in (10) corresponds to the remaining setup delay and we have

$$A_2 = \frac{\lambda\,\mathrm{E}[\Delta^2]}{2(1-\rho)} = \frac{\lambda}{1-\rho}\left(\frac{\delta^2 + e^{-\lambda^* u} s^2}{2} + \frac{1-e^{-\lambda^*\delta} - \lambda^*\delta}{(\lambda^*)^2}\right),$$

The last term in (10) depends on the virtual backlog after the arrival at time T^*, denoted by U^*, and

$$\mathrm{E}[U^*] = \mathrm{P}\{T^* < u\}\,\mathrm{E}[u - T^* | T^* < u] + \mathrm{P}\{T^* > u\}\, s + \mathrm{E}[X^*],$$

which reduces to

$$\mathrm{E}[U^*] = u + \mathrm{E}[X^*] - \frac{1-(1+\lambda^* s)e^{-\lambda^* u}}{\lambda^*}.$$

Multiplying $\mathrm{E}[U^*]$ by the constant factor in the last term of (10) gives the final part for the lookahead cost,

$$A_3 = \frac{-\lambda s(2+\lambda s)}{2(1-\rho)(1+\lambda s)}\left(u + \mathrm{E}[X^*] - \frac{1-(1+\lambda^* s)e^{-\lambda^* u}}{\lambda^*}\right).$$

The sum $A_0 + A_1 + A_2 + A_3$ then gives (12). ∎

Note that $L(z,\lambda^*,X^*) = v_z - v_0 + D(z,\lambda^*,X^*)$, where the extra terms $D(z,\lambda^*,X^*) \to 0$ when $\lambda^* \to \lambda$ and $X^* \to X$, i.e., the identity $L(z,\lambda,X) = v_z - v_0$ holds, because if the next job arrives at rate λ and has size X, the system's future behavior is normal from the beginning.

Corollary 4: Without a setup delay ($s = 0$), the lookahead cost reduces to

$$L(z,\lambda^*,X^*) = \sum_{i=1}^{n} \frac{y_i}{1-\rho} + \frac{\left(n+1-\sum_{i=1}^{n} e^{-\lambda^* y_i}\right)(\rho^*-\rho)}{\lambda^*(1-\rho)}.$$

2) Dispatching jobs: Consider next a system with k parallel servers, where jobs arrive at rate λ. As with the FPI approach, the Lookahead policy chooses the action that gives the smallest expected cost (difference). For the action routing both the current and the next job to Queue i we have

$$a_{ii} = L^{(i)}(z_i \oplus x, \lambda, X) - L^{(i)}(z_i, \lambda, 0),$$

and for actions (i,j) with $j \neq i$,

$$\begin{aligned} a_{ij} = {} & L^{(i)}(z_i \oplus x, \lambda, 0) - L^{(i)}(z_i, \lambda, 0) \\ & + L^{(j)}(z_j, \lambda, X) - L^{(j)}(z_j, \lambda, 0). \end{aligned}$$

In the above, we have subtracted $\sum_i L^{(i)}(z_i, \lambda, 0)$ from all the possible actions, which makes the computation of the lookahead costs a local operation. The Lookahead policy chooses the most promising action,

$$\alpha_{LH}(z,x) = \arg\min_i \left\{ \min_j \; a_{i,j} \right\}.$$

IV. M/D/1-PS with a Setup Delay

Next we consider PS with jobs of fixed size d. The value function with respect to response time in an ordinary M/D/1-PS queue has been given in [16]. We are interested in an M/D/1-PS with a setup delay, and derive first two expressions for the expected costs, denoted by $q(z)$, that the system incurs until the end of the current busy period from an initial state $z = (\delta; x_1, \ldots, x_n)$, where $x_1 \geq \ldots \geq x_n$. That is,

$$q(z) = \mathrm{E}[\int_0^{R_z} N_z(t)\, dt], \tag{13}$$

where R_z denotes the remaining length of the busy period and $N_z(t)$ is the number of jobs in the system at time t.

Proposition 1: The average total response time that an M/D/1-PS queue with a remaining setup delay of $\delta > 0$ incurs during the remaining busy period is

$$q(z) = \frac{\rho(nd+\delta)}{(1-\rho)^2} + \frac{2n^2 d + (1+\rho)(2n+\lambda\delta)\delta}{2(1-\rho)}, \tag{14}$$

where n denotes the initial number of jobs waiting ($x_i \equiv d$).

Proof: Because $\delta > 0$, the setup period is under way and $x_1 = \ldots = x_n = d$ as no job has received any service yet. Let A denote the number of jobs arriving during the remaining setup period, $A \sim \text{Poisson}(\lambda\,\delta)$. Therefore, before the setup is over, the system will incur an average response time of

$$q_1 := n \cdot \delta + \lambda\,\delta \cdot \frac{\delta}{2} = (n + \lambda\,\delta/2)\,\delta,$$

where the first term is the delay of the present n jobs and the second term corresponds to the jobs arriving during $(0, \delta)$. Without new arrivals, the remaining response time of the initial $M = n + A$ jobs is Md, which gives the total cost of $M^2 d$ because serving all M jobs concurrently takes time Md. As M is a random variable, on average we have that the expected cost for this part is

$$q_2 = \text{E}[M^2 d] = \big((n + \lambda\,\delta)^2 + \lambda\,\delta\big)\,d.$$

The final part is to include the jobs that arrive once service has begun. Each arrival after the setup period increases the total response time by (see (3) in [16])[2]

$$c(u) = 2u + d, \tag{15}$$

where u denotes the backlog in the queue upon the arrival. Suppose first that the initial backlog is u. In particular, considering only the total backlog, each arrival starts a *mini busy period* as illustrated in Fig. 2. Apart from the random offset U^* in the backlog, the busy period is identical to that of an ordinary M/D/1 queue. The actual backlog the arriving jobs see is the sum of U^* and the backlog in an M/D/1 queue at a random arrival. The latter is the same as the waiting time W^* in an equivalent FCFS queue, and its mean is given by the Pollazcek-Khinchine formula, $\text{E}[W^*] = \lambda d^2/(2(1-\rho))$. Thus the average actual backlog that new arrivals see is

$$\text{E}[U^* + W^*] = \frac{u}{2} + \frac{\lambda d^2}{2(1-\rho)},$$

because the remaining backlog from the initial backlog u is on average $\text{E}[U^*] = u/2$. Moreover, there are on average λu i.i.d. mini busy periods, each comprising on average $\text{E}[N_B] = (1-\rho)^{-1}$ jobs. Combining these gives the mean additional response time due to new arrivals when the setup delay is over and the initial backlog is u,

$$q_3^{(u)} = \frac{\lambda u}{1-\rho}\,(2\,\text{E}[U^* + W^*] + d) = \frac{\lambda u^2}{1-\rho} + \frac{\rho u}{(1-\rho)^2}. \tag{16}$$

Because the initial backlog when service begins is a random variable, $U_s = Nd$, we need to substitute $\text{E}[U_s] = \text{E}[N]\,d$ and $\text{E}[U_s^2] = \text{E}[N^2]\,d^2$ into (16),

$$q_3 = \frac{\lambda d^2\,\text{E}[N^2]}{1-\rho} + \frac{\lambda d^2\,\text{E}[N]}{(1-\rho)^2} = \frac{\lambda d^2}{1-\rho}\left(\text{E}[N^2] + \frac{\text{E}[N]}{1-\rho}\right).$$

The expected total cost until the end of the busy period is $q = q_1 + q_2 + q_3$, which gives (14) after some manipulation. ∎

[2]Note that q_2 can be also obtained by adding the M jobs to "service" one by one using (15), $c(0) + c(d) + \ldots + c((M-1)d) = M^2 d$.

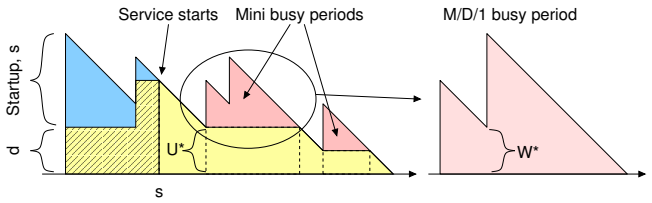

Fig. 2. Busy period with setup delay s contains a random number of ordinary M/D/1 busy periods.

Proposition 2: An M/D/1-PS queue already serving jobs ($\delta = 0$) incurs an average total response time of

$$q(z) = 2\sum_i i x_i + \left(\lambda\frac{d + (1-\rho)u}{(1-\rho)^2} - 1\right)u, \tag{17}$$

where $x_1 \geq \ldots \geq x_n$ is assumed and $u = x_1 + \ldots + x_n$.

Proof: When the server is already active we can utilize (16). Additionally, the remaining response time of the current n jobs without new arrivals must be taken into account. For the latter we have [16],

$$\sum_i (2i-1)\,x_i = 2\sum_i i x_i - u, \qquad (x_n \leq \ldots \leq x_1).$$

Adding the above and (16) together gives

$$q = 2\sum_i i x_i - u + \frac{\lambda u^2}{1-\rho} + \frac{\rho u}{(1-\rho)^2},$$

which yields (17). ∎

Consider next a fixed setup delay of s. According to (13), the mean number of jobs in the system is $\text{E}[N] = q(z)/\text{E}[\ell]$, where $z = (s; d)$, (14) gives $q(z)$ and (2) gives $\text{E}[\ell]$. Little's result then gives the mean response time in an M/D/1-PS,

$$\text{E}[T] = \frac{d}{1-\rho} + (1+\rho)\frac{s(2+\lambda s)}{2(1+\lambda s)}. \tag{18}$$

We note that when $s \to 0$, the above converges to $d/(1-\rho)$ as expected. As with LCFS, by unconditioning one can also generalize (18) to random i.i.d. setup delays $S_i \sim S$:

Theorem 4 (Mean response time in M/D/1-PS): The mean response time in an M/D/1 with PS is

$$\text{E}[T] = \frac{d}{1-\rho} + (1+\rho)\frac{\text{E}[S] + (\lambda/2)\,\text{E}[S^2]}{1 + \lambda\,\text{E}[S]}. \tag{19}$$

Proof: Omitted for brevity (see Eqs. (7) and (8)). ∎

Hence, the delay penalty for M/D/1-PS is is $(1+\rho)p_S$, whereas with FCFS and LCFS we had only p_S, see (8).

Corollary 5 (Mean response time in M/M/1): The mean response time in M/M/1 with setup delay s and an arbitrary work conserving scheduling discipline (e.g., LCFS or PS) is

$$\text{E}[T] = \frac{1}{\mu - \lambda} + \frac{s(2+\lambda s)}{2(1+\lambda s)}. \tag{20}$$

Proof: Substitute $X \sim \text{Exp}(\mu)$ into (7). ∎

A comparison of (18) and (20) reveals an interesting result:

Corollary 6 (Sensitivity): An M/G/1-PS queue loses its insensitivity property when a setup delay is introduced.

Additionally, we have also the result for the size-aware value function for an M/D/1-PS with a fixed setup delay:

Theorem 5 (Value function): The value function w.r.t. the response time in an M/D/1-PS with setup delay s is

$$v_z - v_0 = q(z) - \frac{u}{1-\rho}\, r,$$

where $q(z)$ is given by (14) or (17), $u = \delta + x_1 + \ldots + x_n$ and r is the mean cost rate, $r = \lambda \mathrm{E}[T]$.

Proof: Follows directly from Propositions 1 and 2. ∎

We do not write the value function out explicitly due to the length of the expression. However, the expression for admission costs is more compact. As mentioned, the admission cost corresponds to the increase in the value function, $a(z) = v_{z\oplus d} - v_z$, where $z \oplus d$ denotes the state with one new job. Therefore, the new state when a job arrives to an empty M/D/1-PS system is $z' = (s; d)$ and using (14) for $q(z')$ gives

$$a(0) = q(s;d) - \frac{s+d}{1-\rho}\, r = \frac{d}{1-\rho} + \frac{(1+\rho)s}{2} \cdot \frac{2+\lambda s}{1+\lambda s},$$

and thus $a(0) = \mathrm{E}[T]$ given by (18). For (remaining) setup period of $\delta > 0$, and $u = nd + \delta$

$$a(n,\delta) = \frac{(2n+1)d + (1+\rho)\,\delta}{1-\rho} - \frac{\rho s(1+\rho)(2+\lambda s)}{2(1-\rho)(1+\lambda s)}.$$

When the service has begun, so $\delta = 0$, the admission cost is

$$a(u) = \frac{2u+d}{1-\rho} - \frac{\rho s(1+\rho)(2+\lambda s)}{2(1-\rho)(1+\lambda s)}.$$

Combining the last two cases gives

$$a(u,\delta) = \frac{2u+d}{1-\rho} + \frac{1(u=0)-\rho}{1-\rho} \cdot \frac{s(1+\rho)(2+\lambda s)}{2(1+\lambda s)} - \delta. \qquad (21)$$

V. Numerical examples

First we consider elementary two server systems, where both servers are equally fast, but the secondary server has a setup delay of $s_2 = \{0, 2\}$. Then we consider a system of four servers and determine also the optimal combination of elementary switch-off policies `InstantOff` and `NeverOff` introduced in [20], [21], where the former switches the server immediately off when it becomes idle, and the latter keeps the server always running even when it is idle. With `NeverOff`, Server i incurs energy costs continuously at rate e_i and has no setup delay ($s_i = 0$). Expressions for FPI with respect to running costs were given in Section II-F, and for Lookahead we refer to [2].

A. Two servers

Here we assume a two server system and focus on the mean response time (excluding energy costs). The primary server is `NeverOff` ($s_1 = 0$), whereas the secondary server is switched off when it becomes idle, and it has a setup delay of $s_2 = 0$ or 2. We consider the following heuristic policies:

- Join-the-shortest-queue sends jobs to a shorter queue:
  JSQ resolves ties in favor of the primary server,
  JSQ* in favor of the secondary server, and
  JSQ′ favors primary if empty and secondary otherwise.

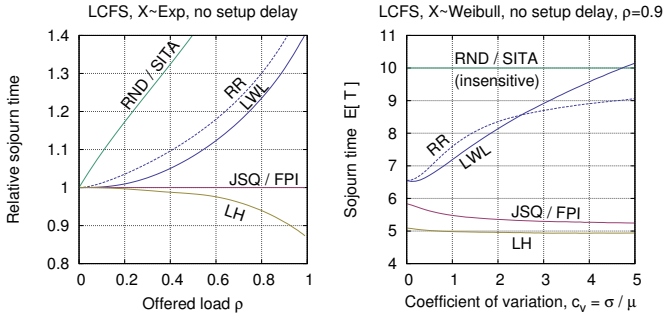

Fig. 3. Simulation results with two LCFS servers without setup delays. On the left, the job sizes are exponentially distributed. On the right, the job sizes obey the Weibull distribution and the offered load is fixed to $\rho = 0.9$.

- SITA sends jobs shorter than ξ to the primary server and the rest to the secondary server. SITA-E uses the load-balancing threshold ξ, and SITA-opt the optimal ξ.
- RND-U sends half of the jobs to the primary server and the rest to the secondary. RND-opt uses the optimal split and favors the server with a shorter setup delay.
- Round-Robin (RR) alternates between the two servers.
- LWL chooses the server with the shortest backlog, where possible setup delay (after the assignment) is included.
- Myopic minimizes the cost assuming no other jobs arrive.

Additionally, we also consider FPI and Lookahead (LH) policies based on SITA-E unless otherwise specified. [3]

1) LCFS: The numerical results with two identical LCFS servers without setup delays and exponentially distributed service times are depicted in Fig. 3 (left). FPI, as mentioned, reduces to JSQ. The heuristic policies RND (with optimal p_i), RR and LWL are worse than JSQ. Note that, depending on the information available to the dispatcher, each of RND, RR, JSQ, and LWL may be optimal when the service discipline is FCFS. LH achieves the lowest response time, especially when ρ is high. In Fig. 3 (right), we have fixed $\rho = 0.9$ and vary the coefficient of variation, c_v, of the Weibull distribution. With $c_v = 1$ one obtains exponential distribution. We see that LH remains better than JSQ/FPI, but the difference gets a bit smaller as c_v increases. Note also that LWL is a poor choice when c_v is high, and even a static RND does a better job.

The numerical results with setup delay s_2=2 and exponentially distributed service times are depicted in Fig. 4 (a)-(c). From (a) we see that RND-U becomes equivalent to RND-opt, and SITA-E becomes equivalent to SITA-opt, when $\rho > 0.5$. From (b) we see that JSQ* should not be used as either JSQ or JSQ′, or both, are at least as good. Also the difference between JSQ and JSQ′ is modest. From (a)-(c) we conclude that JSQ, as expected, is a robust policy for LCFS (with modest setup delays). However, JSQ′, LWL and the optimal static policies are better at some levels of load. In contrast, FPI and LH clearly achieve the lowest mean response for all values of ρ.

[3] Alternatively, one can also use RND or SITA-opt as a base policy. In our experiments, SITA-E typically yielded a better policy than RND, and the difference between SITA-E and SITA-opt was marginal.

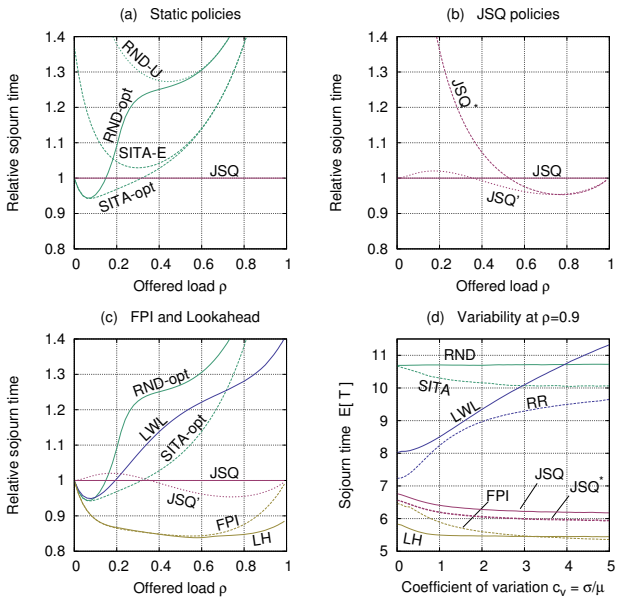

Fig. 4. Results for two LCFS servers with setup delay s_2=2. In (a)-(c), $X \sim$ Exp(1) and ρ is varied. In (d), ρ=0.9 and the coefficient of variation c_v of the Weibull distribution is varied, so c_v=1 corresponds to Exp(1).

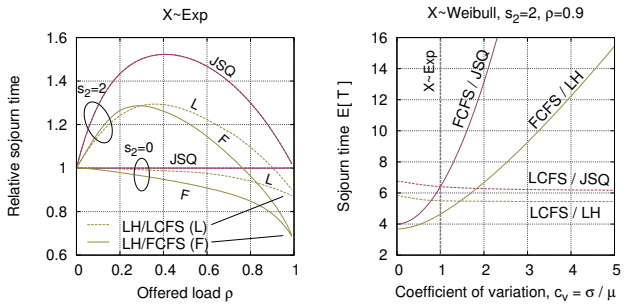

Fig. 5. Comparison of LCFS and FCFS in a system of two parallel servers with and without setup delay. Left: job sizes are exponential and ρ is varied. Right: $\rho = 0.9$ and c_v of the Weibull distribution is varied.

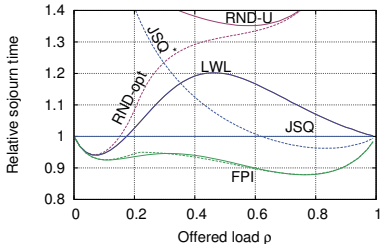

Fig. 6. Numerical results for two PS servers with $s_1 = 0$ and $s_2 = 2$.

Fig. 4 (d) depicts how the variance in the job sizes affects the sojourn time. Note that a single LCFS queue, and thus also RND, are insensitive to the job size distribution. As c_v increases, the performance of LWL and RR degrades, while the performance of SITA, JSQ, JSQ*, FPI and LH improves. The latter are also fairly insensitive to increase in c_v.

2) FCFS vs. LCFS: Let us next compare FCFS to LCFS for the same two server system. Suppose first that the job sizes are exponentially distributed. In this case, JSQ obtains the same response time independently of the scheduling discipline (FCFS, LCFS or PS). We choose JSQ without setup delays ($s = 0$) as the reference policy. As LH was the best policy in the earlier experiments, we compare its performance to that of JSQ. Fig. 5 (left) depicts the relative performance of FCFS and LCFS service under LH. We can see that for each policy the performance with a setup delay converges to that without a setup delay as the offered load ρ tends to one, as expected. Generally, FCFS/LH achieves the lowest response time, but LCFS/LH is marginally better when $s = 2$ and ρ is small.

In Fig. 5 (right), we fix ρ=0.9 and vary c_v again. As is well known, FCFS is better than LCFS in M/G/1 when $c_v < 1$, and vice versa. The same happens with JSQ, but with the more advanced dispatching policy LH the turning point is a bit higher and LCFS/LH is better than FCFS/LH when $c_v > 1.5$. A similar shift in the turning point happens also with LWL.

3) PS: Due to lack of space, we provide only a minimal example for PS with fixed service times, and we exclude LH because the expressions are a bit more complicated. In Fig. 6 we see that RND-U load balancing is a poor choice when the offered load is low and the secondary server with setup delay should be avoided. LWL is initially good, but when $\rho > 0.2$ it is also clearly sub-optimal. FPI is again the best policy.

B. Four servers

Next we assume a system of four LCFS servers and exponentially distributed job sizes. We also take the energy consumption explicitly into account by including running costs. In particular, we assume four identical servers with service rates $\nu = (1, 1, 1, 1)$, running costs $e = (1, 1, 1, 1)$, setup delays $s = (1, 1, 1, 1)$, and seek the optimal energy-performance tradeoff with objective $\min\ \lambda \mathrm{E}[T] + \sum_{i=1}^{4} \mathrm{P}\{\text{Server } i \text{ running}\} \cdot e_i$. The results are illustrated in Fig. 7. In contrast to the previous examples, here we have also chosen the optimal switching off policy for each dispatching policy and level of load ρ. That is, for each policy α and arrival rate λ, we simulated the system with all possible combinations of the switching off policies (`NeverOff` and `InstantOff`), and selected the one with the best performance. We can see that Lookahead yields superior performance for all ρ. Myopic is also very good when $\rho < 0.2$, and similarly, FPI does a good job until $\rho > 0.7$. Fig. 8 depicts the optimal switch off configuration with JSQ and LH. As the load increases, the pool of always running servers increases.

VI. Discussion

Table I summarizes the interesting observations made in the previous sections. Namely, the largely celebrated *insensitivity of the mean response time under PS breaks down dramatically when we include features present in actual systems!* The mean performance of LCFS as well as its value function are both insensitive to service time distribution. Also, the size-aware value function of FCFS is insensitive to service time distribution with or without setup delays. These suggest that the assignment of jobs to FCFS and LCFS servers in a robust manner is straightforward.

Fig. 7. Performance with 4 identical servers when a tradeoff between the mean response time and energy consumption must be made.

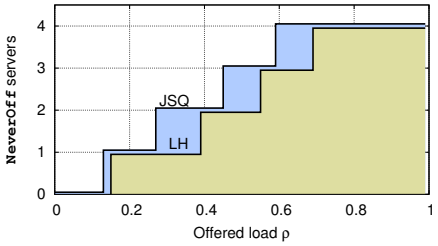

Fig. 8. The optimal switch off policy with JSQ and LH.

The fact that the increase in the mean response time with FCFS and LCFS corresponds to the same additional term holds for a larger class of service systems:

Theorem 6 (Separability): If the mean response time in a work conserving service system with a Poisson arrival process is additively separable, $\mathrm{E}[T] = g_X(\lambda) + p_S(\lambda)$, then

$$p_S(\lambda) = \frac{\mathrm{E}[S] + \lambda\,\mathrm{E}[S^2]/2}{1 + \lambda\,\mathrm{E}[S]}.$$

Proof: If $\mathrm{E}[T]$ separates, then it holds also for the trivial case of $X = 0$. In such a system, the response time incurred is purely the remaining setup delay, $g_X(\lambda) = 0$. Consequently,

$$p_S(\lambda) = \frac{1}{\lambda} \cdot \frac{\mathrm{E}[S + \lambda S^2/2]}{1/\lambda + \mathrm{E}[S]} = \frac{\mathrm{E}[S] + \lambda\,\mathrm{E}[S^2]/2}{1 + \lambda\,\mathrm{E}[S]}.$$

■

In contrast, the mean response time for PS is not separable.

VII. Conclusions

Due to the phenomenal growth of cloud-based services, energy consumption has become an important part of the overall system design. In this paper, we have considered a queueing model for a server farm that captures the essential characteristics of such a system: energy consumption, switching off servers, setup delays and the mean response time. First we obtained interesting queueing theoretic results and showed that the mean response time in M/G/1-PS is no longer insensitive to service time distribution when setup delays are present. In contrast, M/G/1-LCFS preserves this property making it a more robust scheduling discipline than PS. We also derived size-aware value functions for M/G/1-LCFS and M/D/1-PS queues subject to setup delays, and applied these results to develop efficient job assignment policies.

TABLE I
Insensitivity to service time distribution.

	Mean response time			Value function	
	M/G/1	with setup		M/G/1	with setup
LCFS	✓	✓	LCFS	✓	✓
PS	✓	-	PS	-	-
FCFS	-	-	FCFS	✓	✓

References

[1] L. Kleinrock, *Queueing Systems, Volume I: Theory*. Wiley, 1975.

[2] E. Hyytiä, R. Righter, and S. Aalto, "Task assignment in a heterogeneous server farm with switching delays and general energy-aware cost structure," *Performance Evaluation*, vol. 75–76, pp. 17–35, 2014.

[3] T. Bonald and A. Proutière, "Insensitivity in processor-sharing networks," *Performance Evaluation*, vol. 49, no. 1-4, pp. 193–209, 2002.

[4] Z. Rosberg, Y. Peng, J. Fu, J. Guo, E. Wong, and M. Zukerman, "Insensitive job assignment with throughput and energy criteria for processor-sharing server farms," *IEEE/ACM Trans. on Networking*, vol. PP, no. 99, pp. 1–1, 2013, to appear.

[5] W. Winston, "Optimality of the shortest line discipline," *Journal of Applied Probability*, vol. 14, pp. 181–189, 1977.

[6] R. R. Weber, "On the optimal assignment of customers to parallel servers," *Journal of Applied Probability*, vol. 15, no. 2, Jun. 1978.

[7] D. J. Daley, "Certain optimality properties of the first-come first-served discipline for G/G/s queues," *Stochastic Processes and their Applications*, vol. 25, pp. 301–308, 1987.

[8] S. G. Foss, "Approximation of multichannel queueing systems," *Siberian Mathematical Journal*, vol. 21, pp. 851–857, 1980.

[9] G. Koole, "On the optimality of FCFS for networks of multi-server queues," CWI, Amsterdam, Technical report BS-R923, 1992.

[10] O. Akgun, R. Righter, and R. Wolff, "Partial flexibility in routing and scheduling," *Advances in Applied Probability*, vol. 45, pp. 673–691, 2013.

[11] A. Ephremides, P. Varaiya, and J. Walrand, "A simple dynamic routing problem," *IEEE Trans. on Automatic Control*, vol. 25, no. 4, Aug. 1980.

[12] Z. Liu and D. Towsley, "Optimality of the round-robin routing policy," *Journal of Applied Probability*, vol. 31, no. 2, pp. 466–475, Jun. 1994.

[13] Z. Liu and R. Righter, "Optimal load balancing on distributed homogeneous unreliable processors," *Oper. Res.*, vol. 46, no. 4, 1998.

[14] H. Feng, V. Misra, and D. Rubenstein, "Optimal state-free, size-aware dispatching for heterogeneous M/G/-type systems," *Performance Evaluation*, vol. 62, no. 1-4, pp. 475–492, 2005.

[15] K. R. Krishnan, "Joining the right queue: a state-dependent decision rule," *IEEE Trans. on Automatic Control*, vol. 35, no. 1, Jan. 1990.

[16] E. Hyytiä, A. Penttinen, S. Aalto, and J. Virtamo, "Dispatching problem with fixed size jobs and processor sharing discipline," in *23rd International Teletraffic Congress (ITC'23)*, San Fransisco, USA, Sep. 2011.

[17] E. Hyytiä, A. Penttinen, and S. Aalto, "Size- and state-aware dispatching problem with queue-specific job sizes," *European Journal of Operational Research*, vol. 217, no. 2, pp. 357–370, Mar. 2012.

[18] J. R. Artalejo, A. Economou, and M. J. Lopez-Herrero, "Analysis of a multiserver queue with setup times," *Queueing Syst. Theory Appl.*, vol. 51, no. 1–2, pp. 53–76, Oct. 2005.

[19] A. Gandhi and M. Harchol-Balter, "M/G/k with exponential setup," Carnegie Mellon University, Technical Report CMU-CS-09-166, 2009.

[20] A. Gandhi, M. Harchol-Balter, and I. Adan, "Server farms with setup costs," *Performance Evaluation*, vol. 67, no. 11, pp. 1123–1138, 2010.

[21] A. Gandhi, V. Gupta, M. Harchol-Balter, and M. Kozuch, "Optimality analysis of energy-performance trade-off for server farm management," *Performance Evaluation*, vol. 67, no. 11, pp. 1155–1171, 2010.

[22] V. Maccio and D. Down, "On optimal policies for energy-aware servers," in *In Proc. of MASCOTS*, San Francisco, US, Aug. 2013.

[23] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 2005.

[24] E. Hyytiä, "Lookahead actions in dispatching to parallel queues," *Performance Evaluation*, vol. 70, no. 10, pp. 859–872, 2013.

[25] J. Medhi, *Stochastic Models in Queueing Theory*, 2nd ed. Elsevier, 2003.