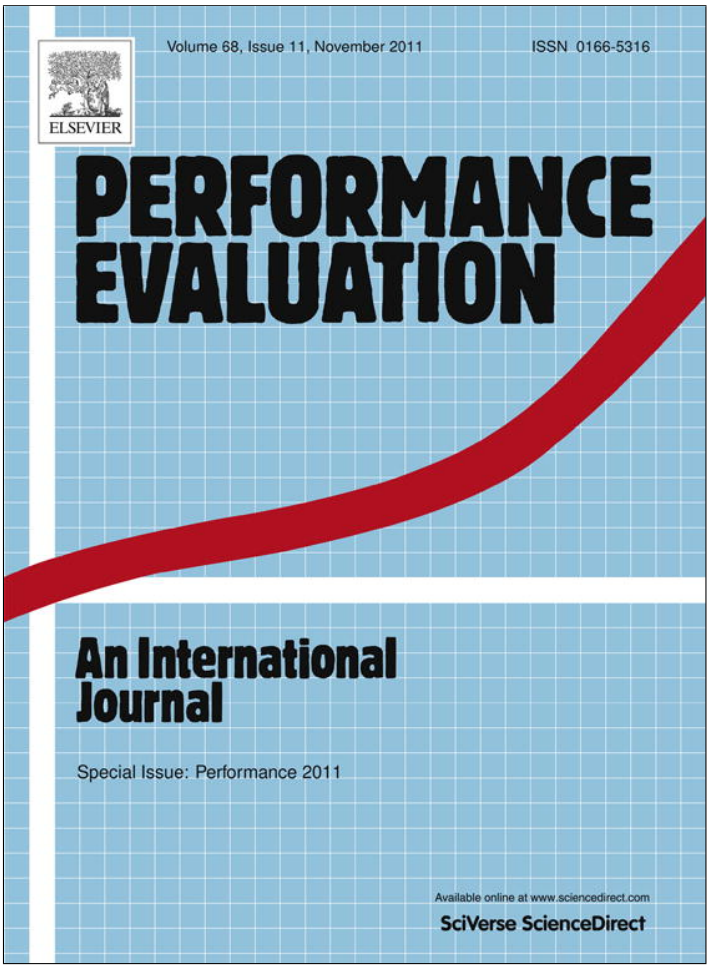

Performance Evaluation 68 (2011) 1136–1148

Contents lists available at SciVerse ScienceDirect

Performance Evaluation

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

M/M/1-PS queue and size-aware task assignment

Esa Hyytiä *, Jorma Virtamo, Samuli Aalto, Aleksi Penttinen

Department of Communications and Networking, Aalto University School of Electrical Engineering, Finland

ARTICLE INFO

Article history:
Available online 3 August 2011

Keywords:
Dispatching
Task assignment
Routing
M/M/1
Processor sharing
Sojourn time
MDP

ABSTRACT

We consider a distributed server system in which heterogeneous servers operate under the processor sharing (PS) discipline. Exponentially distributed jobs arrive to a dispatcher, which assigns each task to one of the servers. In the so-called size-aware system, the dispatcher is assumed to know the remaining service requirements of some or all of the existing jobs in each server. The aim is to minimize the mean sojourn time, i.e., the mean response time. To this end, we first analyze an M/M/1-PS queue in the framework of Markov decision processes, and derive the so-called size-aware relative value of state, which sums up the deviation from the average rate at which sojourn times are accumulated in the infinite time horizon. This task turns out to be non-trivial. The exact analysis yields an infinite system of first order differential equations, for which an explicit solution is derived. The relative values are then utilized to develop efficient dispatching policies by means of the first policy iteration (FPI). Numerically, we show that for the exponentially distributed job sizes the myopic approach, ignoring the future arrivals, yields an efficient and robust policy when compared to other heuristics. However, in the case of highly asymmetric service rates, an FPI based policy outperforms it. Additionally, the size-aware relative value of an M/G/1-PS queue is shown to be sensitive with respect to the form of job size distribution, and indeed, the numerical experiments with constant job sizes confirm that the optimal decision depends on the job size distribution.

1. Introduction

Dispatching problems arise in many contexts such as manufacturing sites, web server farms, and other parallel server systems. Typically one is interested in minimizing the mean sojourn time. Within each server, the first-come-first-served (FCFS) discipline is perhaps the most common due to its nature and ease of implementation. It has also been studied extensively in the literature since the early works by Winston [1], Ephremides et al. [2], and others. However, e.g., web servers and distributed systems are better modeled as processor sharing (PS) queues, where several clients are served at the same time [3,4]. The PS queues [5] have a very convenient insensitivity result stating that the mean sojourn time depends only on the mean job size, facilitating, e.g., flow level analysis in data networks. Unfortunately, transient analysis with respect to sojourn time is difficult [6,4] and often limited to exponential service times [7,8]. While PS offers in some sense a fair schedule, the optimal scheduling discipline with respect to the mean sojourn time is the shortest-remaining-processing-time (SRPT) [9,10].

In a dispatching system, the optimal dispatching decision with respect to sojourn time depends on the scheduling policy and the available information in general. Often the number of jobs (or tasks) per server is assumed to be known; cf. *join-the-shortest-queue* (JSQ) policy. In contrast, [11,12] assume FCFS and that the dispatching policy is aware of the size of the new job, but not about the state of the queues, and propose the so-called *size-interval-task-assignment* (SITA) policy (e.g., short

* Corresponding address: Aalto University, School of Electrical Engineering, P.O. Box 13000, FI-00076 Aalto, Finland.
E-mail addresses: esa.hyytia@tkk.fi, esa@netlab.tkk.fi (E. Hyytiä).

0166-5316/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.peva.2011.07.011

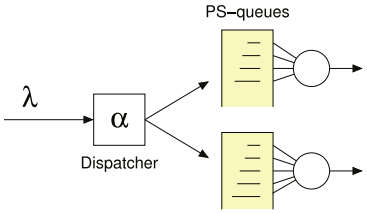

Fig. 1. Dispatching system with processor sharing (PS) queues.

jobs to one queue, and the rest to another). The optimality of SITA was established later in [13]. Bonomi considers PS with various state information in [3] and argues that JSQ is a near optimal policy with identical servers. To this end, a heuristic *Myopic* policy is developed by assuming that the (remaining) service requirements of all existing jobs are available and disregarding the future arrivals. Recently, Gupta et al. provide an approximate analysis of JSQ/PS system with general job size distributions [4]. Without any size or state information, one has to settle with a random policy known as the *Bernoulli splitting*, for which the optimal probabilities can be computed in different settings [14–16].

In this paper, we consider a dispatching problem with exponentially distributed job sizes and processor sharing as illustrated in Fig. 1. The dispatcher is aware of the state of each queue, i.e., the number of tasks and also the remaining service requirements of all or some of them. This kind of system serves as an abstract model, e.g., for file servers in a distributed content delivery network, web server farms, and multitasking computing systems [4,11,12,17].

First we study a single M/M/1-PS queue in isolation and derive an important result regarding the relative value of state, which characterizes the value of queue state with respect to the expected sojourn time in infinite time horizon. Then we apply these results to the dispatching problem and develop efficient and robust state-dependent policies in the Markov decision process (MDP) framework. In particular, starting from an arbitrary state-independent policy, the single queue results allow one to carry out the first policy iteration (FPI) step yielding *improved state-dependent policies* that take into account the future arrivals. The knowledge of the relative values is a prerequisite to this end.

A similar approach has been previously used by Krishnan in the context of routing calls in a telephone network [18] so as to minimize the blocking probability, and in the context of a dispatching problem for parallel M/M/s-FCFS servers [19] so as to minimize the mean sojourn time. With respect to the mean sojourn time, the traditional M/M/1-FCFS queue has been analyzed in [20]. Recently, FCFS, last-come-first-served (LCFS), shortest-processing-time (SPT) and SRPT disciplines with a general service time distribution are analyzed in [21], while the M/D/1-PS queue has been treated in [22].

The rest of this paper is organized as follows. In Section 2, we analyze a single M/M/1-PS queue, and derive a closed-form expression for the size-aware relative value of state with respect to the sojourn time. In Section 3, the theoretical result is applied to develop efficient dispatching policies within the MDP framework, and Section 4 contains the numerical examples. Section 5 concludes the paper.

2. Analysis of an M/M/1-PS queue

In this section, we analyze a single M/M/1-PS queue and develop a closed-form expression for the size-aware relative value with respect to sojourn time. The result is utilized later in Section 3 to carry out FPI on state-independent random dispatching policies.

Let $\mathbf{z} = (n; x_1, \ldots, x_m) = (n; \mathbf{x})$ denote the state of an M/M/1-PS queue with $n + m$ tasks, where n denotes the number of jobs with exponentially distributed unknown (remaining) workload, and $x_1 > x_2 > \cdots > x_m$ denote the remaining workloads of the m known tasks. Let λ denote the Poissonian arrival rate and $1/\mu$ the mean of the exponentially distributed job sizes. Furthermore, $\rho = \lambda/\mu$ is the offered load assumed to be less than one, $\rho < 1$.

The sojourn time costs are accrued at time t at a rate equal to the number of jobs in the queue, denoted by $N_{\mathbf{z}}(t)$ with $\mathbf{z}$ being the initial state of the queue. We are interested in the cumulative sojourn time during time interval $(0, t)$.

$$V_{\mathbf{z}}(t) \triangleq \int_0^t N_{\mathbf{z}}(s)\,\mathrm{d}s.$$

We already know the mean number of jobs in the system [23], i.e., the mean cost rate r,

$$r \triangleq \mathrm{E}[N] = \frac{\rho}{1-\rho} = \frac{\lambda}{\mu-\lambda}. \tag{1}$$

Our aim is to quantify the so-called relative value of state $\mathbf{z}$, denoted by $v_{\mathbf{z}}$ and defined as the expected difference in the cumulative sojourn time in infinite time horizon between a system initially in state $\mathbf{z}$ and a system initially in equilibrium,

$$v_{\mathbf{z}} \triangleq \lim_{t\to\infty} (\mathrm{E}[V_{\mathbf{z}}(t)] - \mathrm{E}[N]\cdot t).$$

For a stable queue with $\lambda < \mu$ the above limit is well-defined and finite. The difference $v_{\mathbf{z}_1} - v_{\mathbf{z}_2}$ then characterizes the expected difference in the future costs (in sojourn time) between two initial states $\mathbf{z}_1$ and $\mathbf{z}_2$. For two special cases, an expression for $v_{\mathbf{z}}$ is already available from the past work:

Lemma 1. *The relative value of a state with n exponentially distributed jobs in M/M/1-PS queue with arrival rate of* λ *and mean job size of* $1/\mu$ *is given by,*

$$v_{(n;0)} = v_n = \frac{1}{2} \cdot \frac{n(n+1)}{\mu - \lambda} - \frac{\lambda\mu}{(\mu-\lambda)^3}. \tag{2}$$

The result is derived in [20,24] and holds for an arbitrary work conserving M/M/1 queue. The constant term follows from the identity $\sum_i \pi_i v_i = 0$, where the π_i denote the steady-state probability distribution of the system.

The *myopic expression* for the cumulative sojourn time is given in [3,22] by assuming that no further jobs arrive at the queue, or equivalently, when $\lambda \to 0$. The (deterministic) cumulative sojourn time is equal to the relative value:

Lemma 2. *The relative value of a state with m known remaining service requirements* $x_1 > x_2 > \cdots > x_m$ *in M/M/1-PS queue with arrival rate of* λ *and mean job size of* $1/\mu$*, at the limit* $\lambda \to 0$*, is given by,*

$$v_{(0;\mathbf{x})|\lambda\to 0} = x_m m^2 + (x_{m-1} - x_m)(m-1)^2 + \cdots + (x_1 - x_2) = \sum_{i=1}^{m} (2i-1)x_i. \tag{3}$$

2.1. Exact analysis of a single task

For simplicity, let us consider first state $\mathbf{z}$ comprising a single known job with size x and n jobs with unknown exponentially distributed service requirements, $\mathbf{z} = (n; x)$. Our aim is to derive an expression for the relative value with respect to sojourn time, $v_{\mathbf{z}}$. For ease of notation, in this case we can simply write $\nu_n(x) \triangleq v_{\mathbf{z}}$ for state $\mathbf{z} = (n; x)$.

First, considering a differential time interval Δ at state $\mathbf{z} = (n; x)$, we have

$$\begin{aligned}\nu_n(x) = {} & (n+1-r)\Delta + \lambda\,\Delta\,\nu_{n+1}(x - \Delta/(n+1)) + \frac{n}{n+1}\,\mu\,\Delta\,\nu_{n-1}(x - \Delta/(n+1)) \\ & + \left(1 - \left(\lambda + \frac{n}{n+1}\mu\right)\Delta\right)\,\nu_n(x - \Delta/(n+1)),\end{aligned} \tag{4}$$

where the first term corresponds to the difference in the cost during a short time interval of Δ, the second and third terms to the future costs when a job arrives or departs during $(0, \Delta)$, respectively, and the fourth term to the future costs with no change in the occupation during $(0, \Delta)$. At the limit $\Delta \to 0$, the above Eq. (4) then gives

$$\nu_n'(x) = (n+1)(n+1-r) + (n+1)\lambda(\nu_{n+1}(x) - \nu_n(x)) + n\,\mu(\nu_{n-1}(x) - \nu_n(x)), \tag{5}$$

where the last term is omitted for $n = 0$. Moreover, (2) already gives the boundary conditions,

$$\lim_{x\to 0} \nu_n(x) = v_n. \tag{6}$$

Eq. (5) is an infinite system of first order (ordinary) differential equations, which together with the boundary conditions (6), has a unique solution. We note that (5) is similar to the system of differential equations obtained in [7] for finding the Laplace transform of the conditional sojourn time distribution of a job with size x in the same setting (M/M/1-PS queue with a job of size x and n other jobs with unknown exponentially distributed sizes).

Proposition 1. *The relative value of state* $(n; x)$ *comprising one known task of size x and n tasks with unknown exponentially distributed sizes in M/M/1-PS queue with arrival rate of* λ *and mean job size of* $1/\mu$ *is given by*

$$v_{(n;x)} = v_n + \frac{1}{\mu(1-\rho)^2}\left[\mu x + (2-\rho)\left(n - \frac{\rho}{1-\rho}\right)(1 - \mathrm{e}^{-\mu(1-\rho)x})\right]. \tag{7}$$

Proof. A proof follows by substituting $v_{(n;x)} = \nu_n(x)$ from (7) to (5) and observing that also the boundary conditions (6) are met. □

Example 1. Let us consider a system with $\lambda = 1$ and $\mu = 2$ so that the offered load is $\rho = 1/2$ and the mean number of jobs in the system is $r = 1$. The numerical results for the first few $\nu_n(x)$ are illustrated in Fig. 2. On x-axis is the length of the known task, and y-axis gives the corresponding relative value (left graph) and its derivative (right graph). The relative values increase approximately linearly for large x, as all derivatives converge to 4 as x tends to ∞. Even though $\nu_1'(x)$ is a constant in Fig. 2, this is not generally the case. Substituting $\lambda = 1$ and $\mu = 2$ into (7) yields

$$\begin{aligned}\nu_0(x) &= 3\mathrm{e}^{-x} + 4x - 5, \\ \nu_1(x) &= 4x - 1, \\ \nu_2(x) &= -3\mathrm{e}^{-x} + 4x + 4, \ldots.\end{aligned}$$

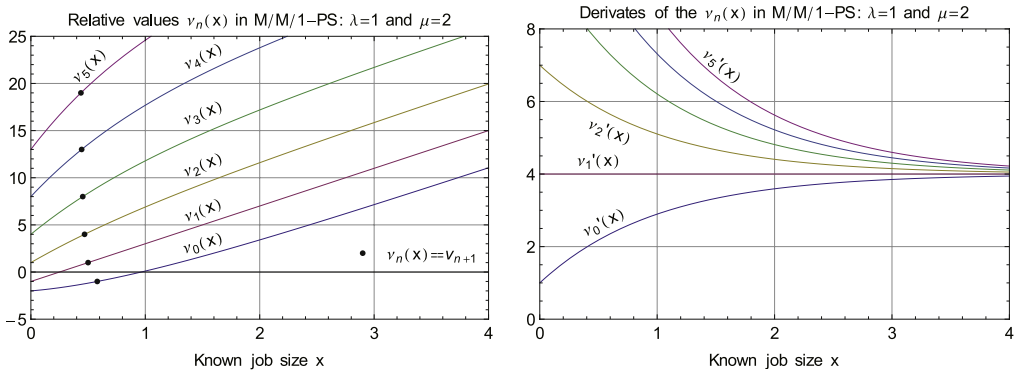

Fig. 2. Relative value $v_n(x)$ of a single known job with size x and its derivative. The subscript n denotes the number of tasks with unknown remaining job sizes.

2.2. Exact analysis of multiple tasks

The previous treatment generalizes to the case of m jobs with known sizes and n jobs with unknown exponentially distributed service requirements (i.i.d.). Again, our objective is to derive an expression for the size-aware relative value with respect to sojourn time, denoted by $v_{\mathbf{z}} = v_{(n;x_1,\ldots;x_m)}$. To this end, it is convenient to define the state by the differences y_j such that (see Fig. 3 (left))

$$\begin{aligned} x_1 &= y_1 + y_2 + \cdots + y_m \\ x_2 &= y_2 + \cdots + y_m \\ &\vdots \\ x_m &= y_m. \end{aligned} \tag{8}$$

So $\mathbf{y} = (n; y_1, \ldots, y_m)$ and $\mathbf{z} = (n; x_1, \ldots, x_m)$ correspond to the same state with different state description. Note that under the PS discipline, all m known jobs receive service at the same rate. Thus the differences $y_1, y_2, \ldots, y_{m-1}$ between the unfinished works do not change; it is only the common part y_m that decreases as the service proceeds. When y_m becomes zero, the shortest job is completed and in the remaining system of $m-1$ known jobs the service continues with only y_{m-1} decreasing etc. The known jobs depart from the system in the predefined order, and we refer to these durations between the departures as the stages. First is stage m with m known tasks present, then stage $m-1$, etc. Let $\mathbf{y}_k = (y_1, \ldots, y_k)$ denote the initial state at the start of stage k, and define $\mathbf{y}_0 \triangleq 0$. Thus, e.g., $\mathbf{y}_1 = (y_1)$ corresponds to the known workload when only one task is remaining, $\mathbf{y}_2 = (y_1, y_2)$, etc. Next, define auxiliary functions for k known jobs, $k = 1, 2, \ldots$,

$$v_n(\mathbf{y}_k) \triangleq v_{(n;y_1+\cdots+y_k,\ldots,y_{k-1}+y_k,y_k)},$$

where n is the number of exponentially distributed unknown workloads at start. Obviously, (7) already gives $v_n(y_1)$ for $k = 1$. The differential equation system for an arbitrary stage with k known tasks reads

$$\frac{\mathrm{d}}{\mathrm{d}y_k} v_n(\mathbf{y}_k) = (n+k)(n+k-r) + (n+k)\,\lambda(v_{n+1}(\mathbf{y}_k) - v_n(\mathbf{y}_k)) + n\,\mu(v_{n-1}(\mathbf{y}_k) - v_n(\mathbf{y}_k)). \tag{9}$$

The general boundary conditions are

$$\lim_{y_k \to 0} v_n(\mathbf{y}_k) = v_n(\mathbf{y}_{k-1}) \iff v_n(y_1; \ldots; y_{k-1}; 0) = v_n(y_1; \ldots; y_{k-1}), \tag{10}$$

with $v_n(0) = v_n$. Thus, the only difference to (5) is the constant factor $(n+k)$ instead of $(n+1)$. Differential equation system (9) with boundary conditions (10) can be solved recursively starting from $k = 1$ and working out the intermediate results for $k = 2, \ldots, m-1$, and then finally $k = m$. As (9) involves only the neighboring functions $v_{n+1}(\mathbf{y}_k)$ and $v_{n-1}(\mathbf{y}_k)$, the procedure to obtain, e.g., $v_0(\mathbf{y}_m)$ requires the computation of the intermediate results at the marked points in the triangle of Fig. 3. This is straightforward numerically. Moreover, it turns out that the following recursive equation holds:

$$\begin{aligned} v_n(\mathbf{y}_k) = v_n(\mathbf{y}_{k-1}) + \frac{1}{\mu(1-\rho)^2}\Bigg(\mu k^2 y_k + (2-\rho)\left(n - \frac{k\rho}{1-\rho}\right)(1 + \mathrm{e}^{-\mu(1-\rho)y_{k-1}} \\ + \mathrm{e}^{-\mu(1-\rho)(y_{k-2}+y_{k-1})} + \cdots + \mathrm{e}^{-\mu(1-\rho)(y_1+\cdots+y_{k-1})})(1 - \mathrm{e}^{-\mu(1-\rho)y_k})\Bigg). \end{aligned} \tag{11}$$

We note that the boundary conditions (10) hold by construction, and substituting (11) into (9) shows that the differential equation is also satisfied. Consequently, we have:

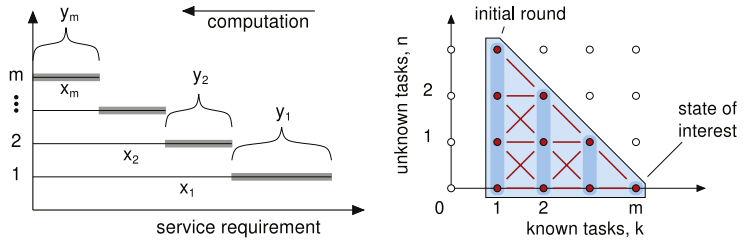

Fig. 3. Relative value of an arbitrary state $\mathbf{y} = (0; y_1, \ldots, y_m)$ is obtained recursively in $m-1$ steps.

$$v_n(\mathbf{y}_m) = v_n + \frac{1}{(1-\rho)^2}\sum_{k=1}^{m} k^2 y_k + \frac{2-\rho}{\mu(1-\rho)^2}\sum_{k=1}^{m}\left(n - \frac{k\rho}{1-\rho}\right)\left(1 + \mathrm{e}^{-\mu(1-\rho)y_{k-1}} + \cdots + \mathrm{e}^{-\mu(1-\rho)(y_1+\cdots+y_{k-1})}\right) \times \left(1 - \mathrm{e}^{-\mu(1-\rho)y_k}\right). \tag{12}$$

The first term merely defines the offset, the second term is the sum of the known sojourn times (3), i.e., the myopic component, multiplied by the constant $(1-\rho)^{-2}$, and the third term is related to the future arrivals. With the original state description, $\mathbf{z} = (n; x_1, \ldots, x_m)$, where $x_i > x_{i+1}$, the above gives our final result:

Proposition 2. *The relative value of state $(n; x_1, \ldots, x_m)$ comprising m tasks with remaining job sizes $x_1 > x_2 > \cdots > x_m$ and n tasks with unknown exponentially distributed (remaining) job sizes in M/M/1-PS queue with arrival rate of λ and mean job size of $1/\mu$ is given by*

$$v_{(n;x_1,\ldots,x_m)} = v_n + \frac{1}{(1-\rho)^2}\sum_{k=1}^{m}(2k-1)x_k + \frac{2-\rho}{\mu(1-\rho)^2}\sum_{k=1}^{m}\left(n - \frac{k\rho}{1-\rho}\right)\left(\sum_{i=1}^{k}\mathrm{e}^{-\mu(1-\rho)(x_i - x_k)}\right) \times \left(1 - \mathrm{e}^{-\mu(1-\rho)(x_k - x_{k+1})}\right), \tag{13}$$

where we have used the convention that $x_{m+1} \triangleq 0$.

We note that (13) converges to (3) when $n = 0$ and $\lambda \to 0$.

Remark 1. For comparison, an M/D/1-PS queue has been analyzed in [22]. With fixed job sizes $x = d$, tasks depart in the same order as they arrive, which facilitates the analysis, and eventually allows one to write an exact expression for the size-aware relative value with respect to sojourn time,

$$v_{(0;x_1,\ldots,x_m)} - v_0 = \frac{\lambda}{1-\rho}u_{\mathbf{z}}^2 - u_{\mathbf{z}} + 2\sum_{i=1}^{m} i\,x_i, \tag{14}$$

where v_0 denotes the relative value of an empty system, and $u_{\mathbf{z}}$ is the unfinished work (backlog), $u_{\mathbf{z}} = \sum_{i=1}^{m} x_i$.

Clearly, (14) is different from (13), i.e., the *size-aware relative values of an M/G/1-PS queue are sensitive to the job size distribution*. We note that [6] finds the Laplace transform for the sojourn time in an M/G/1-PS queue, which also establishes the sensitivity of PS queue on the higher order moments (than the mean) of the job size distribution. In contrast, for FCFS and (preemptive and nonpreemptive) LCFS disciplines, the size-aware relative values depend only on the mean job size and thus they are insensitive to the job size distribution [21].

Example 2. As an example, let us consider an M/M/1-PS queue with two tasks with remaining service requirements x_1 and x_2, respectively. The mean job size is $1/\mu = 1/2$, and the arrival rate is varied so that the offered load is $\rho = 0.1, 0.5, 0.8$. Fig. 4 illustrates the equivalue contours of the size-aware relative value in (x_1, x_2) space. We can observe that with a low load, for a given total workload $x_1 + x_2$, having highly asymmetric service requirements, i.e., $x_1 \gg x_2$ or $x_1 \ll x_2$, is clearly better than equal requirements, $x_1 \approx x_2$. However, when the offered load increases, the total workload $x_1 + x_2$ appears to dominate.

3. Application to dispatching problem

In this section, we utilize the size-aware relative value of M/M/1-PS queue to dispatching problem. First we describe the state-independent random policies and some state-dependent reference policies. Then the size-dependent FPI policies are given, and further illustrated in the numerical examples.

In a dispatching problem, a stream of tasks arrives at a dispatcher, which then forwards them to one of the q servers. Different queues may have different service rates, and we let c_j denote the service rate of queue j. Without lack of generality, we can assume that $c_1 \geq c_2 \geq \cdots \geq c_q$. For the sake of illustration, one can assume that the job sizes X with mean

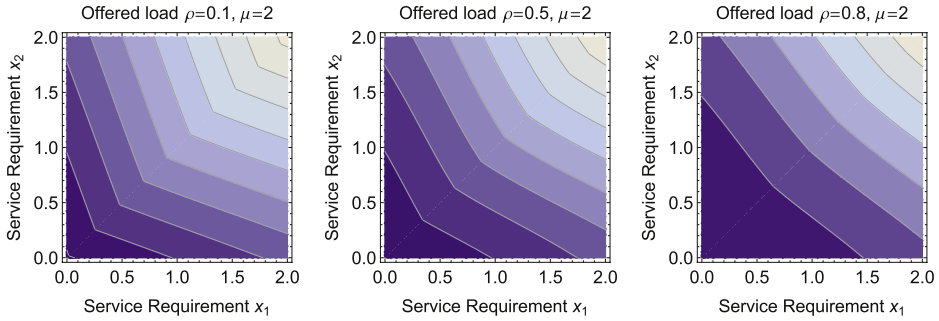

Fig. 4. Contour plots of the size-aware relative value in an M/M/1-PS queue with two tasks having service requirements x_1 and x_2. As load ρ increases, the amount of unfinished work $x_1 + x_2$ appears to dominate.

$1/\mu = \mathrm{E}[X]$ are measured in bytes, service rates c_j in bytes per second, and the queue specific service requirements X_j with mean $1/\mu_j$ in seconds. Thus, for a given job size x the following elementary relations hold:

$$\mu_j = \mu \cdot c_j, \qquad x_j = x/c_j, \qquad \rho_j = \lambda_j \,\mathrm{E}[X]/c_j, \tag{15}$$

where λ_j denotes the task arrival rate to queue j under given policy. The offered load to the whole system is $\rho = \lambda\,\mathrm{E}[X]/\sum_i c_i$ and assumed to be less than 1 (stable system).

3.1. State-independent random policies

A state-independent random policy, known as Bernoulli splitting, assigns each task to one of the q queues independently in random using a probability distribution $p_j, j = 1, \ldots, q$. Therefore, decisions are independent of the queue states. The queue specific loads are given by

$$\rho_j = (p_j/c_j)\,\lambda\,\mathrm{E}[X].$$

The important observation here is the fact that with Poisson arrival process and state-independent policy, *the arrival process to each queue remains also Poissonian*, which facilitates the FPI step (see Section 3.3). In this work, we consider two random policies: RND-ρ and RND-opt.

3.1.1. RND-ρ

One rational choice for the dispatching policy is to balance the load equally among the servers, so that the $\rho_i = \rho_j$ for all i, j. To this end, RND-ρ uses probability distribution

$$p_j = \frac{c_j}{\sum_i c_i},$$

which then gives $\rho_j = \lambda\,\mathrm{E}[X]/\sum_i c_i = \rho$. The resulting system is stable as long as the offered load ρ is less than one.

3.1.2. RND-opt

RND-opt policy uses the probability distribution that minimizes the mean sojourn time. The determination of the p_j is straightforward due to the insensitivity of M/G/1-PS. More specifically, the mean sojourn time is

$$\mathrm{E}[T] = \sum_i \frac{p_i}{\mu_i - p_i\lambda} = \sum_i \frac{p_i}{\mu c_i - p_i\lambda}, \tag{16}$$

from which the optimal probability distribution can be obtained with aid of Lagrange method, [14,15],

$$p_j = \begin{cases} \dfrac{1}{\lambda}\left(\mu_j - \dfrac{\sqrt{\mu_j}}{\beta}\right), & j \le q^*, \\ 0 & \text{otherwise}, \end{cases} \tag{17}$$

where, using the notation that $\mu_i = 0$ for $i > q$,

$$\beta = \frac{\sum_{j=1}^{q^*} \sqrt{\mu_j}}{\left(\sum_{j=1}^{q^*} \mu_j\right) - \lambda}, \quad \text{and} \quad q^* = \min\left\{ i \ge 1 : \mu_{i+1} \le \frac{\left(\left(\sum_{j=1}^{i} \mu_j\right) - \lambda\right)^2}{\left(\sum_{j=1}^{i} \sqrt{\mu_j}\right)^2} \right\}.$$

Thus, the optimal solution uses only the first q^* servers when the offered load is sufficiently small. A more general case with arbitrary server specific holding costs is treated in [16].

3.2. State-dependent policies

Let us next briefly describe some commonly considered state-dependent policies, where, in contrast to the previous section, dispatching decisions depend on the states of the queues. Let $\alpha(\mathbf{z}, x)$ denote the action, i.e., the queue a given policy chooses for a new job with size x when the system is in state $\mathbf{z}$. If the action does not depend on x, with a slight abuse of notation, we write $\alpha(\mathbf{z})$.

3.2.1. JSQ

Join-the-shortest-queue (JSQ) is a prime example of a state-dependent policy, where the dispatcher chooses the queue with the least number of jobs. In case of ties, the fastest server among the candidates is chosen. Letting n_i denote the number of jobs in queue i, the system's state is $\mathbf{z} = (n_1; \ldots; n_q)$, and the JSQ policy can be stated as

$$\alpha(\mathbf{z}) = \min\{i : n_i = \min_j n_j\},$$

as the service rates were ordered, $c_1 \geq c_2 \geq \cdots \geq c_q$. For the given state information, JSQ is also the optimal policy for exponentially distributed job sizes and identical PS servers, $c_i = c_j \; \forall\, i, j$ [1–3].

3.2.2. Myopic

One general dispatching policy stems from the assumption that no further jobs arrive after the job that has just arrived. In particular, when the job sizes are known, (3) allows one to compute the additional cost in terms of cumulative sojourn time for each possible action, and consequently to choose the optimal queue. This policy was introduced by Bonomi in [3] as π_Δ and referred to as the opt-0 policy in [4]. In contrast to the above work, our exact result (13) for the size-aware relative value explicitly includes the myopic term, which suggests that for Poisson arrival process and exponentially distributed job sizes, the myopic expression can be expected to be a robust and efficient policy. A sufficient state description is $\mathbf{z} = (\mathbf{z}_1; \ldots; \mathbf{z}_q)$ with queue states $\mathbf{z}_j = (x_{j,1}, \ldots, x_{j,m_j})$, where $x_{j,1} > x_{j,2} > \cdots > x_{j,m_j}$ denote the remaining service requirements in queue j in the decreasing order. Then, (3) gives

$$\alpha(\mathbf{z}, x) = \operatorname*{argmin}_j \left((2k_j + 1)x_j + 2 \sum_{i=k_j+1}^{m_j} x_{j,i} \right), \tag{18}$$

where $x_j = x/c_j$ denotes the service requirement (in time) of the new job in queue j, and k_j corresponds to the number of jobs longer than x_j in queue j, $k_j = |\{x_{j,i} : x_{j,i} > x_j\}| = m_j - \max\{i : x_{j,i} < x_j\}$ with $x_{j,0} \triangleq 0$.

3.2.3. LWL-Δ

The size-aware relative value for an M/D/1-PS queue is given in (14), and therefore FPI can be carried out also in this case [22]. Starting from RND-ρ basic policy, one ends up with a policy which chooses the queue j minimizing a weighted sum of workload and the additional work,

$$\alpha(\mathbf{z}) = \operatorname*{argmin}_j (u_{\mathbf{z}_j} + (1/2) \cdot d_j), \tag{19}$$

where $u_{\mathbf{z}_j}$ denotes the total workload (backlog) in queue j, and d_j is the (constant) service time in queue j, $d_j = d/c_j$. The above policy belongs to a policy family defined by

$$\alpha(\mathbf{z}, x) = \operatorname*{argmin}_j (u_{\mathbf{z}_j} + \Delta \cdot x/c_j), \tag{20}$$

where Δ is a free policy parameter. For example, choosing the queue with the least amount work, referred to as π_U in [3] and the least-work-left (LWL) in [25,4], is obtained with $\Delta = 0$. We refer to all these policies as LWL-Δ, where $\Delta = 0.5$ (obtained with FPI) is generally a rather good choice for fixed size jobs and PS queues [22]. Note that applying the myopic criterion (18) for such a system yields

$$\alpha(\mathbf{z}) = \operatorname*{argmin}_j (d_j + 2u_{\mathbf{z}_j}),$$

which is equivalent to (19), i.e., in case of fixed size jobs, the myopic approach and the exact FPI step yield the same policy. Moreover, this is the optimal policy to dispatch fixed size jobs when the servers are identical [22]. In contrast, in [3] it is shown that LWL can have a rather poor performance when job sizes vary more.

3.3. Policy iteration with dispatching problem

In general, the knowledge of the relative values enables the FPI step of the Markov decision processes [26–28]. The basic idea in the policy iteration is to think that one deviates once from the default action of the basic policy, and then returns back to the basic policy for all later decisions. If the sum of an immediate cost and the relative value of the resulting state is less than the corresponding quantity with the default action, then, on average, the alternative action yields a lower infinite

horizon cost and thus choosing it improves the policy. Therefore, among the possible actions, the one with the smallest expected future costs is chosen. Repeating the same procedure for all states defines a new improved policy. The first policy iteration step typically yields the highest improvement toward the optimal policy.

For the size- and state-aware dispatching problem, a rather general state description is defined by $\mathbf{z} = (\mathbf{z}_1; \ldots; \mathbf{z}_q)$ where $\mathbf{z}_j = (n_j; x_{j,1}, \ldots, x_{j,m_j})$ denotes the state of queue j, n_j is the number of tasks with unknown exponentially distributed service requirement in queue j, and $x_{j,1} > x_{j,2} > \cdots > x_{j,m_j}$ are the remaining service requirements of the m_j known jobs currently present in queue j. The remaining known workload (backlog) in queue j is thus $u_{\mathbf{z}_j} = \sum_{i=1}^{m_j} x_{j,i}$. In a size- and state-aware system, all remaining service requirements are known and $n_j = 0$. However, e.g., JSQ policy expects only the queue occupations n_j, and therefore in that case $m_j = 0$ instead. In other words, the above general state description lends itself to assumptions different policies make.

When minimizing the mean sojourn time, there are no immediate costs, but instead, the costs are accrued at the state specific rates equal to the number of tasks in the system, $N(t)$. Consequently, for an arrival at state $\mathbf{z}$, the policy iteration step reduces to

$$\alpha(\mathbf{z}, x) = \operatorname*{argmin}_{\mathbf{z}' \in \mathcal{A}(\mathbf{z},x)} (v_{\mathbf{z}'} - v_{\mathbf{z}}), \tag{21}$$

where $\mathcal{A}(\mathbf{z}, x)$ denotes the set of possible destination states from state $\mathbf{z}$ having the new task with size x added to one of the q queues. The quantity $v_{\mathbf{z}}$ on the right-hand side is a common constant and thus could have been omitted. We have written it explicitly here, so that the quantity $v_{\mathbf{z}'} - v_{\mathbf{z}}$ corresponds to the expected *increase* in the cumulative sojourn time.

3.3.1. Separation with state-independent policy

With an arbitrary *state-independent policy*, the arrival process to each queue is Poissonian and the queues behave independently. Consequently, the relative value of state is the sum of the queue specific relative values,

$$v_{\mathbf{z}} = \sum_{j=1}^{q} v_{\mathbf{z}_j}.$$

Moreover, adding a task with size x to queue j does not change the state of the other queues (with a state-independent policy), and therefore, the change in the relative value is

$$w_j(x) \triangleq v_{\mathbf{z}'} - v_{\mathbf{z}} = v_{\mathbf{z}'_j} - v_{\mathbf{z}_j},$$

where $\mathbf{z}'$ and $\mathbf{z}'_j$ correspond to the resulting global state and the resulting state of queue j, respectively. Therefore, with a state-independent basic policy, the policy iteration step (21) becomes

$$\alpha(\mathbf{z}, x) = \operatorname*{argmin}_j \; w_j(x) = \operatorname*{argmin}_j (v_{\mathbf{z}'_j} - v_{\mathbf{z}_j}).$$

3.3.2. Size-oblivious FPI policy PS0

As mentioned, a different amount of information may be available at the dispatcher. For example, JSQ is the optimal policy, e.g., for identical servers with FCFS or PS assuming that only the occupation in each queue is known [1–3]. However, with heterogeneous servers, JSQ is obviously sub-optimal. Eq. (2) enables the FPI step, and using RND-ρ as the basic policy gives

$$\alpha(\mathbf{z}) = \operatorname*{argmin}_j \frac{n_j + 1}{c_j},$$

where $\mathbf{z} = (n_1, \ldots, n_q)$ with n_j denoting the number of tasks in queue j upon arrival. In practice, in case of ties it appears to be beneficial to favor a slower server, and thus for a very small $\epsilon > 0$, we define the PS0 policy by

$$\alpha^*(\mathbf{z}) = \operatorname*{argmin}_j \frac{n_j + 1 - \epsilon}{c_j}.$$

3.3.3. Size-aware FPI policy PS1

Next we assume that a dispatcher is aware of the size of the new task, denoted by x, and the number of active tasks in each queue but not their remaining workloads, so that $\mathbf{z} = (n_1, \ldots, n_q)$. This could be the case, e.g., when several dispatchers share a common pool of servers. In this case, (7) readily gives the required relative value of state for the FPI step. That is, the expected cost of accepting a new task of size x to queue j is

$$w_j(x) = v_{(n_j;x)} - v_{n_j} = \frac{1}{\mu c_j (1-\rho_j)^2} \left[\mu x + (2-\rho_j) \left(n_j - \frac{\rho_j}{1-\rho_j} \right) (1 - \mathrm{e}^{-\mu x (1-\rho_j)}) \right],$$

where we have utilized the identity $\mu_j x_j = \mu x$ that follows from (15). With RND-ρ as the basic policy, the ρ_j are equal and omitting the common constants in the $w_j(x)$ gives

$$w_j^*(x) = \frac{1}{c_j}\left[\mu x + (2-\rho)\left(n_j - \frac{\rho}{1-\rho}\right)\left(1 - \mathrm{e}^{-\mu x(1-\rho)}\right)\right] = \frac{1}{c_j}[A(x)\cdot n_j + B(x)].$$

We refer to the policy minimizing the above as the PS1 policy,

$$\alpha(\mathbf{z}, x) = \operatorname*{argmin}_j \; w_j^*(x).$$

In a symmetric case all servers have equal rates, $c_i = c_j$, and the above reduces further,

$$\alpha(\mathbf{z}, x) = \operatorname*{argmin}_j \; w_j^*(x) = \operatorname*{argmin}_j \; n_j,$$

i.e., in this case both PS0 and PS1 reduce to JSQ. This explains why JSQ has such a good performance with symmetric servers, as observed in [3,4]. Conversely, in the asymmetric cases with $c_i \neq c_j$ for $i \neq j$, PS1 assigns extremely long jobs categorically to the fastest server independently of the states of the queues.

3.3.4. Size- and state-aware FPI policies

However, in general we have assumed a size-aware system, where the dispatcher is aware of the (remaining) job sizes in each queue, $\mathbf{z} = (\mathbf{z}_1; \ldots; \mathbf{z}_q)$ with queue states $\mathbf{z}_j = (x_{j,1}, \ldots, x_{j,m_j})$. In this case, (13) allows the FPI step for an arbitrary state-independent basic policy. Here we consider two such policies, namely RND-ρ and RND-opt, yielding two new policies which we refer to as the FPI-ρ and FPI-opt, respectively. Formally, the improved policy is defined by

$$\alpha(\mathbf{z}, x) = \operatorname*{argmin}_j \; (v_{\mathbf{z}_j \oplus x/c_j} - v_{\mathbf{z}_j}), \tag{22}$$

where $\mathbf{z}_j \oplus x/c_j$ denotes the resulting state of queue j if the new task of size x is assigned to it. This policy generalizes also to cases where each queue j has n_j additional tasks with (unknown) exponentially distributed service requirements, as well as to cases where the size of the new job is not known, but obeys the same exponential distribution, cf. Eq. (13).

4. Numerical examples

In this section, we give numerical results for a two-server dispatching system. First we consider the case of exponentially distributed job sizes, and then, for comparison, also show the corresponding performance under constant job sizes. The dispatching policies evaluated are summarized in Table 1. Additionally, the performance is evaluated with an equivalent capacity *single-server system* with PS discipline. The single-server system serves as a reference for investigating the penalty due to splitting of the server capacity to independent servers. The primary server has a unit service rate $c_1 = 1$, and the secondary server is:

(i) equally fast; symmetric case, $c_2 = 1$,
(ii) two times slower; asymmetric case, $c_2 = 1/2$,
(iii) four times slower; highly asymmetric case, $c_2 = 1/4$.

In other words, the service rates are $c_1 = 1$ and $c_2 = 1,\ 1/2,\ 1/4$. The probability of selecting the faster primary server 1 according to RND-opt policy is illustrated in Fig. 5 for the three example cases. The secondary server is omitted from the solution in the asymmetric cases when the offered load is low, cf. Eq. (17).

4.1. Exponentially distributed job sizes

We start with exponentially distributed job sizes, $X \sim \mathrm{Exp}(1)$. The numerical results with RND, LWL-Δ, JSQ and Myopic dispatching policies are given in Fig. 6, and the corresponding results with the various FPI policies are depicted in Fig. 7. All policies are listed in Table 1. In general, we can observe that the Myopic policy seems to work very well in this case, as one can expect on the basis of the earlier analysis.

In the symmetric case of $c_1 = c_2 = 1$, several policies are in fact identical. First, RND-ρ and RND-opt have the same probability distribution, and consequently FPI-ρ and FPI-opt are identical. Second, JSQ, PS0 and PS1 are identical (see Section 3.3.3) and work reasonably well, as reported also in [3,4]. Third, the LWL policies are all identical and have a similar performance as JSQ. The size- and state-aware FPI-ρ/opt policy operates near the performance of the Myopic policy, but fails to supercede it as the best policy. However, the improvement from the starting point (RND-ρ/opt) is substantial. Recall that the FPI step yields always a better policy than the basic policy, unless the basic policy was already the optimal policy, in case of which the same policy is obtained. However, there is no guarantee that FPI yields a better policy than some other heuristic policy, such as the Myopic policy in this case.

In the asymmetric case with $c_2 = 1/2$, the observations are generally the same, albeit the difference between the Myopic policy and the FPI policies has become smaller. Also, FPI-opt performs slightly better than FPI-ρ, while, somewhat

Table 1
Dispatching policies evaluated in the numerical examples.

RND-ρ	State-independent random policy (Bernoulli splitting) with load balancing
RND-opt	State-independent random policy (Bernoulli splitting) minimizing the mean sojourn time
LWL-Δ	Least-work-left with $\Delta = 0,\ 1/2,\ 1$, and with an optimal Δ (determined each time); backlogs known
JSQ	Join-the-shortest-queue; queue occupancies known
Myopic	Minimize the sojourn time on condition that no further jobs arrive; all job sizes known, see (18)
FPI-ρ	First policy iteration on RND-ρ; queue occupancies and some/all job sizes known
FPI-opt	First policy iteration on RND-opt; queue occupancies and some/all job sizes known
PS0	First policy iteration on RND-ρ; queue occupancies known
PS1	First policy iteration on RND-ρ; queue occupancies and the size of the new job known

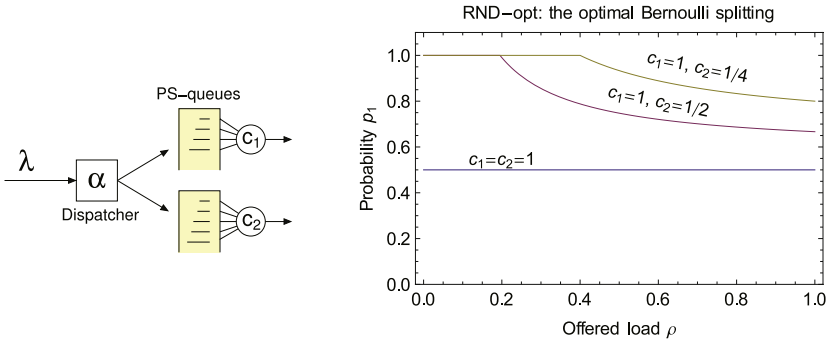

Fig. 5. The optimal Bernoulli splitting probabilities (RND-opt) according to (17) in the three example cases as a function of offered load ρ. With asymmetric servers, initially it is advantageous to route all tasks to the faster primary server ($p_1 = 1$). RND-opt and RND-ρ become identical when $\rho \to 1$.

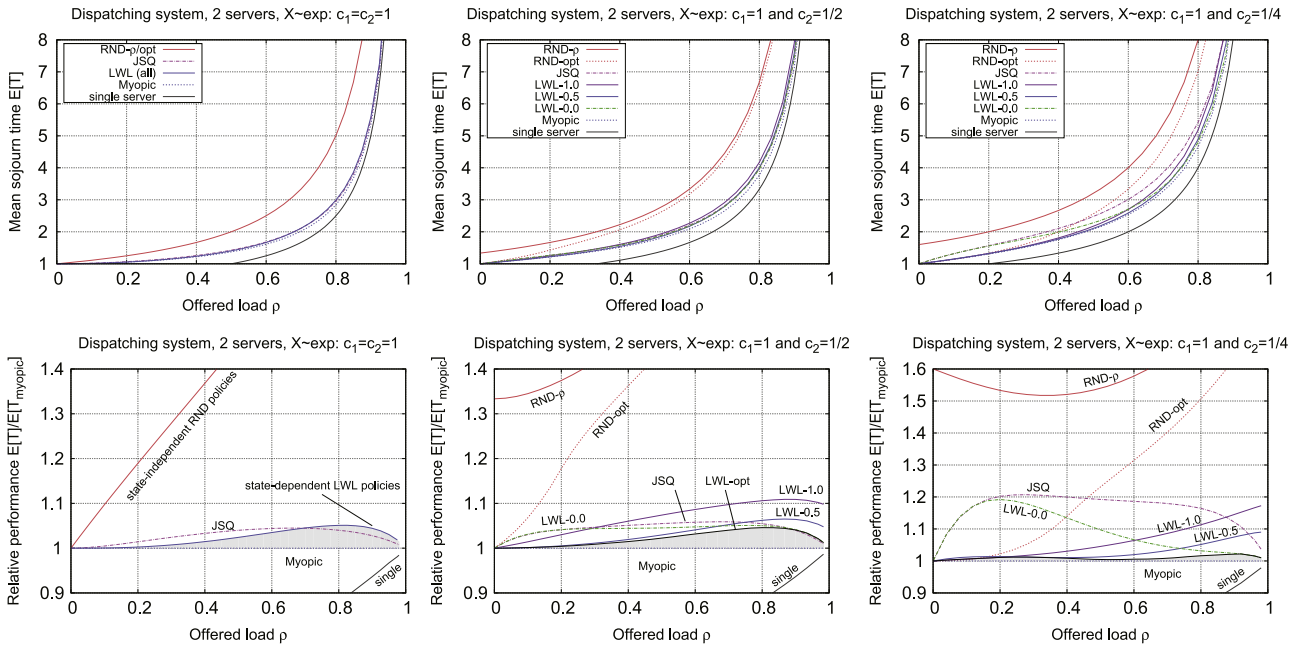

Fig. 6. Numerical results for a two-server dispatching system with heuristic policies and exponentially distributed job sizes, $X \sim \text{Exp}(1)$.

surprisingly, PS0 outperforms PS1 even though it relies on less information. Again, we note that FPI only guarantees an improvement over the basic policy.

In contrast, in the highly asymmetric case with $c_2 = 1/4$, the performance of JSQ has dropped significantly, the PS0 and PS1 policies based on the partial information still work reasonably well, while LWL-opt with an appropriately chosen parameter Δ achieves almost equal performance with the Myopic policy. More importantly, the FPI-opt policy manages to decrease the mean sojourn time clearly below that of the Myopic policy. The sudden change in the performance of FPI-opt at about $\rho = 0.4$ is due to the fact that until then the basic policy RND-opt has neglected the secondary server totally (see Fig. 5). State-dependent policy apparently utilizes the secondary server also when ρ is lower. This suggests that such an extremal starting distribution is disadvantageous for the FPI procedure, and further improvements can be expected with an appropriately chosen random basic policy.

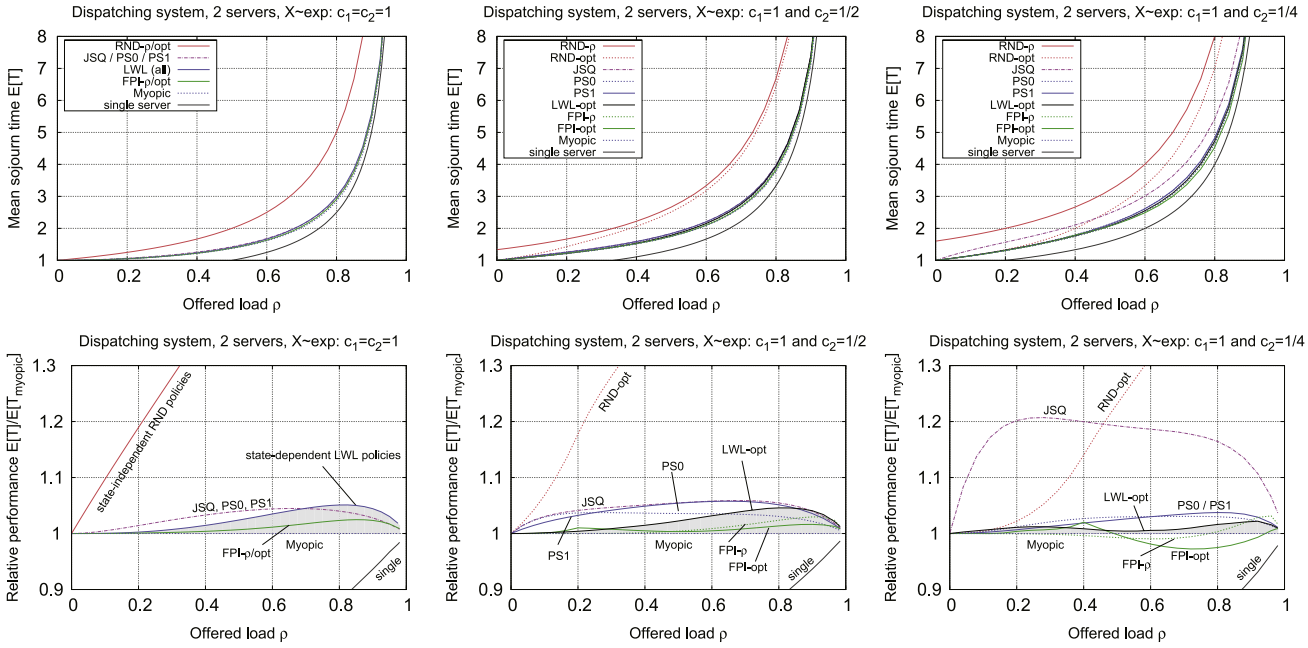

Fig. 7. Numerical results for a two-server dispatching system with FPI based policies and exponentially distributed job sizes, $X \sim \text{Exp}(1)$.

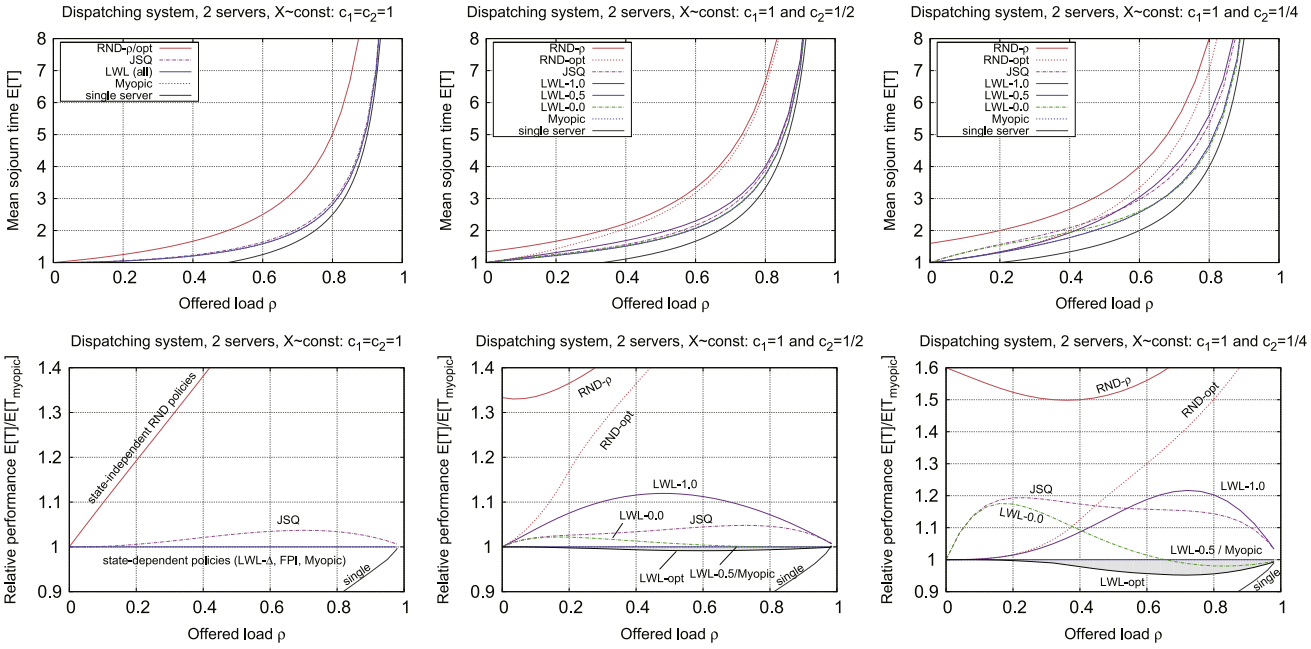

Fig. 8. Numerical results for a two-server dispatching system with constant job sizes.

In general, it seems that Myopic is a strong and robust dispatching policy for PS queues. By definition, it is optimal when $\lambda \to 0$. However, the "averaging" nature of the PS discipline makes it a good choice also otherwise. Only when servers differ sufficiently and the load is high, does FPI-opt (and also FPI-ρ) attain a lower mean sojourn time. The highest improvement in the three elementary cases is less than 5%. However, in more complicated settings the gain can be higher. For example, for five servers with service rates $c_1 = 1$ and $c_2, \ldots, c_5 = 1/4$, FPI-opt attains over 10% lower mean sojourn time than the Myopic policy. Regarding PS0 and PS1, it seems that announcing the size of the new job (PS1) does not help the dispatcher much if the remaining service requirements of the other jobs are not available. Again, the situation may be different in other settings.

4.2. Constant job sizes

For comparison, we also ran similar simulations with constant job sizes. Recall from Section 3.2.3, that in this case Myopic and LWL-0.5 (i.e., FPI) become identical policies. The numerical results are given in Fig. 8. One can observe that in the case of

highly asymmetric servers, LWL-Δ policy, with an appropriately chosen Δ, yields even smaller mean sojourn time than the Myopic policy. Hence, even though the Myopic policy works well in our numerical examples, it is not generally an optimal dispatching policy for PS queues (even among the heuristic policies considered). We note that for FCFS and LCFS disciplines, the size-aware relative value depends only on the mean job size, i.e., the relative value is insensitive with respect to the form of the job size distribution [21]. As mentioned earlier, this is no longer the case with the PS queue, albeit the mean sojourn time in an M/G/1-PS queue is insensitive to job size distribution.

5. Conclusions

In this paper, we have studied the processor sharing (PS) discipline in the context of a dispatching problem with heterogeneous servers and exponentially distributed job sizes. In such a system, arriving tasks are assigned to one of the available servers, which then processes the given tasks in parallel according to the PS discipline. In principle, a good dispatching decision requires that one takes into account also the future arrivals. We have approached this problem within the MDP framework, which provides a systematic methodology to find robust dispatching policies. In particular, we first derived the size-aware relative value with respect to the sojourn time in an M/M/1-PS queue, where the state information comprises the number of tasks and the remaining service requirements of some or all of them. The exact formula is obtained by solving an infinite system of ordinary differential equations. The result has an intriguing form, from which one can identify the already known (myopic) and the expected contributions. For the completely size-aware system with all job sizes known, it turns out that the minimum cumulative sojourn time of the present tasks (i.e., without further arrivals) can be used as a reasonable approximation for the relative value.

The knowledge of the relative values, or their approximations, enables the first policy iteration (FPI) step for an arbitrary state-independent random policy. As a result, one obtains efficient and robust state-dependent policies tailored for M/M/1-PS queues. The myopic approach focuses solely on the known jobs, and can be related to FPI by an approximation of the exact relative value. Its performance is good in the numerical examples due to the averaging nature of the PS discipline. The optimal Bernoulli split based FPI-opt policy manages to clearly outperform it only with highly asymmetric servers and reasonably high load. The differences among the FPI based policies are relatively small. In contrast, e.g., the performance of JSQ, which is known to be good in the symmetric case [3,4], is rather weak in the asymmetric settings. In addition to the dispatching problems, the relative values can also be utilized, e.g., with fair pricing schemes. The future work includes comprehensive analysis of the value of the available information, e.g., about the job sizes and scheduled arrivals.

References

[1] W. Winston, Optimality of the shortest line discipline, Journal of Applied Probability 14 (1977) 181–189.
[2] A. Ephremides, P. Varaiya, J. Walrand, A simple dynamic routing problem, IEEE Transactions on Automatic Control 25 (4) (1980) 690–693.
[3] F. Bonomi, On job assignment for a parallel system of processor sharing queues, IEEE Transactions on Computers 39 (7) (1990) 858–869.
[4] V. Gupta, M. Harchol-Balter, K. Sigman, W. Whitt, Analysis of join-the-shortest-queue routing for web server farms, Performance Evaluation 64 (9–12) (2007) 1062–1081.
[5] L. Kleinrock, Time-shared systems: a theoretical treatment, Journal of the ACM 14 (2) (1967) 242–261.
[6] S.F. Yashkov, Processor-sharing queues: some progress in analysis, Queueing Systems, Theory and Applications 2 (1) (1987) 1–17.
[7] E.G. Coffman Jr., R.R. Muntz, H. Trotter, Waiting time distributions for processor-sharing systems, Journal of the ACM 17 (1) (1970) 123–130.
[8] B. Sengupta, D. Jagerman, A conditional response time of M/M/1 processor-sharing queue, AT&T Bell System Technical Journal 64 (2) (1985) 409–421.
[9] L.E. Schrage, L.W. Miller, The queue M/G/1 with the shortest remaining processing time discipline, Operations Research 14 (4) (1966) 670–684.
[10] L. Schrage, A proof of the optimality of the shortest remaining processing time discipline, Operations Research 16 (3) (1968).
[11] 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, 1998, pp. 268–269.
[12] M. Harchol-Balter, M.E. Crovella, C.D. Murta, On choosing a task assignment policy for a distributed server system, Journal of Parallel and Distributed Computing 59 (1999) 204–228.
[13] H. Feng, V. Misra, D. Rubenstein, Optimal state-free, size-aware dispatching for heterogeneous M/G/-type systems, Performance Evaluation 62 (1–4) (2005) 475–492.
[14] C.E. Bell, J. Shaler Stidham, Individual versus social optimization in the allocation of customers to alternative servers, Management Science 29 (7) (1983) 831–839.
[15] M. Haviv, T. Roughgarden, The price of anarchy in an exponential multi-server, Operations Research Letters 35 (4) (2007) 421–426.
[16] E. Altman, U. Ayesta, B. Prabhu, Load balancing in processor sharing systems, Telecommunication Systems 47 (1) (2011) 35–48.
[17] B. Schroeder, M. Harchol-Balter, Evaluation of task assignment policies for supercomputing servers: the case for load unbalancing and fairness, Cluster Computing 7 (2) (2004) 151–161.
[18] K.R. Krishnan, Markov decision algorithms for dynamic routing, IEEE Communications Magazine (1990) 66–69.
[19] K.R. Krishnan, Joining the right queue: a state-dependent decision rule, IEEE Transactions on Automatic Control 35 (1) (1990) 104–108.
[20] S. Aalto, J. Virtamo, Basic packet routing problem, in: The Thirteenth Nordic Teletraffic Seminar NTS-13, Trondheim, Norway, 1996, pp. 85–97.
[21] E. Hyytiä, A. Penttinen, S. Aalto, Size-and state-aware dispatching problem with queue-specific job sizes (December 2010) (submitted for publication).
[22] E. Hyytiä, A. Penttinen, S. Aalto, J. Virtamo, Dispatching problem with fixed size jobs and processor sharing discipline, in: 23rd International Teletraffic Congress, ITC'23, San Fransisco, USA, 2011.
[23] L. Kleinrock, Queueing Systems, Volume I: Theory, Wiley Interscience, 1975.
[24] J. Virtamo, Lecture notes on Markov decision processes, 38.141 Teletraffic Theory, TKK, 2004.
[25] A. Sharifnia, Instability of the join-the-shortest-queue and FCFS policies in queuing systems and their stabilization, Operations Research 45 (2) (1997) 309–314.
[26] R. Bellman, Dynamic Programming, Princeton University Press, 1957.
[27] R.A. Howard, Dynamic Probabilistic Systems, Volume II: Semi-Markov and Decision Processes, Wiley Interscience, 1971.
[28] S.M. Ross, Applied Probability Models with Optimization Applications, Holden-Day Inc., 1970.

Esa Hyytiä received the M.Sc. (Tech.) degree in engineering physics and Dr.Sc. (Tech.) degree in electrical engineering from the Helsinki University of Technology, in 1998 and 2004, respectively. 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, and from 2005 to 2009, with the Telecommunication Research Center Vienna (ftw.), Austria, as a senior researcher. Currently, he is working as a senior research scientist at Aalto University, Finland. His research interests include performance analysis, design and optimization of communications systems.

Jorma Virtamo received the M.Sc. (Tech.) degree in engineering physics and D.Sc. (Tech.) degree in theoretical physics from Helsinki University of Technology, in 1970 and 1976, respectively. In 1986, he joined Technical Research Centre of Finland, VTT Information Technology, where he led the teletraffic research group, and became a Research Professor in 1995. In 1997, he was nominated as a Professor in the Department of Communications and Networking of Helsinki University of Technology, from which post he retired in 2009. His research interests include queueing theory and performance analysis of the Internet, multihop wireless networks and opportunistic content sharing.

Samuli Aalto received the M.Sc. and Ph.D. degrees in mathematics from University of Helsinki in 1984 and 1998, respectively. From 1984 to 1997, he was with VTT Technical Research Centre of Finland as a research scientist in the area of telecommunications. Since 1997, he has been with TKK Helsinki University of Technology, which is now part of Aalto University in Finland. Currently he acts as a temporary professor leading the Performance Analysis Group in the Department of Communications and Networking.

Aleksi Penttinen received his M.Sc. (Tech.) in systems and operations research and D.Sc. (Tech.) in teletraffic theory from Helsinki University of Technology in 2001 and 2006, respectively. He is currently working as a senior research scientist at the Department of Communications and Networking at the Aalto University School of Electrical Engineering. His research interests include green networking, performance analysis and optimization in data networks.