Dispatching Problem with Fixed Size Jobs and Processor Sharing Discipline

E. Hyytiä, A. Penttinen, S. Aalto and J. Virtamo

Aalto University, School of Electrical Engineering, Finland
email: firstname.lastname@tkk.fi

***Abstract*—We consider a distributed server system with m servers operating under the processor sharing (PS) discipline. A stream of fixed size tasks arrives to a dispatcher, which assigns each task to one of the servers. We are interested in minimizing the mean sojourn time, i.e., the mean response time. To this end, we first analyze an M/D/1-PS queue in the MDP framework. In particular, we derive a closed form expression for 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 result can be applied in numerous situations. Here we give an example in the context of dispatching problems by deriving efficient and robust state-dependent dispatching policies for homogeneous and heterogeneous server systems. The obtained policies are further demonstrated by numerical examples.**

***Index Terms*—dispatching problem, task assignment, M/D/1, processor sharing, sojourn times, MDP**

I. 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 work by Winston [1], Ephremides et al. [2], and others. However, e.g., web servers are better modelled as processor sharing (PS) queues, where several clients are served at the same time [3]. The PS queues [4] have a very convenient insensitivity result, i.e., 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 the expected sojourn times is difficult, see, e.g., [3], [5], [6], [7].

The optimal dispatching decision with respect to sojourn time depends on the available information. Often the number of tasks per server is assumed to be known, cf., e.g., JSQ. In contrast, in [8], [9] and [10], FCFS discipline is assumed and that the dispatching policy is aware of the size of the arriving job, but not about the state of the queues. In this setting, dispatching policy can be based on job size intervals (e.g., short jobs to one queue, and the rest to another).

In this paper, we consider a dispatching problem with fixed size jobs (per server) and processor sharing. The dispatcher is also aware of the states of each queue, i.e., the tasks and their remaining workloads. Such a system is illustrated in Fig. 1, and serves as an abstract model, e.g., for a file servers in a distributed content delivery network. File servers may have different link capacities, and the objective is to utilize them in an optimal manner so as to minimize the mean transfer time. Additionally, job dispatching in a distributed computing system with multitasking operating systems can be seen as a dispatching problem with processor sharing discipline.

First we study a single M/D/1-PS queue in isolation and derive an important result giving a relative value of state, which essentially characterizes the value of queue state with respect to the expected sojourn times in infinite time horizon.

Then we apply these results to some example dispatching problems, and demonstrate how one can determine efficient and robust state-dependent dispatching policies in a straightforward fashion in Markov decision process (MDP) framework. In particular, starting from an arbitrary state-independent policy, carrying out the first policy iteration (FPI) step yields an improved state-dependent policy. The knowledge of the relative values is a prerequisite to this end. For example, the so-called least-work-left policy, which chooses the queue with the least amount of unfinished work, turns out to perform well in the example cases. However, even better performance is obtained by an FPI based policy.

Similar approach has been previously used by Krishnan in the context of routing calls in a telephone network [11] so as to minimize the blocking probability, and in the context of a dispatching problem for parallel M/M/s-FCFS servers [12] so as to minimize the mean sojourn time. With respect to the mean sojourn time, the traditional M/M/1-FCFS queue has been also analyzed in [13]. Recently, FCFS and also last-come-first-served (LCFS), shortest-processing-time (SPT) and shortest-remaining-processing-time (SRPT) disciplines with arbitrary service time distribution were analyzed in [14]. The join-the-shortest (JSQ) queue with processor sharing discipline has been considered by Gupta et al. in [3], but as mentioned, transient analysis of a general M/G/1-PS queue is difficult, and therefore, similarly as in [7], we also limit ourselves to consider the M/D/1-PS queue.

The rest of this paper is organized as follows. In Section II, we analyze a single M/D/1-PS queue, and derive a closed form expression for the so-called size-aware relative value of state with respect to the sojourn time. In Section III, the theoretical results are applied to the dispatching problem within the MDP framework. Section IV contains the conclusions.

978-0-9836283-0-9

This paper was peer reviewed by subject matter experts for publication in the Proceedings of ITC 2011

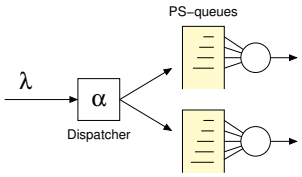

Fig. 1. Dispatching system with processor sharing (PS) queues.

II. Analysis of an M/D/1-PS Queue

In this section, we analyze a single M/D/1-PS queue with respect to the sojourn time. The optimization objective is to minimize the mean sojourn time $\mathrm{E}[T]$. According to Little's result, $\mathrm{E}[N] = \lambda\,\mathrm{E}[T]$, and thus defining the cost rate as the number of tasks in the system, N, gives us an equivalent cost minimization problem.

Let $\mathbf{z} = (\Delta_1; ..; \Delta_n)$ denote the state of an M/D/1-PS queue, where $\Delta_1 \geq \Delta_2 \geq \ldots \geq \Delta_n$ denote the remaining workloads of the n tasks (measured in time) in the order of arrival times (task n being the oldest). Our aim is to quantify the difference in the expected cumulative sojourn times in infinite time horizon between two states. In particular, we are interested in to find out the so-called relative values of states, denoted by $v_{\mathbf{z}}$, which are defined as the expected difference in infinite time horizon between a system in a given state $\mathbf{z}$, and a system initially in equilibrium,

$$v_{\mathbf{z}} := \lim_{t\to\infty} \mathrm{E}[V_{\mathbf{z}}(t) - r \cdot t],$$

where r denotes the mean cost rate and random variable $V_{\mathbf{z}}(t)$ the cumulative costs (sojourn time) during $(0, t)$ when system is initially in state $\mathbf{z}$. In our case, $V_{\mathbf{z}}(t)$ increases at rate $N_{\mathbf{z}}(t)$ and the mean cost rate corresponds to the mean occupation, $r = \mathrm{E}[N]$. In a stable system, $v_{\mathbf{z}}$ is finite and well-defined, and $v_{\mathbf{z}_1} - v_{\mathbf{z}_2}$ characterizes the expected difference in the future costs (in sojourn time) between two initial states $\mathbf{z}_1$ and $\mathbf{z}_2$.

Let λ denote the Poissonian arrival rate and d the constant job size, so that for all states $\mathbf{z} = (\Delta_1; ..; \Delta_n)$ it holds that $\Delta_i \leq d\ \forall i$. Moreover, $\rho = \lambda d$ is the offered load, $\rho < 1$, and $u_{\mathbf{z}}$ the total remaining workload, $u_{\mathbf{z}} = \sum_{i=1}^{n} \Delta_i$ measured in time. The following elementary relation holds for the difference in relative values between state $\mathbf{z}$ and an empty system:

Proposition 2.1 (M/D/1-PS): The size-aware relative value of state $\mathbf{z}$ with respect to the sojourn time in an M/D/1 queue with a processor sharing (PS) discipline is given by

$$v_{(\Delta_1;..;\Delta_n)} - v_0 = \frac{\lambda}{1-\rho} u_{\mathbf{z}}^2 - u_{\mathbf{z}} + 2\sum_{i=1}^{n} i\,\Delta_i. \qquad (1)$$

where v_0 denotes the relative value of an empty system.

Proof: First, consider an arbitrary state $\mathbf{z} = (\Delta_1; ..; \Delta_n)$ and assume that no new tasks arrive during their service time of $u_{\mathbf{z}}$. By definition, the n tasks are in decreasing order, $\Delta_1 \geq \Delta_2 \geq \ldots \geq \Delta_n$, and the cumulative sojourn time accrued during time $(0, u_{\mathbf{z}})$ is given by

$$\tau_{\mathbf{z}} = \Delta_n \cdot n^2 + (\Delta_{n-1} - \Delta_n) \cdot (n-1)^2 + \ldots + (\Delta_1 - \Delta_2),$$

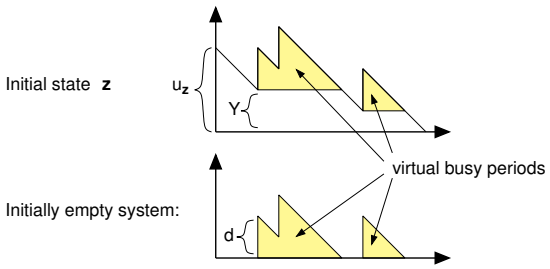

Fig. 2. Difference in the accrued sojourn times between System 1 initially at state $\mathbf{z}$, and System 2 initially empty.

which gives

$$\tau_{\mathbf{z}} = \sum_{i=1}^{n} (2i-1)\Delta_i. \qquad (2)$$

Second, with constant service times, an arriving job has more remaining workload than those already in the system, and thus the sojourn time of new arrivals depends only on the total workload in the queue upon an arrival. Consider next a new arrival at time t. A direct application of (2) gives that the total sojourn time *already scheduled* to be incurred increases by an amount of

$$s_{\mathbf{z}} = 2\,u_{\mathbf{z}} + d, \qquad (3)$$

where $u_{\mathbf{z}}$ denotes the remaining workload just before an arrival in state $\mathbf{z}$. Consequently, one can say that each arrival has an *immediate cost* of $2\,u_{\mathbf{z}} + d$, independently of what type of tasks the total workload $u_{\mathbf{z}}$ consists of, which is somewhat unexpected in the context of PS queues. For comparison, with FCFS the immediate cost is $u_{\mathbf{z}} + d$.

Then we refer to Fig. 2, which illustrates the difference in the accrued sojourn times between System 1 starting from state $\mathbf{z}$ and System 2 initially empty. For each arrival realization, both systems behave identically once System 1 becomes idle and thus it is sufficient to consider the remaining busy period of System 1. The mean number of mini busy periods (shaded areas) is equal to $\lambda\,u_{\mathbf{z}}$. During each busy period, on average $1/(1-\rho)$ tasks are served (general result for M/G/1 queues [15]). Each mini busy period is also stochastically independent and identical with respect to the total additional workload (shaded areas). Moreover, the remaining workload Y_i upon a start of mini busy period i is uniformly and independently distributed on $(0, u_{\mathbf{z}})$, $Y_i \sim \mathrm{U}(0, u_{\mathbf{z}})$, (a property of a Poisson process), and thus for the mean we have $\mathrm{E}[Y_i] = u_{\mathbf{z}}/2$. Finally, (3) means that the difference in costs per arriving job during each mini busy period is two times the offset Y_i, i.e., on average $2 \cdot u_{\mathbf{z}}/2 = u_{\mathbf{z}}$. Therefore,

$$\begin{aligned} v_{\mathbf{z}} - v_0 &= \sum_{i=1}^{n} (2i-1)\Delta_i + \lambda u_{\mathbf{z}} \cdot \frac{1}{1-\rho} \cdot u_{\mathbf{z}} \\ &= \frac{\lambda}{1-\rho} u_{\mathbf{z}}^2 - u_{\mathbf{z}} + 2\sum_{i=1}^{n} i\,\Delta_i. \end{aligned}$$

■

Corollary 2.2 (Cost of a new task): The mean cost due to accepting a new task at state $\mathbf{z}$ is given by

$$c_{\mathbf{z}} = v_{(d;\Delta_1;..;\Delta_n)} - v_{(\Delta_1;..;\Delta_n)} = \frac{2\,u_{\mathbf{z}} + d}{1-\rho}. \qquad (4)$$

The above result is surprisingly compact and again independent of the underlying constellation of remaining workloads Δ_i apart from the total workload $u_{\mathbf{z}}$. The simplicity of the result basically stems from the assumption of a constant service time. We note that the known costs (i.e., assuming that no further task arrives) are $2\,u_{\mathbf{z}} + d$, and the denominator $1-\rho$ "takes into account" the future arrivals. For an arbitrary M/G/1-PS queue we have:

Lemma 2.3: The cost in terms of an additional sojourn time an arriving task brings in to an M/G/1-PS queue is on average

$$\mathrm{E}[S] = \frac{\mathrm{E}[X]}{1-\rho}. \qquad (5)$$

where $\mathrm{E}[X]$ denotes the mean job size and $\rho = \lambda\,\mathrm{E}[X]$ is the offered load.

A proof follows trivially from noting that $\mathrm{E}[S] = \mathrm{E}[T]$, i.e., the mean sojourn time. For an M/D/1-PS, (3) gives $\mathrm{E}[S] = 2\,\mathrm{E}[U] + d$, which, as a cross-check, can be shown to satisfy (5) with aid of Pollaczek-Khinchine M/G/1 formula (see, e.g., [15], [16]), which gives $\mathrm{E}[U] = \lambda\,\mathrm{E}[X^2]/(2(1-\rho))$. Whereas the relative value comprises sojourn times of the existing and future jobs, one may also be interested in the mean conditional sojourn time of a given job. To this end, we refer to [6].

Finally, we note that the difference of the relative values in an M/D/1-PS queue, given by (1), has a close resemblance to that obtained for an M/G/1-FCFS queue: ([14], [13])

$$v_{(\Delta_1;..;\Delta_n)} - v_0 = \frac{\lambda}{2(1-\rho)}\,u_{\mathbf{z}}^2 + \sum_{i=1}^{n} i\,\Delta_i.$$

The fact that in both systems the tasks depart in the same order as they have arrived naturally contributes to this.

III. Application to Dispatching Problem

In general, the knowledge of the relative values enables the first policy iteration (FPI) step of Markov decision processes [17], [18], [19]. The basic idea is to think that one deviates once from the default action of the basic policy α, and then returns back to α 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 (note that relative values describe the expected future costs), then, on average, the alternative action yields a lower cost and thus 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 α', that is often close to the optimal policy in practice. Indeed, in Section III-B we see how FPI yields an optimal dispatching policy for identical servers, while in [14] it is shown that for M/G/1-FCFS and M/G/1-LCFS queues carrying out the FPI step on the service order yields the optimal SRPT service order.

A. Policy Iteration with Dispatching Problem

In a dispatching problem, a stream of tasks arrive at a dispatcher, which then forwards them to one of the servers. Let m denote the number of servers, and $\mathbf{z}$ the system's state, $\mathbf{z} = (\mathbf{z}_1;..;\mathbf{z}_m)$ where $\mathbf{z}_i = (\Delta_{i,1};..;\Delta_{i,n_i})$, so that the remaining workload in queue i is $u_i(\mathbf{z}) = \sum_{j=1}^{n_i}\Delta_{i,j}$.

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. Consequently, for an arrival at state $\mathbf{z}$, the policy iteration step reduces to

$$\alpha'(\mathbf{z}) = \underset{\mathbf{z}'\in\mathcal{A}(\mathbf{z})}{\operatorname{argmin}}\ \left(v_{\mathbf{z}'} - v_{\mathbf{z}}\right),$$

where $\mathcal{A}(\mathbf{z})$ denotes the set of possible destination states from state $\mathbf{z}$ having one additional fixed size task in one of the m queues. Note that $v_{\mathbf{z}}$ 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.

B. Dispatching problem with homogeneous servers

Consider first a traditional dispatching system, where a single stream of tasks is served by a server farm consisting of m *homogeneous servers* each with own queue. The constant size jobs arrive according to a Poisson process with rate λ, so that the offered load $\rho = \lambda\,d$.

A random basic policy, known as *Bernoulli splitting* [2], selects the server independently for each arriving task using a given probability distribution $(p_1,\ldots,p_m)$. With such a policy, the *arrival process to each server remains Poissonian*, and each server behaves according to an M/D/1-PS queue with $\lambda_i = p_i\cdot\lambda$. The random policy is *state-independent*, by which we mean that the dispatching decision does not depend on the state of the queues or past decisions. In this case, as the arrival process to each queue is a Poisson process, the queues can be analyzed independently in isolation, (4) gives the relative value for each queue, and consequently, also for the whole system (sum of m independent components),

$$v_{\mathbf{z}} = \sum_{i=1}^{m} v_{\mathbf{z}_i}.$$

Hence, the first policy iteration step can be carried out for an arbitrary Bernoulli splitting yielding a new improved state-dependent policy:

$$\alpha'(\mathbf{z}) = \underset{i=1,..,m}{\operatorname{argmin}}\ \left(v_{\mathbf{z}'_i} - v_{\mathbf{z}_i}\right),$$

where $\mathbf{z}'_i$ denotes the new state of queue i if the given task is assigned to it, $\mathbf{z}'_i = (d;\Delta_{i,1};..;\Delta_{i,n_i})$. The quantity $v_{\mathbf{z}'_i} - v_{\mathbf{z}_i}$ is the expected increase in the cumulative sojourn time accrued in queue i, while the other queues remain unchanged.

Definition 3.1 (RND-U): The random policy RND-U is state-independent and assigns tasks uniformly in random to m queues, $p_i = 1/m,\ \forall\, i$.

Definition 3.2 (Least-work-left): Applying the first policy iteration step on the RND-U policy yields a so-called *least-work-left* (LWL$^-$) policy ([20], [3]),

$$\alpha_{\text{lwl}^-}(\mathbf{z}) = \underset{i}{\operatorname{argmin}}\, u_i(\mathbf{z}),$$

where $u_i(\mathbf{z})$ denotes the workload in queue i at state $\mathbf{z}$. That is, the FPI-U, i.e., LWL$^-$, policy simply chooses the queue with the smallest workload (backlog), which is obviously a rational but also *the optimal decision with homogeneous servers*.[1] Moreover, by sample path arguments, it is easy to show that for an initially empty system, the decisions that LWL$^-$ and round-robin make are essentially the same. Note that ties occur only when two or more queues are empty, in case of which those decisions are equal. The superscript "-" stands for *a priori* workload, i.e., the workload before the new task enters a queue. Here it makes no difference if one considers workloads before or after the arrival as the job size is a common constant d for all servers. However, in a heterogeneous system this is no longer the case.

C. Dispatching problem with heterogeneous servers

Consider next a dispatching system with m *heterogeneous PS-servers*, so that d_i denotes the fixed service time if a task is assigned to server i. That is, if x denotes a constant job size and c_i the rate of server i, then $d_i = x/c_i,\ \forall\, i$. For a random policy with server selection probability distribution $(p_1, \ldots, p_m)$, the stability conditions are

$$p_i\, \lambda\, d_i < 1,\ \forall\, i.$$

For $m = 2$ servers, the above reduces to

$$1 - \frac{1}{\lambda\, d_2} < p_1 < \frac{1}{\lambda\, d_1}.$$

Similarly as with the homogeneous servers, with state-independent random policies, the arrival process to each queue remains as a Poisson process and (4) can be applied.

One rational and robust random policy is to assign jobs with such probabilities that the offered load is uniformly distributed among the servers:

Definition 3.3 (RND-ρ): The random with uniform load policy, RND-ρ, balances the load, $\rho_i = \rho_j\ \forall\, i, j$, by assigning tasks in random using probabilities,

$$p_j = \frac{1/d_j}{\sum_{i=1}^{m} 1/d_i}.$$

When $d_i = d_j$, then $p_j = 1/m$ and the RND-ρ policy reduces to the RND-U policy. For example, for $m = 2$ we have

$$p_1 = \frac{d_2}{d_1 + d_2}, \quad \text{and} \quad p_2 = \frac{d_1}{d_1 + d_2}.$$

[1]Costs due to an arrival according to (3) hold with an arbitrary arrival process. With identical servers, d is a common constant and a dispatching policy α that is optimal for FCFS, is optimal also for PS, and vice versa. Moreover, LWL$^-$ with FCFS queues is equivalent to an M/G/m multi-server system with a central FCFS queue [9], from which the optimality of LWL$^-$ with either FCFS or PS queues for fixed size jobs trivially follows.

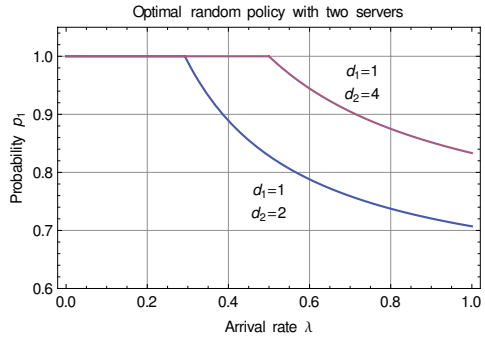

Fig. 3. Optimal random policy with two servers when i) $d_1 = 1$ and $d_2 = 2$, and ii) $d_1 = 1$ and $d_2 = 4$. The y-axis corresponds to probability of choosing the faster server 1. When λ is small all tasks are assigned to server 1.

RND-ρ policy can sustain the maximal arrival rate,

$$\lambda_{\max} = \sum_{i=1}^{m} 1/d_i.$$

One can also consider random policies having unequal loads. Especially when ρ is clearly less than the capacity of the system, it makes sense to favour the faster server(s). In particular, a straightforward application of the mean sojourn time result of M/G/1-PS queues [15],

$$\mathrm{E}[T] = \frac{\mathrm{E}[X]}{1 - \rho}, \tag{6}$$

allows one to determine the optimal random policy:

Definition 3.4 (RND-opt): The optimal random policy, referred to as RND-opt, is defined by a probability distribution $(p_1, \ldots, p_m)$ that minimizes the mean sojourn time

$$\mathrm{E}[T] = \sum_{i=1}^{m} p_i \cdot \frac{d_i}{1 - p_i \cdot \lambda\, d_i}.$$

For $m = 2$ servers, with aid of (6), minimizing the sum $p \cdot \mathrm{E}[T_1] + (1 - p) \cdot \mathrm{E}[T_2]$ gives for the p_1 of the optimal random policy *RND-opt*,

$$p_1 = \max\{0, \min\{1, p^*\}\}, \quad p^* = \frac{\sqrt{d_2} - \sqrt{d_1} + \lambda\sqrt{d_1}d_2}{\lambda\sqrt{d_1}\sqrt{d_2}(\sqrt{d_1} + \sqrt{d_2})}.$$

Fig. 3 illustrates the optimal random routing in two example cases with $m = 2$ servers: i) $(d_1, d_2) = (1, 2)$, and ii) $(d_1, d_2) = (1, 4)$. In both cases, when arrival rate λ is small, the secondary slower server is not used at all, and the faster the secondary server is, the sooner it is taken into use.

As mentioned, with state-independent random policies, the resulting arrival process to each queue remains Poissonian, and thus (4) also holds. Applying the FPI step then immediately gives a new state-dependent policy:

Definition 3.5 (FPI-ρ): The first policy iteration step on RND-ρ basic policy yields,

$$\alpha_{\text{fpi-}\rho}(\mathbf{z}) = \underset{i}{\operatorname{argmin}}\ \left(u_i(\mathbf{z}) + d_i/2\right),$$

which we refer to as the *FPI-ρ* policy.

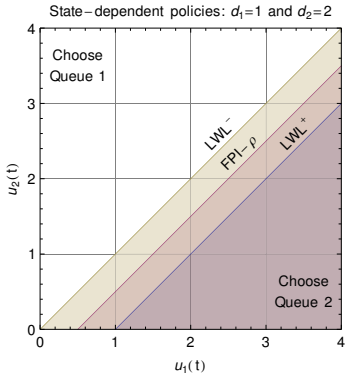

Fig. 4. The state-dependent policies LWL$^-$, LWL$^+$, and FPI-ρ are of the switch-over type, i.e., they are defined by a switch-over curve in (u_1, u_2)-plane (2 servers), which in this case is simply a straight line.

Obviously, when $d_i = d_j$, the FPI-ρ policy reduces to the LWL$^-$ policy, as RND-ρ reduces to RND-U. With heterogeneous servers, ties are resolved in favour of a faster server (cf. an empty system).

In the heterogeneous case, one can also define an LWL policy by considering *a posteriori* workloads:

Definition 3.6 (LWL$^+$): The LWL$^+$ (*a posteriori*) policy assigns a new task to the server which workload, including the new task, is the smallest:

$$\alpha_{\text{lwl}^+}(\mathbf{z}) = \underset{i}{\text{argmin}} \ (u_i(\mathbf{z}) + d_i) \, .$$

The LWL$^+$ (*a posteriori*) policy is different from FPI-ρ, as the FPI-ρ policy has a half smaller weight on parameter d_i. Similarly, LWL$^-$ (*a priori*) can be seen as a weighted sum with a zero weight on parameter d_i. Thus, the FPI-ρ policy is a kind of compromise between the two LWL policies.

It is also worth noting that RND-U, RND-ρ, FPI-ρ, and the two LWL policies are *independent of the arrival rate* λ, which is itself a desired property improving the robustness. Had we chosen to use, e.g., RND-opt or RND-U as a basis for the first policy iteration in a heterogeneous case, then the resulting FPI policy would also depend on the arrival rate, making it somewhat less practical (the value of λ must be typically estimated, and it can also vary as a function of time). Moreover, RND-U has a limited capacity region in case of heterogeneous servers, which makes it a weak choice for the FPI. Also the optimal state-independent policy RND-opt depends on the arrival rate λ.

The state-dependent (but arrival rate independent) policies LWL$^-$, LWL$^+$ and FPI-ρ, are of the *switch-over* type. Fig. 4 illustrates the corresponding decision criterion in (u_1, u_2)-plane for the case when $d_1 = 1$ and $d_2 = 2$. Both axes represent the remaining workload in the corresponding queue measured in time. With known job sizes and state-dependent policy, one effectively selects a certain subspace of the state space, where the system will operate. For example, with the FPI-ρ policy, the Fig. 4 system's state-space reduces to the area bounded by two straight lines, $u_2 \leq u_1 + 2$ and $u_2 \geq u_1 - 1$.

D. Numerical examples

Next we illustrate the derived dispatching policies by considering homogeneous and heterogeneous two server systems.

a) Homogeneous case: Let us start with the most elementary example and consider a system with two identical servers with $d = 1$. In this case RND-ρ and RND-opt reduce to RND-U. Similarly, LWL and FPI-U are equivalent. Fig. 5 illustrates the resulting performance with i) the optimal state-indepent policy RND-U, ii) the optimal state-dependent policy LWL choosing the shorter queue (in terms of processing time), and iii) an equivalent single server system with the PS discipline and job size of $d = 0.5$. Left figure depicts the absolute performance in terms of the mean sojourn time, and the right figure, the relative performance against the optimal state-dependent policy LWL. We can observe, that LWL works fairly well, and, e.g., at the limit $\rho \to 1$, its performance is similar to that of the equivalent single server system. Moreover, the gain from having the queue state information available increases "linearly" as a function of offered load ρ, and the mean sojourn time with the state-independent policy is about 2 times higher than with the optimal policy.

b) Moderately asymmetric case: Numerical results with six policies in a two server system with $d_1 = 1$ and $d_2 = 2$ are illustrated in Fig. 6. The two random policies (RND-ρ and RND-opt) are stateless and thus preserve the Poisson nature in the arrival process. The random policy RND-U is a rather poor choice with heterogeneous servers. In this case, it becomes unstable as $\lambda \to 1$ and the queue length in server 2 explodes, and thus we have omitted it. The other two random policies have the maximal stability region, $\lambda_{\max} = 1.5$, which equals $\rho = 1$ in the figure. At the limit $\lambda \to \lambda_{\max}$, RND-$\rho$ and RND-opt also become identical.

The two LWL policies and the two FPI policies are state-dependent policies, and assume that the information regarding the workload in each queue is available. The LWL$^+$ (*a posteriori*) policy is somewhat interestingly worse than the (*a priori*) variant. However, the best performance is obtained by the FPI-ρ policy, which offers a slightly better performance than LWL$^-$ (*a priori*).

We conducted also numerical experiments with the policy family $\mathcal{P}(\beta)$, defined by cost functions of form

$$c_i(\mathbf{z}) = u_i(\mathbf{z}) + \beta \cdot d_i,$$

where β is a free policy parameter, $0 \leq \beta \leq 1$. That is, LWL$^-$ corresponds to $\beta = 0$, FPI-ρ to $\beta = 1/2$, and LWL$^+$ to $\beta = 1$. It turns out that $\beta = 1/2$, i.e., the FPI-ρ policy, is practically the optimal choice in this case.

c) Strongly asymmetric case: Fig. 7 shows the results in otherwise same setting except that here $d_2 = 4$, i.e., the secondary server is four times slower than the primary. Here the situation gets clearly more interesting. Both LWL policies are clearly suboptimal with certain values of λ: with small values of λ, LWL$^-$ is weak, and with higher values the performance of LWL$^+$ deteriorates. The FPI-ρ policy, instead, is capable of making very good dispatching decisions independently of the arrival rate λ.

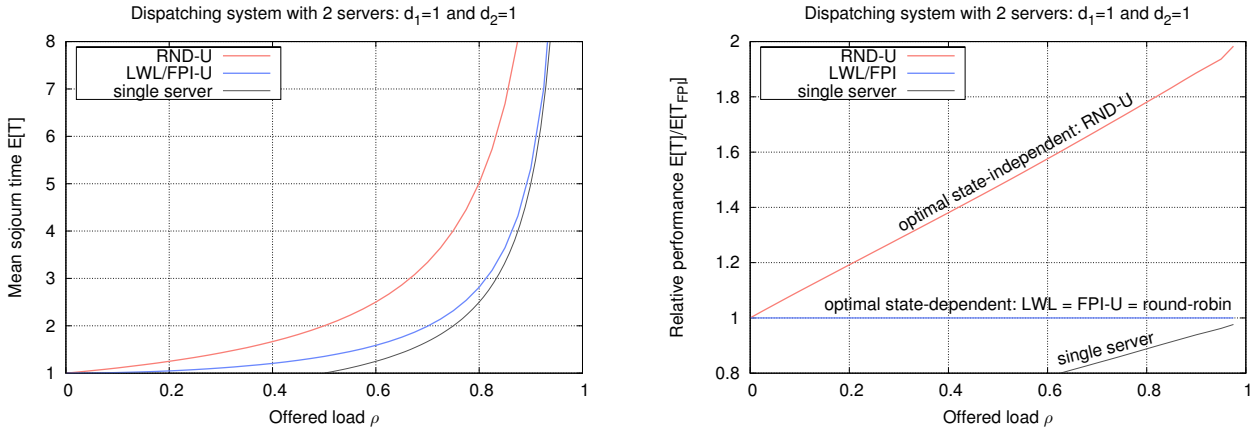

Fig. 5. On left, the resulting mean sojourn time with two homogeneous servers. The optimal state-independent policy RND-U splits the arrival stream evenly between the 2 servers, while the optimal state-dependent policy, LWL^- = LWL^+ = FPI-U, chooses the queue with a smaller workload. The performance of an equivalent single server system is also illustrated (the lowest gray curve). On right, the relative performance against the LWL policy is illustrated.

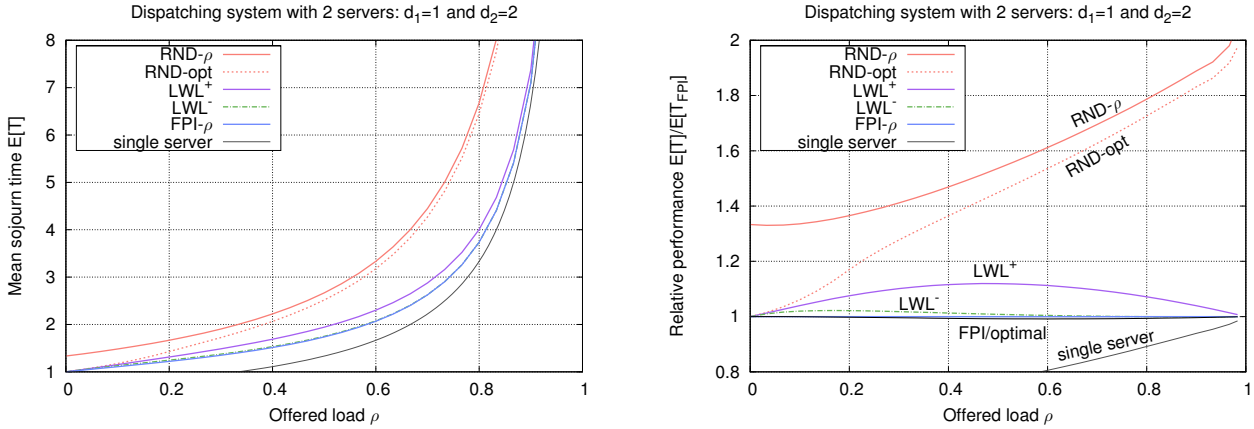

Fig. 6. On left, the resulting mean sojourn time with five policies (from worst to best): random RND-ρ, (i.e., equal load) random RND-opt (i.e., optimal split), LWL^+ (*a posteriori*), LWL^- (*a priori*), and FPI-ρ based on RND-ρ. The lowest gray curve corresponds to an equivalent single server system with capacity $c = 1.5$. On right, the relative performance against the FPI-ρ policy.

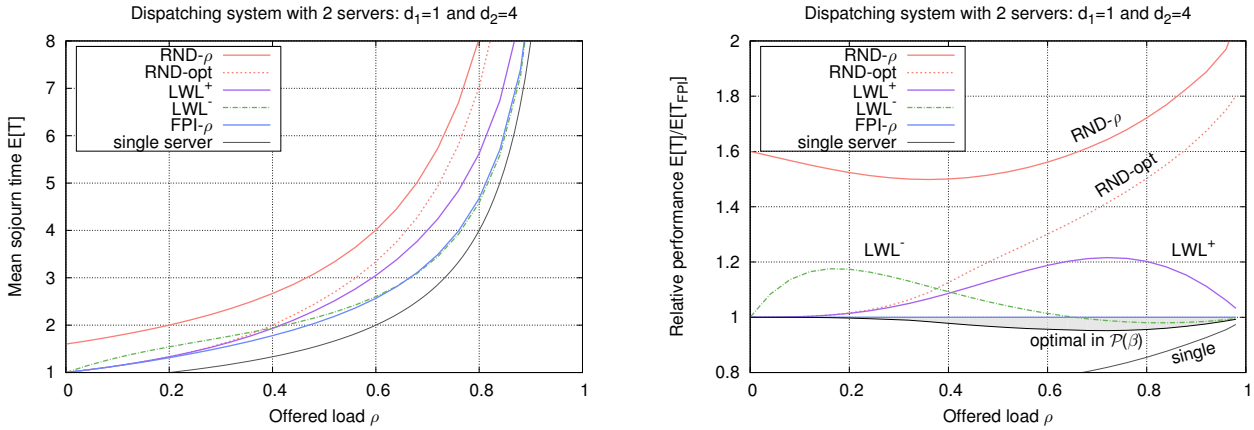

Fig. 7. On left, the resulting mean sojourn time with the five dispatching policies in relatively asymmetric case with $d_1 = 1$ and $d_2 = 4$. The lowest gray curve corresponds to an equivalent single server system with capacity $c = 1.25$. On right, the relative performance against the FPI-ρ policy. The gray area below reference level $y = 1$ corresponds to the performance of an optimal policy from the policy family $\mathcal{P}(\beta)$ defined by the cost functions of form $u_i(\mathbf{z}) + \beta \cdot d_i$. For each value of ρ, the optimal value for β has been determined.

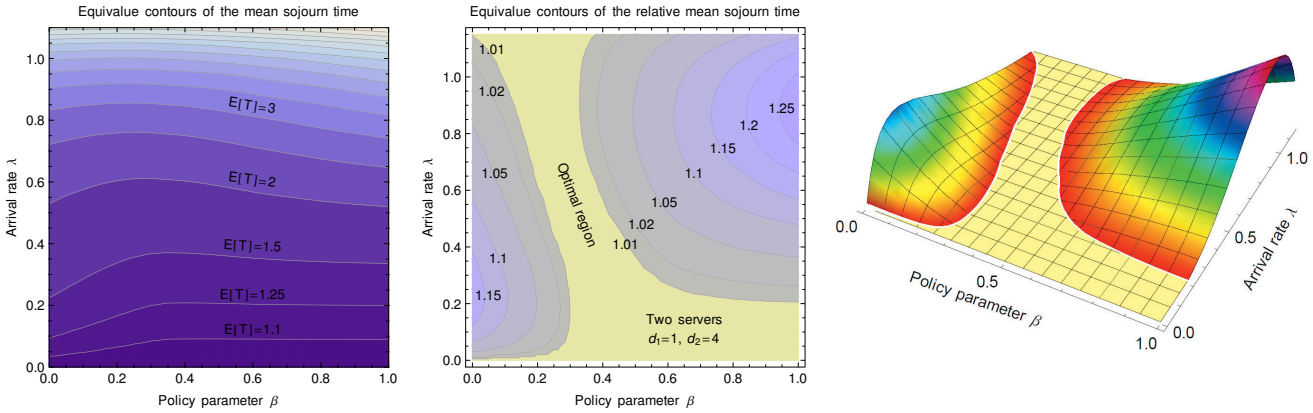

Fig. 8. On left, the resulting mean sojourn time $\mathrm{E}[T(\lambda,\beta)]$ for the policy family $\mathcal{P}(\beta)$ in a two server dispatching system with service times $d_1 = 1$ and $d_2 = 4$. On x-axis is the policy parameter β, and on y-axis the arrival rate λ. The figures on the middle and right illustrate the relative performance defined by $\mathrm{E}[T(\lambda,\beta)]/\min_{\beta'} \mathrm{E}[T(\lambda,\beta')]$. The "valley" corresponds to the optimal region where the mean sojourn time is within 1% from the minimum at the given rate λ. We can observe that with FPI-ρ, corresponding to $\beta = 0.5$, the performance remains within 5% from the optimal (within this policy family).

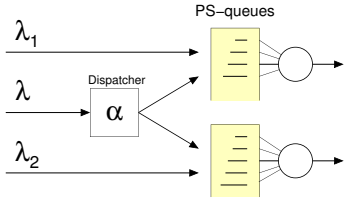

Fig. 9. Dispatching system with processor sharing (PS) queues, where each server also receives a dedicated stream of jobs.

Optimizing within the policy family $\mathcal{P}(\beta)$ yields a marginal improvement in this case. The gray area under the reference level $y = 1$ in Fig. 7 (right) corresponds to the lowest mean sojourn time achievable with an optimal choice of $\beta = \beta(\rho)$. The highest gain is obtained about $\rho = 0.7$, where the optimal policy from $\mathcal{P}(\beta)$ is only about 5% better than the FPI-ρ. Hence, also in this case FPI-ρ is a very good choice in practice.

Fig. 8 illustrates the situation in more detail with the policy family $\mathcal{P}(\beta)$. On x-axis is the policy parameter β, and y-axis corresponds to the arrival rate λ. Left figure depicts the equivalue contours of the mean sojourn time $\mathrm{E}[T(\lambda,\beta)]$, and one can observe how β affects to the performance under different loads λ. The middle and right figures depict the relative mean sojourn time defined as

$$\frac{\mathrm{E}[T(\lambda,\beta)]}{\min_{\beta'} \mathrm{E}[T(\lambda,\beta')]},$$

which thus describes how far a given policy parameter β is from the optimal value (within this policy family) for each arrival rate λ. One can observe that with $\beta = 0.5$ (i.e., FPI-ρ) the performance remains constantly rather close to the optimal, as expected. The maximum deviation is about 5% when the arrival rate is relatively high, about $\lambda = 0.9$. However, e.g., $\beta = 0.3$ does give a slightly better performance in this case, given that one is inclined to carry out the numerical policy optimization for each case individually.

E. Dedicated streams

The systems considered so far have been traditional dispatching systems, where all tasks arrive to a central node, which then forwards each task, one at a time, to a suitable server. However, the chosen approach lends itself directly to more general scenarios. One such is illustrated in Fig. 9, where each node receives a dedicated stream of tasks in addition to those that a dispatcher sends to the queue. In this case, we can assume that the dispatcher is aware of the mean arrival rates of the node specific streams.

As the sum of Poisson processes is also a Poisson process, it is straightforward to see that such an extension to the system changes essentially nothing. The FPI approach can still be applied to an arbitrary state-independent policy, yielding a state-dependent policy that takes into account i) server rates c_i and ii) all arrival rates, λ and the λ_i. In particular, when the server specific arrivals dominate, i.e., when $\lambda \ll \min_i \lambda_i$, the FPI based policy becomes the optimal dispatching policy. It is also worth noting that in this setting, even with identical servers $c_i = c_j$, LWL$^-$ is not generally the optimal policy as it ignores the dedicated streams λ_i.

IV. Conclusions

In this paper, we have studied the processor sharing (PS) discipline, which is an important concept both in practice and in theory. Our motivation are various dispatching systems, where arriving tasks are assigned to one of the available servers. We have assumed that each server processes the given tasks in parallel, i.e., the PS discipline. In order to make a good dispatching decision, one has to take into account also the future arrivals. We have approached this problem in the MDP framework, which provides a systematic methodology to find robust dispatching policies. In particular, we have first derived an exact formula for the size-aware relative value with respect to the sojourn time for an M/D/1-PS queue. The knowledge of the relative values allows one to carry out the first policy iteration (FPI) step for an arbitrary state-independent policy.

As a result, an efficient and robust state-dependent policies can be obtained. In the example cases, the so-called FPI-ρ policy, which is also insensitive to the arrival rate, outperformed its heuristic competitors. A numerical optimization within the policy family $\mathcal{P}(\beta)$, in which also the FPI-ρ and the heuristic least-work-left (LWL) policies belong to, yielded only a small improvement, if any, over the FPI-ρ policy, as expected.

A transient analysis of a PS queue is generally a difficult task due to the numerous dependencies (e.g., each arriving tasks affects the sojourn time of the current jobs, and vice versa). However, in an M/D/1-PS queue, the tasks depart in the same order as they arrived, which greatly facilitates the analysis. Due to the nature of PS, one can expect that the obtained results serve as a reasonable approximation also for a general case of M/G/1-PS queue when the job sizes do not vary much. The exact analysis of the relative values with respect to sojourn time in an M/G/1-PS queue is left as a future work.

REFERENCES

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

[2] A. Ephremides, P. Varaiya, and J. Walrand, "A simple dynamic routing problem," *IEEE Transactions on Automatic Control*, vol. 25, no. 4, pp. 690–693, Aug. 1980.

[3] V. Gupta, M. Harchol-Balter, K. Sigman, and W. Whitt, "Analysis of join-the-shortest-queue routing for web server farms," *Performance Evaluation*, vol. 64, no. 9-12, pp. 1062–1081, Oct. 2007.

[4] L. Kleinrock, "Time-shared systems: a theoretical treatment," *J. ACM*, vol. 14, pp. 242–261, Apr. 1967.

[5] E. Altman, U. Ayesta, and B. J. Prabhu, "Load balancing in processor sharing systems," in *The Second International Workshop on Game Theory in Communication Networks (GameComm)*, Athens, Greece, Oct. 2008.

[6] K. M. Rege and B. Sengupta, "A decomposition theorem and related results for the discriminatory processor sharing queue," *Queueing Systems*, vol. 18, pp. 333–351, 1994.

[7] R. Egorova, B. Zwart, and O. Boxma, "Sojourn time tails in the M/D/1 processor sharing queue," *Probab. Eng. Inf. Sci.*, vol. 20, pp. 429–446, Jul. 2006.

[8] M. E. Crovella, M. Harchol-Balter, and C. D. Murta, "Task assignment in a distributed system: Improving performance by unbalancing load," in *Proceedings of SIGMETRICS '98*, Jun. 1998, pp. 268–269.

[9] M. Harchol-Balter, M. E. Crovella, and C. D. Murta, "On choosing a task assignment policy for a distributed server system," *Journal of Parallel and Distributed Computing*, vol. 59, pp. 204–228, 1999.

[10] 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.

[11] K. R. Krishnan, "Markov decision algorithms for dynamic routing," *IEEE Communications Magazine*, pp. 66–69, Oct. 1990.

[12] ——, "Joining the right queue: a state-dependent decision rule," *IEEE Transactions on Automatic Control*, vol. 35, no. 1, pp. 104–108, Jan. 1990.

[13] S. Aalto and J. Virtamo, "Basic packet routing problem," in *The thirteenth Nordic teletraffic seminar NTS-13*, Trondheim, Norway, Aug. 1996, pp. 85–97.

[14] E. Hyytiä, A. Penttinen, and S. Aalto, "Size- and state-aware dispatching problem with queue-specific job sizes," Dec. 2010, submitted.

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

[16] S. M. Ross, *Introduction to Probability Models*, 7th ed. Academic Press, 2000.

[17] R. Bellman, *Dynamic programming*. Princeton University Press, 1957.

[18] R. A. Howard, *Dynamic Probabilistic Systems, Volume II: Semi-Markov and Decision Processes*. Wiley Interscience, 1971.

[19] S. M. Ross, *Applied Probability Models with Optimization Applications*. Holden-Day Inc., 1970.

[20] A. Sharifnia, "Instability of the join-the-shortest-queue and FCFS policies in queuing systems and their stabilization," *Operations Research*, vol. 45, no. 2, pp. 309–314, 1997.