Optimal routing of fixed size jobs to two parallel servers

Esa Hyytiä
Aalto University, Department of Communications and Networking
PO Box 13000, FI-00076 Aalto, Finland
mail: esa.hyytia@aalto.fi

April 17, 2014

Abstract

We consider a heterogeneous two-server system processing fixed size jobs. This includes the scheduling system, where jobs wait in a common queue, and the dispatching system, where jobs are assigned to server-specific queues upon arrival. The optimal policy with respect to the delay in both systems is a threshold policy characterized by a single parameter. In this special case, the scheduling and dispatching systems achieve the same performance with the optimal threshold. The optimal threshold depends on the arrival rate and the service rates. It can be determined by means of dynamic programming, where the required value functions can be evaluated only at the necessary points by means of efficient Monte Carlo simulations. We also give the optimal threshold at three different limits, which yield a simple closed-form expression for a near-optimal threshold. The optimal policy is illustrated and compared with several heuristic dispatching policies.

Keywords: parallel servers, slow server problem, task assignment, latency, MDP, LWL, FPI, Lookahead

1 INTRODUCTION

The optimal routing or task assignment to parallel queues is a classical but still important queueing theoretic problem to which surprisingly few optimality results are known. Two basic variants exist depending on when the routing decision is made. *In dispatching problems*, each server has its own queue and the task is to assign the arriving jobs immediately upon arrival. In contrast, *in scheduling problems*, there is a common shared queue from which jobs are assigned to idle servers, i.e., the routing decision is made later when more information is often available. The typical objective is to minimize the mean delay, i.e., the latency. The dispatching problem is an interesting problem even when the servers are identical, whereas the scheduling problem is non-trivial only if the servers are non-identical (e.g., a fast and a slow server) or the cost structure includes, e.g., an energy consumption component. The routing problems arise in many contexts including manufacturing systems, data traffic routing, distributed servers and various other computer systems, where jobs or tasks are processed in parallel.

Historically, the first work on dispatching problems is by Haight (1958), who considered a system of two parallel queues under the join-the-shortest-queue (JSQ) routing. Since then, several contributions with respect to the optimal routing have been made. Most of the optimality results assume identical servers and minimal state information. For example, Winston (1977) has shown the optimality of JSQ with respect to delay (i.e., latency) when service times are exponentially distributed and only the number in queues are known. Since then the optimality of JSQ has been shown in many other similar settings, e.g., Weber (1978) considered arbitrary arrivals together with service times having a non-decreasing failure rate, while Johri (1989) considered state-dependent exponential service times, having a multi-server queues as a special case. Hordijk and Koole (1992) considered non-identical exponential servers and showed that jobs should always

be routed to a faster queue if it's shorter. Frostig and Levikson (1999) extended this structural result to service times with increasing hazard rates in the case of two servers. The most recent optimality results for JSQ are by Akgun et al. (2011). Similarly, Round-robin is optimal when the available information is the set of past decisions and the servers were initially in the same state (Ephremides et al., 1980; Liu and Towsley, 1994; Liu and Righter, 1998). The so-called Size-Interval-Task-Assignment SITA (Crovella et al., 1998; Harchol-Balter et al., 1999) is a static policy (actions are independent of the past actions and the state of the queues), where the job size distribution is divided by $k+1$ thresholds, $0 = \xi_0 < \xi_1 < \ldots < \xi_{k-1} < \xi_k$, and Server i receives those jobs which size is in the i^{th} interval $[\xi_{i-1}, \xi_i)$. The idea is to reduce the variance of the job sizes in each server. Feng and Misra (2003) have shown that SITA is the optimal size-aware static policy for the first-come-first-served (FCFS) servers.

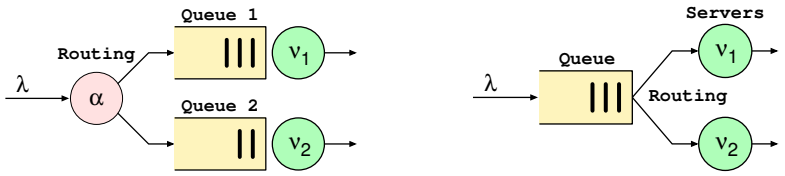
The most related scheduling problem is the so-called *slow-server-problem*, where the system comprises two (or more) servers with non-identical service rates and the jobs wait in a common queue. The servers are non-preemptive, i.e., once a job is assigned and the service has begun, the job cannot be re-assigned to another server. The service times are often assumed to be exponentially distributed with rates μ_i for Server i, and the state information is the number of jobs in the system. The problem is to decide when to assign jobs also to the slower server. The slow server problem was first studied by Larsen (1981), who conjectured that the optimal policy is of threshold type: the slower server is activated only when the number of jobs in the system is greater than n^*. Agrawala et al. (1984) proved this for a set of jobs with exponentially distributed sizes (i.e., without arrivals). Lin and Kumar (1984) and Walrand (1984) were the first to prove this with Poisson arrivals. Later, Koole (1995) also gave a simple iterative proof for the same result. The slow server problem is also a special case of the model considered by Hajek (1984). See also (Aalto and Virtamo, 1996) and (Akgun et al., 2014), where the latter includes also the energy consumption in the problem formulation. The general slow server problem with $k > 2$ servers is hard (Véricourt and Zhou, 2006). The non-exponential service times have been considered by Viniotis and Ephremides (1988), and Righter and Xu (1991).

In this paper, we consider a *heterogeneous* system of two parallel servers processing fixed size jobs. We let ν_i denote the service rate of Server i, and without loss of generality, assume that job sizes are 1 and $\nu_1 \geq \nu_2$, so that the service times are $\Delta_1 = 1/\nu_1$ and $\Delta_2 = 1/\nu_2$. With the least-work-left (LWL) dispatching policy, this system is equivalent to an M/D/2 queue with a shared buffer. Moreover, if the servers are identical, $\nu_1 = \nu_2$, then LWL is the optimal dispatching policy (Harchol-Balter et al., 1999; Hyytiä et al., 2011). However, the routing problems are non-trivial with heterogeneous service rates. The aim of this paper is three-fold: first, to show how the optimal dispatching and scheduling policies look in the heterogeneous case, second, to show how they can be determined efficiently by applying the optimality criterion of the Markov decision processes (MDP), and third, to compare several heuristic dispatching policies to the optimal one.

The rest of the paper is organized as follows. In Section 2 we discuss the relationship between the dispatching and scheduling problems and show that in our case the optimal scheduling policy can be realized with a dispatching policy. In Section 3, we derive three explicit expressions for the optimal scheduling/dispatching policy at different limits, and show how the optimal policy can be computed numerically for an arbitrary system. Section 4 gives a closed-form expression for a near-optimal policy, and compares different dispatching policies. Section 5 concludes the paper.

2 DISPATCHING AND SCHEDULING PROBLEMS

First we give some necessary definitions and observations. In the number- or class-aware setting one knows the size distribution of each job, but not their actual size (cf. a voice or video call). In this case, two jobs are identical if they are from the same distribution. However, their actual service time may be different. In contrast, in the size-aware setting one has more information as the exact size of a job becomes known upon arrival (cf. a file transfer). In this case, two jobs are identical if their size happens to be the same. It follows that with identical jobs and non-preemptive servers, the scheduling order in a queue is irrelevant. This includes the basic slow server model with exponentially distributed service times, as well as, our model with a fixed job size.



(a) Dispatching system (b) Scheduling system

Figure 1: Left: A dispatching system of two parallel queues processing identical fixed size jobs. The optimal size-aware routing policy α_{opt} takes into account the backlogs u_i and the service rates ν_i when assigning the arriving jobs to the servers. Right: The equivalent scheduling system with a shared queue, where the task is to decide when the slower server is utilized.

2.1 Threshold policy

Consider next a scheduling system with a common queue and two servers with service rates $\nu_1 \geq \nu_2$.

Definition 1 (Threshold Scheduling Policy) *Upon an arrival or departure event, a (work-conserving) threshold scheduling policy routes a job from the common queue to the faster Server 1 if it is idle, and if not, then to the slower Server 2 if it is idle and the amount of unfinished work in the system is χ or higher.*

In the number-aware system (cf. the basic slow server problem with exponential service times), the unfinished work is measured in terms of the number of jobs in the queue denoted by n, whereas in a size-aware case we can consider the actual backlog and remaining service times. Suppose Server 2 is idle and let r_1 denote the remaining service time in Server 1, then the *system's virtual backlog* (measured in time) can be defined as

$$u^* = n/\nu_1 + r_1,$$

where $0 < r_1 < 1/\nu_1$ and a tentative assumption is that the n jobs waiting in the queue would be served in the faster Server 1. In a size-aware setting, the threshold policy assigns a job to Server 2 iff $u^* \geq \chi$.

As already mentioned, the threshold policy has been shown to be the optimal for the basic slow server problem with exponential service times. As expected, the same holds also with fixed size jobs.

Lemma 1 (Optimal Scheduling) *The optimal non-preemptive scheduling minimizing the mean delay for Poisson arrival process of fixed size jobs in a system of two heterogeneous servers with service rates $\nu_1 \geq \nu_2$ is a threshold policy, where a job is routed to the slower Server 2 only when $u^* = n/\nu_1 + r_1 \geq \chi$ for some χ.*

Proof: Instead of giving a rigorous proof, we settle with arguing why a non-threshold policy cannot be optimal (cf. Walrand (1984)). First, all jobs are identical and therefore it is clear that the faster server is utilized whenever possible. Hence, the question is when a job should be assigned also to the slower server. The number of jobs in the queue, n, together with the remaining service times r_1 and r_2 clearly describe the system's state fully. When Server 2 is idle ($r_2 = 0$) and there are jobs waiting in the queue ($n > 0$ and also $r_1 > 0$), the decision is whether to assign a job also to Server 2 or not. In these states, the virtual backlog $u^* = n/\nu_1 + r_1$ describes the system's state fully. With a threshold policy, a job is assigned to Server 2 only if $u^* \geq \chi$. Now if the optimal policy is not a threshold policy, this means that there are states u^* and $u^* + \Delta$ with $\Delta > 0$ such that in u^* a job is assigned to Server 2, but in $u^* + \Delta$ it is not. This means that in state $u^* + \Delta$, the optimal policy would wait for the virtual backlog either to increase or to decrease enough, before routing a job to Server 2. In either case, it has already been decided that at least one job will be routed to Server 2 (before the busy period ends). This clearly makes no sense as all jobs are identical, and a better performance is obtained if a job is routed to Server 2 immediately. ■

Note that Lemma 1 holds also for an arbitrary arrival process with i.i.d. inter-arrival times. The full state information at the time of an arrival is the virtual backlog u^* also in this case, and due to the fixed job sizes, a decision to schedule a one more job to the slower server can be made upon the arrival.

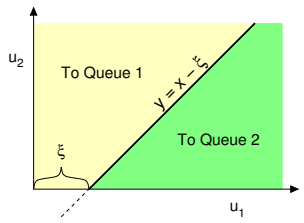


Figure 2: A dispatching policy for two queues can be defined by a switching curve. For threshold based policies, the switching curve is a straight line with slope 1. The threshold ξ defines the point where the line crosses the x-axis, i.e., $u_1 - u_2 = \xi$.

2.2 Equivalence between dispatching and scheduling

Both in dispatching and scheduling systems, the decisions govern the routing of jobs to the servers. The difference is that in a dispatching system this decision must be done upon arrival, whereas in a scheduling system jobs can wait in a common queue from which they are then routed to servers, as illustrated in Fig. 1. Any dispatching policy can be *imitated* in a scheduling system simply by "marking" the decisions upon arrival. Therefore, a scheduling system with optimal decisions is never worse than an equivalent dispatching system. In contrast, it is easy to see that many scheduling policies cannot be realized in an equivalent dispatching system. An interesting question is when the early dispatching decisions do not degrade the performance of the system when compared to a scheduling system (Aalto and Virtamo, 1996).

We note that a standard M/G/k-FCFS queue utilizing all the servers unconditionally (i.e., $\chi_i = 0\ \forall i$) is equivalent with a size-aware dispatching system with LWL (Harchol-Balter et al., 1999), where a queue with the shortest backlog is chosen. Consequently, when a threshold policy with $\chi_i = 0\ \forall i$ is optimal, the corresponding dispatching system with LWL achieves the same performance and there is no performance penalty due to the earlier dispatching decisions. The dispatching policy LWL is a special case of the threshold based dispatching policies:

Definition 2 (Threshold Dispatching Policy) *A threshold based dispatching policy α chooses the queue with the shortest shifted backlog,*

$$\alpha = \underset{i}{\operatorname{argmin}}\ (u_i + \xi_i),$$

where the ξ_i are server-specific constants.

Thus, LWL is obtained with $\xi_i = 0\ \forall\, i$. In case of two servers, a single constant is sufficient and we can write

$$\alpha = \operatorname{argmin}\ \{u_1 - \xi,\, u_2\}.$$

This policy corresponds to a straight line in (u_1, u_2)-plane with slope 1 that crosses the x-axis at $(\xi, 0)$, as illustrated in Fig. 2. (throughout this paper we assume that $\nu_1 \geq \nu_2$, so that sensible $\xi \geq 0$). Alternatively, the threshold dispatching policy for two servers can be stated as follows:

Definition 3 *Threshold based dispatching policy routes a new job to the slower Server 2 iff $u_1 - u_2 > \xi$.*

In our case, the size of the jobs is fixed and we have the following result:

Lemma 2 *For fixed size jobs and two servers with service rates $\nu_1 \geq \nu_2$, any non-preemptive scheduling policy defined by threshold χ can be realized as a dispatching system with threshold $\xi = \chi - 1/\nu_1$.*

This result follows from the fact that in this special case it is possible to fix a queue for each new job upon arrival in a way that matches the scheduling order. Consequently, *dispatching jobs upon arrival does not degrade the performance*. The offset difference in the thresholds is due to the fact that in the scheduling system the decision to route a job to a slower server is done after a new job has already entered a queue, whereas in the dispatching system one considers the backlogs before the new job is included.

3 OPTIMAL THRESHOLDS

In the previous section, we assumed a rather general arrival process and showed that the optimal policy for scheduling and dispatching systems with two servers and fixed size jobs takes the form of a threshold policy.

The remaining task is to determine this threshold. To this end, we assume a Poisson arrival process with rate λ (even though some of the following results hold more generally). We first show how the optimal threshold can be determined efficiently for an arbitrary system by means of Monte Carlo simulations, and then derive the optimal threshold at three different limits Let γ denote the *asymmetry in the service rates*,

$$\gamma = \nu_2/\nu_1,$$

where $\nu_1 \geq \nu_2$ so that $0 < \gamma < 1$.

3.1 Computation of the optimal policy

As discussed, the optimal routing policy takes a specific form that can be described by a single threshold value, which we need to determine next. As the corresponding scheduling and dispatching systems behave equivalently, we can choose to consider a dispatching system with the server-specific FCFS queues, as then the state of the system can be defined by a pair (u_1, u_2), where u_i denotes the backlog in Queue i.

When determining the optimal threshold, we need to evaluate the so-called value functions. Formally, a value function is the mean difference in the infinite-horizon costs between a system initially in state (u_1, u_2) and a system initially in equilibrium,

$$v(u_1, u_2) \triangleq \lim_{t\to\infty} \mathrm{E}[V_{u_1,u_2}(t) - r \cdot t], \tag{1}$$

where $V_{u_1,u_2}(t)$ denotes the costs incurred during the time interval $(0, t)$ when initially in state (u_1, u_2), and r is the mean cost rate (with the current policy) (Puterman, 2005). In our case, the objective is to minimize the mean delay and we can define that each job incurs an immediate cost equal to its sojourn time when it enters a queue (with FCFS, the sojourn time gets fixed upon arrival). Moreover, the mean cost rate is $r = \lambda \mathrm{E}[T]$, where $\mathrm{E}[T]$ denotes the mean sojourn time. Given a value function, one can carry out the policy improvement step that yields a better policy (unless the current policy is already the optimal). Repeating the policy improvement steps yields eventually the optimal policy.[1]

Due to analogy with the scheduling system, the threshold policy is known to be the optimal. Hence, the optimal dispatching policy is defined by a *switching curve* that is a straight line with slope 1 in the (u_1, u_2)-space. Let Δ_i denote the service time of a job at Server i, $\Delta_i = 1/\nu_i$. The optimal routing policy can be characterized by point $(\xi, 0)$, where the threshold line crosses the u_1-axis. In passing, we note that it is straightforward to argue that the optimal (threshold) policy assigns a job to Queue 2 at states $(x, 0)$ with $x \geq \Delta_2$. Thus, with ν_2 scaled to 1, the optimal curve crosses the x-axis always somewhere in $[0, 1]$. At any point on the switching curve, including $(\xi, 0)$, it is irrelevant which queue is chosen. We can compare the cost of assigning a job to Queue 1 and Queue 2 at state $(\xi, 0)$ with aid of the value function,

$$\begin{aligned} a_1 &= \xi + \Delta_1 + v(\xi + \Delta_1, 0) - v(\xi, 0), \\ a_2 &= \quad \Delta_2 + v(\xi, \Delta_2) \quad\quad - v(\xi, 0), \end{aligned} \tag{2}$$

which gives for the difference $a_1 - a_2$,

$$c(\xi) \triangleq \xi + \Delta_1 - \Delta_2 + v(\xi + \Delta_1, 0) - v(\xi, \Delta_2) \tag{3}$$

where the first two terms correspond to the delays of the current job with the two alternative actions, and the last two terms to the additional delay the future arrivals experience on average. According to the dynamic programming, if $c(\xi) > 0$, then the correct ξ is smaller, and vice versa. Thus, we need to evaluate the value function only at $(\xi + \Delta_1, 0)$ and (ξ, Δ_2), and with the optimal $\xi = \xi_{\mathrm{opt}}$, i.e., with the optimal policy, $c(\xi) = 0$.

[1] Strictly speaking, this is true for systems with a finite state-space, whereas in our case the state-space is continuous and infinite. Nonetheless, we apply the methodology and assume that the iteration converges to the globally optimal policy.

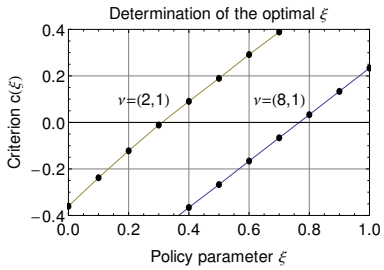


Figure 3: The optimal threshold ξ_{opt} corresponds to the root of (3).

Unfortunately, as already mentioned, the value function or its difference $v(A) - v(B)$ for two different states A and B is not readily available for an arbitrary threshold policy. However, given the current routing policy is "sensible", it is possible to estimate the difference $v(A) - v(B)$ efficiently by means of Monte Carlo simulations similarly as in (Virtamo and Aalto, 1996) and (Hyytiä and Virtamo, 2000). Due to the complex settings, the value functions were estimated in a somewhat rudimentary manner by simulating the system from two different initial states A and B for a fixed duration of t, and then comparing the costs that the two systems incurred during the fixed time t,

$$v(A) - v(B) \approx \mathrm{E}[V_A(t) - V_B(t)],$$

where $V_z(t)$ denotes the costs incurred during time $(0, t)$ when initially in state z. A large number of repeated simulation runs were used to generate samples of $V_A(t) - V_B(t)$ based on which the difference $v(A) - v(B)$ was then estimated. However, according to (1), the above relation is exact only when $t \to \infty$.

In our case, the stochastic system is far simpler than, e.g., in (Hyytiä and Virtamo, 2000), and we can utilize the Markov property to generate unbiased samples of $v(A) - v(B)$ directly. More specifically, we can simulate two systems, one initially in state A and the other in state B, *in parallel with the same arrival sequence until they reach the same state*. After that, they will behave identically and incur the same costs. That is, we have an exact relation

$$v(A) - v(B) = \mathrm{E}[V_A(T_c) - V_B(T_c)], \tag{4}$$

where T_c denotes the time duration until the two systems converge with the same arrival sequence.[2] As we are interested in stable systems, the two systems are eventually both empty at the same time (a renewal point), which serves as an upper bound for T_c. However, due to the fixed job size, all jobs are identical and the same state is typically reached much earlier. Consequently, it is computationally feasible to evaluate a system also under a heavy load.

Specifically, we propose that the optimal threshold ξ_{opt} is determined by efficient Monte Carlo simulations as follows. First, we simulate the system from two different initial states, $(\xi+\Delta_1, 0)$ and (ξ, Δ_2), with identical arrival patterns until the two systems reach the same state. The difference in the incurred costs until that moment gives an independent sample H_i for $v(\xi + \Delta_1, 0) - v(\xi, \Delta_2)$. Carrying out n simulation runs each with a different initial random seed, gives n samples, $H_1, \ldots, H_n$, and their mean $\bar{H} = (H_1 + \ldots + H_n)/n$ yields an accurate estimate for $v(\xi + \Delta_1, 0) - v(\xi, \Delta_2)$. Consequently, $c(\xi)$ can be evaluated by substituting $\bar{H}$ into (3), $c(\xi) \approx \xi + \Delta_1 - \Delta_2 + \bar{H}$. If $c(\xi) < 0$, we increase ξ, and vice versa, until the root of $c(\xi)$ has been found with an appropriate accuracy.

Fig. 3 illustrates how $c(\xi)$, increases smoothly as ξ increases from 0 to 1. Here we have scaled $\nu_2 = 1$ so that $\xi_{\text{opt}} < 1$. The optimal threshold curve corresponds to the (single) root of $c(\xi)$. Numerically, $\xi \approx 0.312$ for $\boldsymbol{\nu} = (2, 1)$, and $\xi \approx 0.767$ for $\boldsymbol{\nu} = (8, 1)$.

[2]We note that this general technique to compare a stochastic system with two different initial states is known as the *coupling method*, and it was first proposed by Döblin already in 1938 (Lindvall, 1992).

3.2 Limiting cases

Next we give the optimal threshold for the dispatching system at three different limits. The corresponding scheduling threshold $\chi_{\rm opt}$ follows then from Lemma 2. The first result is obtained when the service rates ν_1 and ν_2 become equal:

Lemma 3 *In a homogeneous system with $\gamma = \nu_2/\nu_1 = 1$, the optimal threshold is $\xi_{\rm opt} = 0$ by symmetry.*

The proof is self-evident and omitted. Consider next the light-load case when the offered load

$$\rho = \lambda/(\nu_1 + \nu_2),$$

tends to zero (i.e., $\lambda \to 0$).

Lemma 4 *When the offered load tends to zero, $\rho \to 0$, the optimal threshold is $\xi_{\rm opt} = \nu_2^{-1} - \nu_1^{-1}$.*

Proof: In this case, one can ignore the future arrivals and the Myopic policy, serving a job at the server with the shortest sojourn time, becomes optimal. Hence, a job is routed to Server 2 only when $u_1 + 1/\nu_1 > u_2 + 1/\nu_2$, which gives $\xi_{\rm opt} = (\nu_1 - \nu_2)/(\nu_1\nu_2) = \nu_2^{-1} - \nu_1^{-1}$. ■

Our final result gives the optimal policy at the limit when the two servers become highly asymmetric, $\nu_1 \gg \nu_2$. We fix ρ and ν_2, and then let $\nu_1, \lambda \to \infty$ in such a way that ρ remains at its fixed value.

Lemma 5 *When the service rates become highly asymmetric, $\gamma \to 0$, the optimal threshold is $\xi_{\rm opt} = \dfrac{1-\rho}{\nu_2}$.*

Proof: First we fix $\rho < 1$ and ν_2, i.e., the arrival rate is $\lambda = (\nu_1 + \nu_2)\rho$. Then we let $\nu_1 \to \infty$ so that the asymmetry $\gamma \to \infty$. At this limit, $\lambda/\nu_1 \to \rho < 1$, which means that eventually, for all $\rho < 1$, Server 1 can handle the offered load alone as $\nu_1 \to \infty$. Consequently, the slower Server 2 is utilized only occasionally as only a negligible fraction of the jobs is routed there. In other words, routing a job to the slower server is generally an isolated event and most of the time the slower server is idle.

Consider next the admission costs a_1 and a_2 given in (2) at state $(\xi, 0)$. The admission cost a_1 can be estimated by considering a modified system where the arriving job j is processed only at the end of the current busy period in Server 1. This way admitting job j does not harm any job (until the busy period ends). The jobs are identical and any work-conserving non-preemptive service order is equivalent. Therefore, the sojourn time of job j in the modified system includes also the additional sojourn time that later arriving jobs would experience under FCFS due to the admission of job j (until it enters the service). Moreover, the final states (Δ_1, u_2^*) (with job j) and $(0, u_2^*)$ (without it), for some u_2^*, become equivalent when $\nu_1 \to \infty$ and $\Delta_1 \to 0$. Thus the mean sojourn time of job j in the modified system approaches a_1 as $\nu_1 \to \infty$. At this limit, the remaining busy period in Server 1 is similar as in M/G/1, $u/(1-\rho)$ (Kleinrock, 1975), which gives

$$a_1 \approx \frac{\xi}{1-\rho}.$$

If job j is instead assigned to Server 2, then its sojourn time is $\Delta_2 = 1/\nu_2$. When $\nu_1 \to \infty$, Server 1 receives almost all jobs, and $v(\xi, \Delta_2) \to v(\xi, 0)$. Therefore

$$a_2 \approx \Delta_2.$$

These two are equal when $\xi = \xi_{\rm opt} = (1-\rho)\Delta_2$. ■

By scaling, we can further assume that $\nu_2 = 1$ so that $\gamma = 1/\nu_1$, and the above three results become:

(i) $\xi_{\rm opt} = 0$, when $\nu_1 \to \nu_2$,
(ii) $\xi_{\rm opt} = 1 - \gamma$, when $\rho \to 0$,
(iii) $\xi_{\rm opt} = 1 - \rho$, when $\gamma \to 0$ and $\rho < 0$ is fixed.

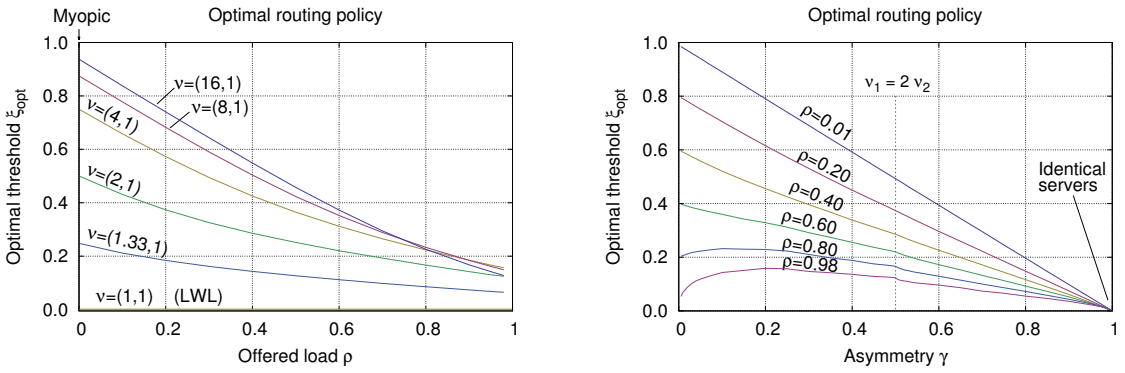


Figure 4: The optimal value for the threshold $\xi_{\rm opt}$ as a function of the offered load $\rho = \lambda/(\nu_1 + \nu_2)$ (left) and the asymmetry $\gamma = \nu_2/\nu_1$ (right).

4 NUMERICAL EXAMPLES

In this section, we consider the dispatching system with Poisson arrival process with rate λ and fix $\nu_2 = 1$. First we illustrate how the optimal threshold ξ depends on the system parameters (the offered load and the asymmetry in the service rates), and then we study the performance gain available from choosing the threshold appropriately. For comparison, we consider also the following heuristic dispatching policies that belong to the same family of threshold policies:

i) *Least-work-left* (LWL), as already mentioned, is obtained with $\xi_{\rm LWL} = 0$.

ii) *Myopic* dispatching policy chooses the queue which is the fastest for the new job. This corresponds to a threshold policy with $\xi_{\rm Myopic} = (\nu_1 - \nu_2)/(\nu_1\nu_2) = 1 - \gamma$.

iii) *Triple* (TRI) dispatching policy with $\xi_{\rm TRI} = (\nu_2^{-1} - \nu_1^{-1})(1-\rho) = (1-\gamma)(1-\rho)$ satisfies Lemmas 3–5 at the corresponding three limits, i.e, when $\gamma \to 1$, $\gamma \to 0$ and $\rho \to 0$.

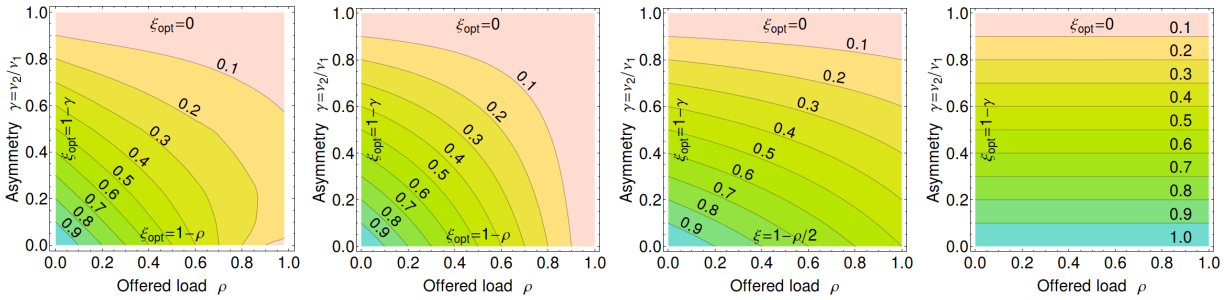
iv) *First policy iteration* (FPI) when applied to a random Bernoulli split (see Appendix for details) gives also a threshold dispatching policy with $\xi_{\rm FPI} = (\nu_2^{-1} - \nu_1^{-1})(1-\rho/2) = (1-\gamma)(1-\rho/2)$.

Thus, LWL and Myopic are insensitive to the arrival rate λ and the offered load ρ. Note also that the threshold of TRI can be seen as a synthesis of the form FPI gives and the exact results at the three limits.

4.1 Optimal thresholds

Let us next illustrate how the optimal threshold varies as a function of the two system parameters, the offered load ρ and the asymmetry γ. The exact results at the three limits were derived in Section 3.2, and at other points we can numerically find the optimal threshold $\xi_{\rm opt}$ using the Monte Carlo method as explained in Section 3.1.

Fig. 4 (left) depicts $\xi_{\rm opt}$ for $\boldsymbol{\nu} = (\nu_1, \nu_2)$, where $\nu_1 = 1$, 1.33, 2, 4, 8 and 16 and $\nu_2 = 1$. We observe that in each case the threshold decreases as the offered load increases, i.e., the slower server should be taken into use earlier when the load is higher, as expected. The left side at $\rho = 0$ corresponds to the Myopic policy (see Lemma 4). Similarly, the lowest curve with $\boldsymbol{\nu} = (1, 1)$ corresponds to LWL (see Lemma 3). Consequently, it becomes obvious that *neither LWL or Myopic is optimal* in the heterogeneous case with $\nu_1 \neq \nu_2$ and $\rho > 0$. In Fig. 4 (right), the x-axis corresponds to the asymmetry $\gamma = \nu_2/\nu_1$ (with $\nu_2 = 1$) and the offered load ρ is the curve parameter. At very high loads, e.g., at $\rho = 0.98$, we can notice a small notch at $\gamma = 0.5$. At this point, Server 1 becomes two times faster than Server 2.



(a) Optimal threshold (b) Triple (c) FPI based threshold (d) Threshold with Myopic

Figure 5: Threshold based dispatching policies illustrated (from left to right): optimal policy, Triple based on the three limiting cases, FPI based on RND, and Myopic policy (neglecting the later arrivals). LWL is omitted as it has the constant threshold $\xi = 0$ for all values of ρ and γ.

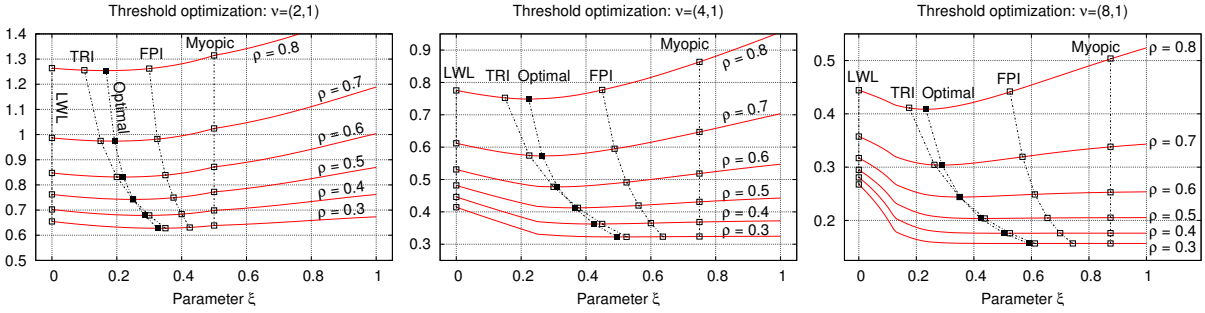


Figure 6: Mean delay as a function of the policy parameter ξ for $\boldsymbol{\nu} = (2, 1)$, $\boldsymbol{\nu} = (4, 1)$, and $\boldsymbol{\nu} = (8, 1)$ (from left to right). Triple (TRI) achieves almost the performance as the optimal policy in each case.

Fig. 5(a) illustrates the optimal threshold for all feasible values of (ρ, γ): $0 \leq \rho < 1$ and $0 < \gamma \leq 1$. When both ρ and γ are small, the equi-value contour lines are almost straight lines with slope -1. As ρ and γ increase, the lines turn to a smoothly bending curves. Figs. 5(b)-(d) illustrate the heuristic thresholds according to TRI, FPI and Myopic (LWL is omitted as ξ_{LWL} is a constant). We note that the shape of ξ_{TRI} is rather close to the optimal threshold ξ_{opt} for all values of (ρ, γ), and exactly optimal at the three limits: when $\gamma \to 0$, $\gamma \to 1$ and $\rho \to 0$. In contrast, both FPI[3] and Myopic are optimal at two limits, when $\rho \to 0$ and when $\gamma \to 1$, whereas LWL is optimal only when $\gamma \to 1$.

4.2 Performance gain

No closed-form expression is known for the delay in an M/D/2 queue (Franx, 2001), i.e., for LWL with $\boldsymbol{\nu} = (1, 1)$. In other words, finding an expression for the mean delay or waiting time for a general threshold policy is not trivial. Therefore, we resort to process simulation in order to evaluate the gain from the optimal choice of ξ. Fig. 6 (left) depicts the mean delay as a function of ξ for $\boldsymbol{\nu} = (2, 1)$, i.e., the asymmetry parameter $\gamma = 0.5$. The lowest mean delay, indicated with the black dots, is obtained at the expected points at different levels of the offered load ρ. The heuristic thresholds, LWL, TRI, FPI and Myopic, are indicated with empty squares. We can observe that the mean delay with TRI is close to optimal in each case. Also FPI does a fairly good job in overall. The performance of LWL improves as ρ increases, whereas with Myopic the situation is the opposite. Fig. 6 (middle) and (right) depict the results for the more asymmetric settings with $\boldsymbol{\nu} = (4, 1)$ and $\boldsymbol{\nu} = (8, 1)$. In these cases, the objective function is very flat about the optimal point

[3]The policy given by FPI depends on the basic policy, and some other choice, e.g., RND with optimal split, may yield better results. Here we have, however, chosen RND with load balancing for the sake of compact expressions.

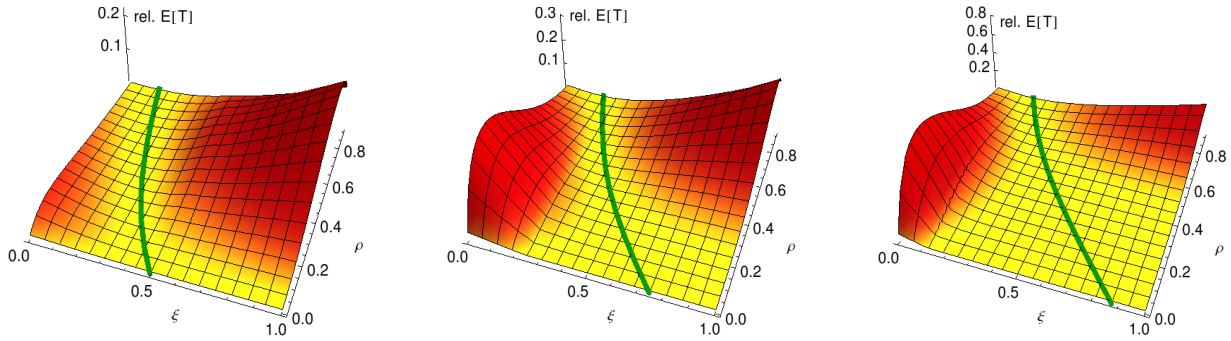


Figure 7: Mean delay relative to the optimal as a function of the policy parameter ξ for $\boldsymbol{\nu} = (\nu_1, 1)$ with $\nu_1 = 2, 4, 8$ (from left to right). The optimal threshold is indicated with the path at the bottom of the "valley".

when the offered load is low, and all policies except LWL are almost equally good. In fact, even a simple policy that assigns all jobs unconditionally to the faster server has already a reasonably good performance. As the load increases, choosing the threshold appropriately becomes more important. Again, TRI yields almost optimal performance in each case, and the deficiencies with LWL and Myopic get magnified as the asymmetry increases.

In Fig. 7, we have depicted the relative delay, $\mathrm{E}[T_\xi]/\mathrm{E}[T_{\xi_{\mathrm{opt}}}] - 1$, i.e., the z-axis corresponds to the penalty for using a suboptimal threshold ξ instead of the optimal ξ_{opt}. The optimal threshold ξ_{opt} is indicated with the path at the bottom of the "valley". With a moderate asymmetry at $\gamma = 2$, the optimal path is self-evident. In contrast, when asymmetry increases and the offered load is small, the bottom of the valley is very flat and any ξ sufficiently close to ξ_{opt} yields practically the same delay. As mentioned, in these points the slower server is basically useless.

In summary, it seems that when the offered load is low it is important to have a sufficiently high ξ, whereas under a heavy load, a too high ξ can deteriorate the performance significantly. Moreover, setting the threshold according to TRI yields almost the optimal delay in the example cases. As TRI is also the optimal policy at the three limits, we can conclude that it is an excellent heuristic policy for the considered system.

4.3 Evaluation of the Lookahead policy

Instead of minimizing only the mean delay, a more general cost structure can include also energy, fairness, priorities etc. In other words, there is a clear need for a systematic procedure to derive robust scenario-specific cost- and state-aware dispatching policies. The already mentioned FPI is one such procedure, where the cost of each decision is estimated by assuming that the consecutive decisions are according to a basic policy. Typically the basic policy is static (e.g. RND), as then the system decomposes and the corresponding value function is straightforward to compute (see, e.g., (Aalto and Virtamo, 1996; Krishnan, 1990; Bhulai, 2006; Hyytiä et al., 2012b)). The downside is that the static basic policy effectively separates the servers immediately after the decision and dependencies between the queues are not present in the cost estimates. The recently proposed *Lookahead* approach, introduced in (Hyytiä, 2013), builds on FPI. Instead of focusing only on the current job like FPI, it considers also the job(s) arriving next explicitly before letting a static basic policy to take over. The routing decision for the next job is tentative, but it enables a better grasp of the (expected) inter-play between the servers in the near-future. The numerical results are promising, suggesting that Lookahead makes near-optimal decisions.

Fig. 8 (left) illustrates the switching curves for LWL, Myopic, TRI, FPI, Lookahead (with a static second action) and the optimal dispatching policy for $\rho = 0.8$ and $\boldsymbol{\nu} = (4, 1)$. For the threshold policies, the switching curves are straight lines with slope 1. The switching curve of Lookahead is slightly non-linear, but

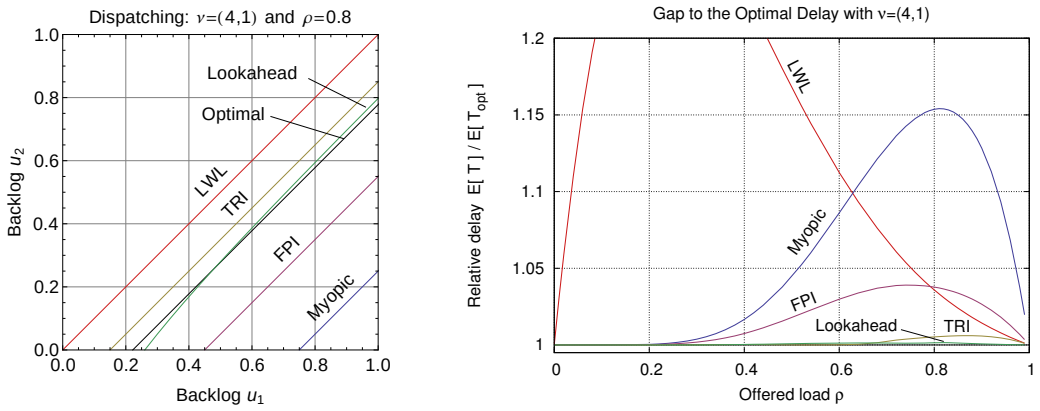


Figure 8: Right: Different threshold based dispatching policies illustrated for $\boldsymbol{\nu} = (4, 1)$ when $\rho = 0.8$. With Lookahead, the switching curve is nearly linear and very close to the optimal with $\xi_{\text{opt}} \approx 0.222$. Left: Gap to the optimal performance for $\boldsymbol{\nu} = (4, 1)$ as a function of offered load ρ.

deviates only a little from the optimal switching curve. In the end, the gap between the optimal policy and, e.g., FPI or Lookahead based policies remains unknown until the optimal decisions have been determined. As we can now compute the optimal policy for this specific system, we can also quantify the gap in this case. Fig. 8 (right) depicts the gap to the optimal delay with the different policies as a function of ρ. As expected, FPI provides a robust policy that works fairly well for any ρ. The mean delay with TRI is very close to the optimal except when ρ is high, where an almost negligible increase in the delay appears. Lookahead, on the other hand, achieves a practically optimal delay with all values of ρ. Thus, one can expect a strong performance from it also in more complicated scenarios with diverse cost structures.

5 CONCLUSIONS

There are very few optimality results for dispatching systems with heterogeneous servers, which was the main motivation behind this work. In this paper, we considered a two-server system with jobs of a fixed size. First we showed the equivalence between a threshold based scheduling policy (cf. the slow server problem) and a dispatching policy characterized by a straight threshold line with slope 1. The fact that the threshold scheduling policy is optimal means that the optimal dispatching policy is a threshold policy for the difference in the backlogs, $u_1 - u_2$. The optimal dispatching (scheduling) policy is characterized by the threshold ξ_{opt}, the value of which can be determined by finding ξ that satisfies the optimality criterion (3). This requires the evaluation of the value function, for which we proposed an efficient Monte Carlo simulation based approach. Moreover, we gave the optimal threshold ξ_{opt} at three different limits, which led to a simple near-optimal expression for the threshold. The future work includes considerations of other cost structures and systems with $k > 2$ servers.

ACKNOWLEDGEMENTS

This work was supported by the Academy of Finland in the Top-Energy project (grant no. 268992). The author would like to thank Prof. R. Righter and the anonymous reviewers for their valuable comments that greatly improved the manuscript.

References

Aalto, S. and Virtamo, J. (1996). Basic packet routing problem. In *The thirteenth Nordic teletraffic seminar NTS-13*, pages 85–97, Trondheim, Norway.

Agrawala, A., E.G., J. C., Garey, M., and Tripathi, S. (1984). A stochastic optimization algorithm minimizing expected flow times on uniform processors. *IEEE Transactions on Computers*, 33(4):351–356.

Akgun, O., Righter, R., and Wolff, R. (2011). Multiple server system with flexible arrivals. *Advances in Applied Probability*, 43:985–1004.

Akgun, O. T., Down, D. G., and Righter, R. (2014). Energy-aware scheduling on heterogeneous processors. *IEEE Transactions on Automatic Control*, 59(3).

Bhulai, S. (2006). On the value function of the M/Cox(r)/1 queue. *Journal of Applied Probability*, 43(2):363–376.

Crovella, M. E., Harchol-Balter, M., and Murta, C. D. (1998). Task assignment in a distributed system: Improving performance by unbalancing load. In *Proceedings of SIGMETRICS '98*, pages 268–269, Madison, Wisconsin, USA.

Ephremides, A., Varaiya, P., and Walrand, J. (1980). A simple dynamic routing problem. *IEEE Transactions on Automatic Control*, 25(4):690–693.

Feng, H. and Misra, V. (2003). Mixed scheduling disciplines for network flows. *SIGMETRICS Perform. Eval. Rev.*, 31:36–39.

Franx, G. J. (2001). A simple solution for the M/D/c waiting time distribution. *Oper. Res. Lett.*, 29(5):221–229.

Frostig, E. and Levikson, B. (1999). Optimal routing of customers to two parallel heterogeneous servers: The case of IHR service times. *Operations Research*, 47(3):438–444.

Haight, F. A. (1958). Two queues in parallel. *Biometrika*, 45(3-4):401–410.

Hajek, B. (1984). Optimal control of two interacting service stations. *IEEE Transactions on Automatic Control*, 29(6):491–499.

Harchol-Balter, M., Crovella, M. E., and Murta, C. D. (1999). On choosing a task assignment policy for a distributed server system. *Journal of Parallel and Distributed Computing*, 59:204–228.

Hordijk, A. and Koole, G. (1992). On the assignment of customers to parallel queues. *Probability in the Engineering and Informational Sciences*, 6:495–511.

Hyytiä, E. (2013). Lookahead actions in dispatching to parallel queues. *Performance Evaluation*, 70(10):859–872. (IFIP Performance'13).

Hyytiä, E., Aalto, S., and Penttinen, A. (2012a). Minimizing slowdown in heterogeneous size-aware dispatching systems. *ACM SIGMETRICS Performance Evaluation Review*, 40:29–40.

Hyytiä, E., Penttinen, A., and Aalto, S. (2012b). Size- and state-aware dispatching problem with queue-specific job sizes. *European Journal of Operational Research*, 217(2):357–370.

Hyytiä, E., Penttinen, A., Aalto, S., and Virtamo, J. (2011). Dispatching problem with fixed size jobs and processor sharing discipline. In *23rd International Teletraffic Congress (ITC'23)*, pages 190–197, San Fransisco, USA.

Hyytiä, E. and Virtamo, J. (2000). Dynamic Routing and Wavelength Assignment Using First Policy Iteration. In *the Fifth IEEE Symposium on Computers and Communications, ISCC'2000*, pages 146–151, Antibes, Juan les Pins, France.

Johri, P. K. (1989). Optimality of the shortest line discipline with state-dependent service rates. *European Journal of Operational Research*, 41(2):157–161.

Kleinrock, L. (1975). *Queueing Systems, Volume I: Theory*. Wiley Interscience.

Koole, G. (1995). A simple proof of the optimality of a threshold policy in a two-server queueing system. *Systems and Control Letters*, 26(5):301–303.

Krishnan, K. R. (1990). Joining the right queue: a state-dependent decision rule. *IEEE Transactions on Automatic Control*, 35(1):104–108.

Krishnan, K. R. and Ott, T. J. (1986). State-dependent routing for telephone traffic: Theory and results. In *IEEE Conference on Decision and Control*, volume 25, pages 2124–2128.

Larsen, R. L. (1981). *Control of multiple exponential servers with application to computer systems*. PhD thesis, University of Maryland at College Park, College Park, MD, USA.

Lin, W. and Kumar, P. (1984). Optimal control of a queueing system with two heterogeneous servers. *IEEE Transactions on Automatic Control*, 29(8):696–703.

Lindvall, T. (1992). *Lectures on the Coupling Method*. Wiley.

Liu, Z. and Righter, R. (1998). Optimal load balancing on distributed homogeneous unreliable processors. *Operations Research*, 46(4):563–573.

Liu, Z. and Towsley, D. (1994). Optimality of the round-robin routing policy. *Journal of Applied Probability*, 31(2):466–475.

Puterman, M. L. (2005). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley.

Righter, R. and Xu, S. (1991). Scheduling jobs on nonidentical IFR processors to minimize general cost functions. *Adv. Appl. Prob.*, 23:909–924.

Véricourt, F. and Zhou, Y.-P. (2006). On the incomplete results for the heterogeneous server problem. *Queueing Syst. Theory Appl.*, 52(3):189–191.

Viniotis, I. and Ephremides, A. (1988). Extension of the optimality of the threshold policy in heterogeneous multiserver queueing systems. *IEEE Transactions on Automatic Control*, 33(1):104–109.

Virtamo, J. and Aalto, S. (1996). Procedure for controlling an elevator group. US Patent 5,503,249.

Walrand, J. (1984). A note on optimal control of a queueing system with two heterogeneous servers. *Systems and Control Letters*, 4:131–134.

Weber, R. R. (1978). On the optimal assignment of customers to parallel servers. *Journal of Applied Probability*, 15(2):406–413.

Winston, W. (1977). Optimality of the shortest line discipline. *Journal of Applied Probability*, 14:181–189.

A First policy iteration (FPI)

A standard MDP technique to derive a robust dynamic dispatching policy is to start with a static (i.e., a queue-state independent) basic policy α_0, and carry out one policy iteration round. Assuming a Poisson arrival process, i.i.d. jobs and a static dispatching policy, the system decomposes to k independent M/G/1 queues. Consequently, the value function of the system is obtained as a sum of the queue-specific value functions,

$$v(u_1, \ldots, u_k) = v^{(1)}(u_1) + \ldots + v^{(k)}(u_k).$$

The size-aware value function for M/G/1-FCFS (for later arrivals) has been derived in (Hyytiä et al., 2012b),

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

For M/D/1-FCFS queues (with $\Delta_i = 1/\nu_i$), the admission cost to Queue i is thus

$$a_i = (u_i + \Delta_i) + v_i(u_i + \Delta_i) - v_i(u_i) = (u_i + \Delta_i) + \frac{\lambda_i}{2(1-\rho_i)} \left(2u_i\Delta_i + \Delta_i^2\right),$$

where λ_i and $\rho_i = \lambda_i\Delta_i$ depend on the basic policy. The FPI policy chooses the action with the smallest expected cost,

$$\alpha = \operatorname*{argmin}_i a_i.$$

One reasonable choice for the basic policy is a random Bernoulli split (RND) that assigns a job to Server 1 with probability of p, and otherwise to Server 2. Choosing $p = \nu_1/(\nu_1 + \nu_2)$ balances the load between the two servers (which is not generally the optimal, however). Carrying out the FPI step yields a threshold dispatching policy with

$$\xi_{\text{FPI}} = (1-\gamma)(1-\rho/2).$$

For more details on the FPI approach, see, e.g., (Krishnan and Ott, 1986; Krishnan, 1990; Aalto and Virtamo, 1996; Bhulai, 2006; Hyytiä et al., 2012b,a), and the references therein.