Beyond the shortest queue routing with heterogeneous servers and general cost function

Esa Hyytiä
University of Iceland

Rhonda Righter
University of California Berkeley

Sigurður Gauti Samúelsson
Aalto University

ABSTRACT
Routing jobs to parallel servers is a common and important task in today’s computer systems. Join-the-shortest-queue (JSQ) routing minimizes the mean response time under rather general settings as long as the servers are identical and service times are independent and exponentially distributed. Apart from this, surprisingly few optimality results exist, mainly due to the complexities arising from the infinite state spaces. Indeed, it is difficult to analyze the performance of any given routing policy. In this paper, we consider an elementary job routing problem with heterogeneous servers and a general cost structure. By a novel approximation, we reduce the state space to finite size, which enables us to estimate the mean performance, and to determine (practically) optimal routing policies, for a large class of cost structures. We demonstrate the approximation and its application to optimization in numerical examples.

1. INTRODUCTION
Routing jobs to parallel servers has been a long standing problem class for queueing theory. The problem was first studied by Haight already in 1958 [1]. Today, the same problem arises in many new contexts. For example, when routing data traffic in Internet, alternative routes can be modelled as parallel servers. Similarly, in cloud computing, each task needs to be assigned to one of the available servers. In supercomputing, the time scales are longer but the same fundamental question appears. Moreover, the heterogeneity of computing hardware is increasing both in large-scale systems comprising several (thousands of) physical computers, as well as within a single physical device (cf. GPUs vs. CPUs, or new heterogeneous multi-core architectures such as those introduced by ARM for mobile devices, where some cores have higher capacity at the expense of higher energy consumption).

In this paper, we study one of the most elementary routing problems, where both job inter-arrival times and service times are exponentially distributed, so the state information is the number of jobs at each server. For clarity, we consider systems with two heterogeneous parallel servers subject to a large class of cost structures. The modelling approach itself generalizes straightforwardly to \( K > 2 \) servers at the cost of computational complexity. We discuss this later. One of the most popular routing policies is Join-the-Shortest-Queue (JSQ), which chooses the server with the fewest jobs. JSQ has been shown to be optimal in some specific cases, but, especially when the service rates are unequal, the exact analysis of the system becomes surprisingly tedious. The key idea in our approach is to accurately model the system where decisions matter the most, and rely on appropriate approximation elsewhere.

The main contribution of this paper is a novel modification of the system model with arbitrary cost structures, yielding a finite state space, which in turn enables us (i) to estimate the mean performance of arbitrary routing policies that are reasonable when the system is congested (i.e., stabilize the system), and (ii) to determine (near) optimal routing policies. Moreover, we obtain numerical evidence on how quickly policy iteration converges for this type of routing system. In particular, we observe that the first policy iteration round tends to yield the largest improvement (a phenomenon that has been assumed in numerous papers). These new (practically) optimal policies serve also as benchmarks when evaluating, e.g., simple (yet robust) policies such as JSQ.

1.1 Related Work
Routing problems have been studied actively during the last decades in very different contexts. Three classes of results are relevant to us: exact optimality results, approximate performance analysis, and heuristics for approximate optimization.

In terms of optimality results, Winston [2] showed that JSQ minimizes the mean response time under exponential assumptions.\(^1\) This result was then generalized by Weber [3] for arbitrary arrival process and i.i.d. service times with a non-decreasing hazard rate. See also, e.g., [4, 5, 6, 7].

The most common performance measure is mean response time, which is also non-trivial to compute in general for dynamic routing policies. However, under exponential assumptions good approximations exist for JSQ. For example, Nelson and Philips [8] develop a systematic approach based on the observation that the total number of jobs in the system under JSQ tends to behave similarly to the M/M/k system (with a shared queue). Then conditioning on the

\(^1\)Poisson arrival process, and exponentially distributed service times.
total number of jobs, one still needs to estimate the length of the shortest queue in order to find the steady state distribution. Our approach is based on the same observation, but we model the system accurately for states with a small number of jobs where the routing decision can be critical. See also [9]. Analysis of queueing systems tends to become harder when exponential assumptions are relaxed, including systems with JSQ routing. Some results do exist, e.g., Gupta et al. [10] consider JSQ with a general job size distribution and processor sharing (PS). With heterogeneous servers, the natural generalization of JSQ is the Shortest-Expected-Delay (SED) routing which chooses the server with the smallest expected response time. Selen et al. [11] show that the steady state distribution for two heterogeneous FCFS servers under SED can be expressed as a series of product forms that can be determined recursively, enabling the computation of, e.g., the mean response time, numerically.

The third class of results provide good (or optimal) routing heuristics that outperform JSQ and SED for heterogeneous systems with different cost structures. The basic routing problem is a classical Markov decision problem (MDP) with an infinite state space. If the state space were finite, the optimal routing policy would be trivially available (at least numerically) by carrying out policy or value iteration until it converges. In our setting, one often resorts to heuristic routing policies obtained by starting from a static policy, where the value function can be computed, and then carrying out one policy iteration round. This approach, referred to as first policy iteration (FPI), tends to yield an efficient, though generally not optimal, policy. The FPI approach has been utilized in numerous papers [12, 13].

We also study the routing problem in the MDP framework. Instead of trying to solve the original problem directly, we first develop an approximation for the system that has a finite number of states. Our approach is similar to the successive lumping method [14], where the state space is partitioned in such a way that the stationary distribution can be computed recursively (at least for finite systems). In contrast to [14], we partition the state space into two sets: the finite primary set includes states with few jobs where routing decisions tend to be most critical, and the secondary infinite set includes states with many jobs. Moreover, we first assume a fixed routing such as JSQ or SED in the secondary set, and then “compress” the infinite sub space to a single super state in a novel manner allowing us to handle also heavy load scenarios accurately. This approach enables us to analyze the system (with any load) and to compute the optimal routing policy exactly for the modified system. This in turn will provide an accurate and efficient heuristic for the original problem. Our approach also yields a computationally efficient procedure to estimate the mean performance (under any load) with respect to a large class of cost structures and routing policies (including JSQ and SED as special cases).

2. MODELLING

2.1 System A: Original model

The basic model we consider is illustrated in Figure 1 and is essentially the same as in [13]:

1. Jobs arrive according to Poisson process with rate $\lambda$ and they are routed immediately upon arrival to one of the available servers.

2. The system has $K$ parallel servers, where the service time at server $k$ is exponentially distributed with parameter $\mu_k$. In general, systems are heterogeneous, $\mu_i \neq \mu_j$. Let $\mu = \mu_1 + \ldots + \mu_K$.

3. We consider the number-aware setting, where state $n = (n_1, \ldots, n_K)$ means that server $k$ has $n_k$ jobs.

4. The admission cost function, denoted by $c_i = c_i^{(k)}$, defines the cost when a job is added to server $k$ currently having $i$ jobs. The admission cost is general and possibly server-specific function to accommodate unequal service rates $\mu_k$. As an example, we consider

(a) $c_i^{(k)} = (i + 1)/\mu_k$, i.e., the expected response time in server $k$

(b) $c_i = 1(i > 2)$, which represents how people tend to feel about queuing

Even though the model is elementary, finding the optimal routing policy is a surprisingly difficult problem due to the infinite state space. As mentioned, when $\mu_i = \mu_j$ and the objective is to minimize the mean response time, then JSQ is known to be optimal [2, 3]. With heterogeneous servers and arbitrary cost functions, one typically resorts to elementary heuristics like JSQ, or efficient routing policies based on FPI or Gittin’s index [13].

We assume that $c_i$ is some non-negative increasing function of $i$. Consequently, some kind of load balancing is eventually needed when the load increases, and multiple servers are required to support it. In particular, we assume that JSQ routing works reasonably well in higher states with many jobs, even though it is sub-optimal in general.

In what follows, we consider a small system with two servers, $K = 2$, but note that the developments generalize to arbitrary $K > 2$, as will be discussed later in Section 2.8. With $K = 2$, the model, referred to as System A, is a two dimensional Markov decision process.

2.2 System B: Fixed routing when many jobs

Next we will modify the system step-by-step, eventually obtaining an MDP with a finite state space, as illustrated in Figure 2. In the first step, we limit our focus on those states that we deem to be the most important:
Figure 3: Shorter queue upon returning to \(n \times n\) box for \(n = 4, 8, 16\). Typically the return point is close to (\(n, n\)), justifying the simplification referred to as the state-space collapse.

- Routing decisions tend to be most crucial when servers have only few jobs, and therefore we consider decisions only in a finite number of states near the origin,

\[
A = \{(i, j) \mid i < n, j < n\},
\]

where \(n\) is a free parameter (eventually defining the size of the final system’s state space).

- Elsewhere, i.e., whenever the length of at least one queue is \(n\) or more, a fixed policy, say JSQ, kicks in.

Define \(B\) as the union of \(A\) and its boundary,

\[
B = \{(i, j) \mid i \leq n, j \leq n\}.
\]

Hence, at this point, we have simply fixed the routing decision in states where we believe that JSQ (or a similar policy) is near optimal (in the sense that use of it does not significantly reduce the achievable mean performance). Heuristically, “when there is an abundance of jobs, keep all servers busy.” This system with routing policy fixed outside \(A\) is referred to as System B and depicted in Figure 2(b). Note that we still have an infinite state space to deal with, even though the routing action is free only in a finite number of states.

2.3 System C: (Partial) state-space collapse

Past work analyzing JSQ has made the important observation that in higher states, when both \(i, j \gg 0\), the system tends to stay near the diagonal and \(i \approx j\). In particular, [8] assumes that the total number of jobs, \(N = N_1 + N_2\), with JSQ behaves approximately the same way as in the \(M/M/2\) queue, yielding an approximation for the steady state distribution of \(N\). In fact, this is exactly the so-called heavy-traffic approximation [15]. We take advantage of the same phenomenon in this paper, but allow any stable load, \(0 < \rho < 1\).

First we note that in System B, for any \(n\), departures from set \(B\) to states outside it take place only through state \((n, n)\), while returns to set \(B\) can happen anywhere on the boundary, as illustrated in Figure 2(b). This property of having a single “exit gate” holds also for \(K > 2\) servers.

Example 1. Consider a system with two identical servers, \(\mu_1 = \mu_2 = 1\). The arrival rate \(\lambda\) is varied from zero to a heavily-loaded system with \(\lambda \approx 2\). Initially, the system is in state \((n + 1, n)\) corresponding to the first state after departing the \(n \times n\) box. Upon return, the longer queue has \(n\) jobs, and the shorter has a random number \(X \in \{0, \ldots, n\}\).

Figure 3 illustrates the mean and variability of \(X\) as a function of \(\lambda\). We can see that the variability is highest when \(\lambda \to 0\), which is easy to understand as arrivals will push the state closer to the diagonal under JSQ. For \(\lambda \to 0\), it is easy to show analytically that the difference between the shorter and longer queue, denoted by \(D\), has a truncated geometric distribution,

\[
P\{D = i\} = \left\{ \begin{array}{ll} q^i(1-q), & i = 0, \ldots, (n-1), \\ q^n, & i = n, \end{array} \right.
\]

where \(q = 1/2\), and as \(X = n - D\), we have

\[
E[X] = n - 1 + 2^{-n},
\]

\[
\sigma_X^2 = 2 - 4^{-n} - (1 + 2n)2^{-n},
\]

which rapidly converge to \(n - 1\) and \(2\), respectively, for large \(n\). Hence, when \(n\) is larger, typically the state in \(B\) where the system returns is one of \((n, n), (n, n-1), \ldots, (n, n-3)\).

This generalizes to \(K > 2\) servers, where the initial state is, say, \((n+1, n, \ldots, n)\) and the return state is \((n, X_2, \ldots, X_K)\), where the \(X_i\) are i.i.d. random variables with the same distribution as in above. Unequal service rates \(\mu_i \neq \mu_j\) can be worked out the same way.

With this insight in mind, we next propose that instead of analyzing the original model, we simplify our system by assuming that state-space collapse occurs beyond \(B\) [15]. More specifically, we define a new System C that agrees with System B for the states in \(B\), but for states \((i, j)\) in \(B^c\), reduces to a standard \(M/M/1\) queue with arrival rate \(\lambda\) and service rate \(\mu = \mu_1 + \mu_2\). When a job arrives at state \((n, n)\) in System C, it starts a mini-busy period during which jockeying\(^2\) is allowed and the system behaves as an \(M/M/1\) queue. Alternatively, one can think that for the higher states the two servers join forces and work on one job at a time until the state \((n, n)\) is reached again. In fact, System C corresponds a system, where each server has a finite number of system places, and if all those are full, then a job is stalled at the dispatcher, as illustrated in Figure 4. The cost rate in the higher states are denoted by \(\bar{r}\), where \(i\) denotes the total number of jobs in the system. Obviously, \(\bar{r}\) should resemble the corresponding exact cost rates \(r_{ij}\) near the diagonal, \(i \approx j\).

By doing this, we tend to underestimate the costs during the mini busy period a bit (depending on the cost structure). The state space of System C is depicted in Figure 2(c). It is significantly smaller than those of Systems A and B, but still infinite. We note that System C represents an approximation of System A; the costs incurred are very similar, but not exactly the same (even in expectation).

\(^2\)Jockeying refers to moving jobs between queues after the initial assignment.
2.4 System D: Aggregated super state

Let us next consider System C from the moment it enters state \((n, n)\) until it moves to state \((n - 1, n)\) or to state \((n, n - 1)\). This corresponds to a mini busy period in the M/M/1 queue initially having 2n jobs, with mean duration

\[
E\{B\} = \frac{1/\mu}{1 - \lambda/\mu} = \frac{1}{\mu - \lambda}
\]

Moreover, the fraction of time there are 2n + i jobs, \(i = 0, 1, \ldots\), is geometrically distributed,

\[
\pi_{2n+i} = (1 - \rho)\rho^i,
\]

and therefore the mean cost rate \(r_{n^*}\) during the mini busy period (where the job routing policy is “fixed”) is

\[
r_{n^*} = (1 - \rho) \sum_{i=0}^{\infty} r_{2n+i} \rho^i,
\]

where \(r_i\) denotes the cost rate with \(i\) jobs in the system.

Next we replace “the M/M/1 queue” in System C with an equivalent super state, state \(n^* = (n, n)\), which has mean duration \(E\{B\}\) and incurs costs at rate \(r_{n^*}\). This model is referred to as System D. The corresponding transition and cost rates of the (modified) Markov process for state \(n^*\) are,

\[
q_{n^*,(n-1,n)} = \mu_1(1 - \rho),
q_{n^*,(n,n-1)} = \mu_2(1 - \rho),
\]

\[
r_{n^*} = (1 - \rho) \sum_{i=0}^{\infty} r_{2n+i} \rho^i.
\]

Elsewhere within \(\mathcal{B}\), the transition rates are according to the service rates \(\mu_k\) and the arrival rate \(\lambda\) with the destination state defined by the routing policy (e.g., JSQ). The resulting Markov process of System D has \((n + 1)^2\) states as depicted in Figure 2(d), and can be easily analyzed. Note that in terms of (expected) costs, System C and D are equivalent.

Note that our modified model has (i) a finite state space \(\mathcal{B}\) with \((n + 1)^2\) states and (ii) finite well-defined cost rates in each state. For the interior points, i.e., states in \(\mathcal{A}\), we can choose the routing decision freely, i.e., whether to route a new job to Server 1 or Server 2. With a routing policy fixed, we have a finite Markov process which steady state distribution \(\pi_{ij}\) can be easily computed, and, e.g., the mean cost rate is \(r = \sum_{i,j} \pi_{ij} r_{ij}\).

2.5 Response time metric

Let us consider first the typical cost metric of minimizing the mean response time, for which the admission cost is

\[
c_i = \frac{i + 1}{\mu}.
\]

Equivalently, we can define cost rate according to the number of jobs in the system (cf. Little’s result), i.e.,

\[
r_{ij} = i + j, \quad \text{and} \quad \bar{r}_i = i,
\]

where the latter is for the states beyond \(B\). With these, the mean cost rate in the aggregated super state \(n^*\) is

\[
r_{n^*} = 2n + \frac{\rho}{1 - \rho},
\]

where the first term corresponds to the baseline of having at least 2n jobs, and the second term adds the mean number of additional jobs present during the mini busy period.

2.6 Queue length threshold metric

Let us next consider the non-linear sample cost metric defined by the admission cost

\[
c_i = 1(i > 2),
\]

i.e., in this case a unit cost is incurred if an arriving job sees more than two jobs ahead of itself upon arrival (in the same server).

Let \(a_{ij}\) denote the probability that an arriving job is routed to server 1 in state \((i, j)\), so with probability \(1 - a_{ij}\) it is routed to Server 2. Typically, \(a_{ij} = 0\) or \(a_{ij} = 1\), but this notation allows also probabilistic routing in every state.\(^3\)

Due to PASTA, instead of admission costs, we can define the equivalent cost rate in state \((i, j)\) as

\[
r_{ij} = \lambda (a_{ij}(i > 2) + (1 - a_{ij}))1(j > 2)).
\]

The above holds both for the original and modified system, and \(\bar{r}_i = \lambda\) given \(n > 2\). Hence,

\[
q_{n^*,(n-1,n)} = \mu_1(1 - \rho),
q_{n^*,(n,n-1)} = \mu_2(1 - \rho),
\]

\[
r_{n^*} = \lambda.
\]

With \(a_{ij}\) fixed, we again have a finite Markov process which steady state distribution \(\pi_{ij}\) and mean cost rate \(r\) can be easily determined.

2.7 Difference between Models

Let us next compare any two systems (a) and (b) that make the same decisions within \(\mathcal{B}\) and use JSQ on the boundary. This includes System B with arbitrary, but stable, routing decisions outside \(\mathcal{B}\), as well as Systems C-D. The long-run mean cost rates are \(\mu^{(a)}\) and \(\mu^{(b)}\), where the superscripts indicate the system, and in general \(\mu^{(a)} \neq \mu^{(b)}\).

Then consider an arbitrary state \(n \in \mathcal{B}\). As the two systems make the same decisions until reaching the corner point \(n^*\), their sample paths during this time interval are identical. It follows that for the value functions, \(\psi_n^{(a)}\) and \(\psi_n^{(b)}\),

\[
v_n^{(a)} - v_{n^*}^{(a)} = \mathbb{E}\{C(n, n^*) - r^{(a)}(T(n, n^*))\},
\]

\[
v_n^{(b)} - v_{n^*}^{(b)} = \mathbb{E}\{C(n, n^*) - r^{(b)}(T(n, n^*))\},
\]

where \(C(n_1, n_2)\) and \(T(n_1, n_2)\) denote the costs incurred and the duration of time before a system initially in state \(n_1\) reaches state \(n_2\) for the first time.\(^4\) Due to the identical routing decisions within \(\mathcal{B}\), the only difference on the right-hand side is in the mean cost rates, \(r^{(a)} \neq r^{(b)}\). Given that, for \(n\) sufficiently large, \(r^{(a)} \approx r^{(b)}\) the corresponding relative values are also practically identical (within \(\mathcal{B}\)).

If (a) is System B with \(K\) parallel queues with any, including the optimal, stable routing, except for on the boundary, and (b) is a system where outside \(\mathcal{B}\) the system is reduced to an M/M/1 queue (or an equivalent super state), then costs such as the mean response time are clearly smaller, \(r^{(b)} < r^{(a)}\), because (a) could have idling in states outside \(\mathcal{B}\). Later we show numerically that with respect to mean.

\(^3\)For example, the load balancing random split (RND) is defined by \(a_{ij} = \mu_1 / (\mu_3 + \mu_2)\).

\(^4\)Considering sample paths until reaching state \(n^*\) instead of, e.g., the origin has the benefit that no state outside \(\mathcal{B}\) is visited before termination. This holds as applying JSQ on the boundary forms a “surface” that ensures that the corner point \(n^*\) is the only exit gate from \(\mathcal{B}\) to outside.
Table 1: Estimates for the mean number of jobs with two identical servers and JSQ based on the modified model with $n = 0, \ldots , 3$.

response time, $r^{(a)} \leq r^{(b)}$ for $n > 2, n$ when the servers are identical and the optimal routing policy JSQ is applied in every state in both ($a$) and ($b$). Hence, also the corresponding value functions are practically equivalent, and the adverse effects of simplifying the system to a finite Markov process are negligible.

2.8 General case with $K$ servers and arbitrary box as a boundary

In this section, we illustrate how the approach generalizes to $K > 2$ servers and an arbitrary box defining the boundary for $B$. As before, we assume that the routing on the boundary ensures a single exit state (see Figure 2(c)), and that the fixed routing policy outside $B$ is such that a state-space collapse occurs and the return state is (approximately) the same as the exit state, so the M/M/$1$ model for the mini busy period is justified.

Let $K$ denote the number of servers and $m = (m_1, \ldots , m_K)$ the dimensions of the finite state space, where $m_k$ is the maximum number of jobs in server $k$ we are tracking. Hence, the number of states is

$$M = \prod_{k=1}^{K} (m_k + 1).$$

An arbitrary state is denoted with $n = (n_1, \ldots , n_K)$, where $n_k$ is the number of jobs in server $k$. We can easily map the $K$-dimensional state space to one dimension using

$$s(n) = \sum_{i=1}^{K} \left( n_i \prod_{k=1}^{i-1} (m_k + 1) \right).$$

Then an arbitrary probabilistic routing is defined by an $M \times K$ matrix $\alpha$, where $\alpha_{ik}$ defines the fraction of jobs routed to server $k$ in state $i$. We assume that $\alpha$ honors the boundaries, so that, e.g., $\alpha_{ik} = 0$ for all $k$ (in the full system, arriving jobs are “blocked”). With the routing policy fixed, the transition rate matrix $Q$ is easy to obtain.

Consider first the departure rates. For an arbitrary state $n$, given $n_k > 0$ and server $k$ is busy, the corresponding departure rate shows up in $Q$ as

$$q_{s(n),s(n-e_k)} = \mu_k,$$

where $e_k$ denotes a vector with all elements zero except the $k$th element that is one. For the combined super state $n = m$, i.e., $i = M$, we have

$$q_{s(m),s(m-e_k)} = (1 - \rho)\mu_k, \quad \forall k.$$
Figure 6: Mean occupation scaled by $\rho/(1 - \rho)$ with JSQ and $K = 2, 3, 4$ identical servers (solid lines). Lower bounds with $n = 1$ (i.e., M/M/K system) are depicted with dashed lines.

literature (also for $k > 2$ servers). For two identical servers, Blanc [9] gives,

$$\mathbb{E}(N) \approx \frac{\rho (4 + 10\rho - 5\rho^2)}{(1 - \rho)(7\rho + 2)},$$

whereas Nelson’s and Philips’ approximation [8] in the same case is,

$$\mathbb{E}(N) \approx \frac{2\rho (1 + \rho + \rho^2 - \rho^3)}{(1 - \rho)(1 + \rho)^2}.$$

The accuracy of Blanc’s expression is approximately the same as that of $\mathbb{E}(N_2)$, while the accuracy of Nelson’s and Philips’ expression is somewhere between those of $\mathbb{E}(N_2)$ and $\mathbb{E}(N_1)$ (in terms of maximum relative error), see Figure 5. In any case, all these approximations are easy to evaluate and they all (except $\mathbb{E}(N_1)$) behave correctly at the limits (1) and (2). For us, these approximations are “a free side product” as our main goal is to find the (near) optimal routing policy for systems with heterogeneous parallel servers, and the modified model with a finite state space is designed with this goal in mind. Note that our approximations are also lower bounds.

3.2 Three and four servers

Let us now use our approach to a bit larger systems of three and four identical servers.

The mean number of jobs $\mathbb{E}(N)$ in the M/M/1 queue is $\rho/(1 - \rho)$. With $K$ parallel servers, having an equal total service rate, and fed by JSQ, $\mathbb{E}(N)$ is obviously higher as sometimes servers can be idle even if there are jobs in the system. Figure 6 illustrates the penalty, based on our approximation, due to having multiple servers instead of a single faster one for $K = 2, 3, 4$. The solid curves correspond to estimates obtained using sufficiently large values of $n$, and they can be easily verified to be surprisingly accurate with simulations. Dashed lines correspond to the lower bounds obtained with $n = 1$, i.e., to the M/M/K system (where routing is replaced with a common queue).

### 4. OPTIMAL ROUTING POLICY

In this section, we shift our focus to finding the optimal routing policy. To this end, we consider the modified System D and determine the optimal routing for it. This is then assumed to serve as a (near) optimal routing policy also for the original System A (within $\mathcal{B}$).

Recall that we managed to reduce the infinite state space of the original system to a classical MDP problem with a finite number of states and well-defined cost rates in each state. Such problems are commonly solved by using policy or value iteration methods [16, 17]. The former involves solving Howard’s equations yielding relative values.

Let $n_k$ denote the number of jobs in server $k$, so that the state of the whole system is $n = (n_1, n_2)$. With these, Howard’s equations are

$$r_n - r + \sum_{n' \neq n} q_{n'n}(v_{n'} - v_n) = 0,$$

where $r_n$ is the cost rate in state $n$ and $r$ is the mean cost rate. Fixing, e.g., $v_0 = 0$, the above set of linear equations can be easily solved, yielding both the value function $v_n$ and the mean cost rate $r$. Next the policy iteration step is carried out,

$$\alpha^*(n) \equiv \arg \min_k \left( c_{n_k}^{(k)} + v(n + u_k) - v(n) \right),$$

where $c_{n_k}^{(k)}$ is the admission cost to server $k$ at state $n_k$, and $u_k$ denotes a vector with 1 at position $k$ and otherwise zero. This is repeated until the procedure converges ($r$ remains the same). Typically, policy iteration converges rapidly, and later in the examples we see that this is the case also here.

### 4.1 Numerical examples

Next we will illustrate the procedure and optimal policies for heterogeneous two server systems with $\mu_1 \geq \mu_2$. As a reference, we consider the following three heuristic policies:

1. Load balancing random split (RND) that chooses the server $k$ with probability of $\mu_k/(\mu_1 + \mu_2)$ (JSQ); and (iii) Shortest Expected Delay (SED) that chooses the server that minimizes the expected response time [11, 14]. Ties with JSQ and SED are resolved in favor of the faster or slower server. We indicate the tie breaking rule in parentheses, e.g., JSQ(1) resolves ties in favor of the faster Server 1.

**Example 2.** Suppose $(\mu_1, \mu_2) = (h \mu, \mu)$, where $h \geq 1$ measures the asymmetry in the service rates. The modified system with JSQ and $n = 2$ is the smallest system with a non-trivial routing decision. That is, which server should be chosen in state $(1, 0)$? When $\rho \to 0$, the greedy SED policies are optimal and one chooses the faster server only if $h > 2$.

With $n = 2$, the system has 9 states and both steady-state distribution and value function can be easily computed analytically, and the threshold for routing also the second job to Server 1 can be computed. It turns out that the threshold increases almost linearly from 2 to 3.74 as $\rho$ increases from zero to one. That is, when the load is higher the secondary server is taken into use earlier. This observation holds also when $n > 2$ with slightly changed numerical values.
Figure 8: Routing policies for $(\mu_1, \mu_2) = (3,1)$ system with $n = 16$, where lighter (yellow) states are those in which jobs are routed to Server 1. From left to right, SED(1), SED(2), and the optimal routing for $\rho = \{0.25, 0.75, 0.98\}$. The higher the load, the more aggressively the optimal policy utilizes the slower secondary server.

Figure 9: Sub-optimal routing policies JSQ and SED with different tie breaking rules compared to the optimal routing when $(\mu_1, \mu_2) = (3,1)$.

**Example 3.** Suppose $\lambda = 3$, $(\mu_1, \mu_2) = (3,1)$, i.e., the secondary Server 2 is three times slower than the primary server, and the offered load is relatively high, $\rho = 0.75$. The parameter $n$ we set to 16.

Figure 7 illustrates the convergence of policy iteration when starting from three different basic policies. On the $x$-axis is the iteration round (zero corresponds to the basic policy), and on the $y$-axis is the mean number of jobs in the system. We observe that it takes 3-5 rounds before the optimal policy is found, and that the largest improvement from every basic policy indeed takes place in the first step, which supports the claim that FPI policies are often “near-optimal”, and the basic policy does not matter much.5

Now we take a closer look at when SED and optimal routing policies make different decisions. SED(1) and SED(2) are illustrated in Figure 8 (left), and the optimal policy for $\rho = 0.25, 0.75$ and 0.98 in the following three graphs on the right. First we observe that SED(2) appears to be optimal when load is low. As the load increases, the optimal policy routes the first job “earlier” to Server 2 in anticipation of new jobs arriving soon. The higher the load, the more pronounced the proactive action is. Otherwise the “ladder” shows “three jobs to faster server, and then one to slower” pattern, as expected.

Figure 9 illustrates the loss in terms of relative increase in mean response time when JSQ and SED with different tie breaking rules are used instead of the optimal policy. As expected, JSQ and SED are good and robust routing policies but not optimal (in this case). With JSQ it is clearly important to favor the faster Server 1 so that the first job goes there. SED routes the first job automatically to the faster server, and it is actually better to route a job to slower Server 2 in case of ties. We can observe that JSQ(1) increases the mean response time by about 10% and SED(2) up to 2%, except when $\lambda$ is (very) low or high.

**Example 4.** Next we compare a single fast server with $\mu = 4$ to (i) two identical servers with $(\mu_1, \mu_2) = (2,2)$, (ii) two heterogeneous servers with $(\mu_1, \mu_2) = (3,1)$, and (iii) two heterogeneous servers with $(\mu_1, \mu_2) = (3.5,0.5)$. With identical jobs and the mean response time metric, the single fast server is obviously the optimal configuration. As for the two server systems, we consider both JSQ(1) and SED(2), and optimal routing. As some systems are highly asymmetric, it is important to use an appropriate $n \times n$ box instead of an $n \times n$ square (see Section 2.8) for SED and optimal routing. The numerical results are depicted in Figure 10. Note that especially JSQ(1) suffers from heterogeneity in the sense that a lower mean response time can be achieved with two identical servers even though JSQ(1) favors the faster server.6 In the homogeneous case, JSQ and SED coincide with the optimal policy. With the optimal routing policy, the mean response time decreases as the heterogeneity increases, as expected.

**Example 5.** Let us next consider the unit step function $c_i^{(4)} = 1(i > 2)$ as the admission cost. The service rates are again $(\mu_1, \mu_2) = (3,1)$ and the offered load is $\rho = 0.5$. For policy iteration, the immediate (admission) cost $c_i = 1(i > 2)$ must be taken explicitly into account. The optimal routing policy is depicted in Figure 11(left). We note that when both queues are too long, the optimal policy routes the new job to the slower server, thus minimizing the time until a job

5When RND is applied in every state, $E\{N\} = 6$. However, we assumed JSQ/SED outside $A$, and this is why RND (within $A$) has a bit better performance. For us, this discrepancy is irrelevant as our focus is on the optimal dynamic policies, for which JSQ/SED at higher states is a fair choice.

6In fact, when the asymmetry increases a bit more, JSQ(1) becomes worse than the simple load balancing RND.
can be admitted to the system without a penalty. However, overloading (usually) the slowest server can lead to instability issues when ρ is sufficiently high. This is actually an artifact of the cost model as it may well be beneficial to overload one queue in order to keep the others sufficiently short. However, the assumed JSQ/SED beyond B ensures stability as long as ρ < 1.

Example 6. Finally, let us consider a non-linear cost structure, where we combine the response time metric and the unit cost if queue length exceeds the chosen threshold of 2. In this case, the cost rates in each state are simply summed up. For example, in the super state $n^*$ we have

$$r_{n^*} = 2n + \frac{\rho}{1-\rho} + \lambda.$$

As before, for policy iteration, the immediate (admission) cost $c_i = 1(i > 2)$ must be taken into account, whereas for response time this cost is included in the state-specific cost rates.

The optimal routing policy is depicted in Figure 11(right) for ρ = 0.5. Interestingly, in this case jobs are routed to the slow secondary server only when its queue length is below the threshold 2. Having the response time component in the cost structure discourages overloading one queue and thus prevents instability issues. In our case, the assumed JSQ outside B also ensures stability as long as ρ < 1.

5. CONCLUSIONS

In this paper, we have studied the classical routing problem to parallel heterogeneous servers with Poisson arrivals, exponential services, and arbitrary cost structure arbitrary. Although JSQ is a widely used dynamic routing policy for such systems, it is not generally optimal as it neglects both the service rates and the cost structure.

We proposed an approach where the infinite state space is “compressed” to $(n+1)^2$ states, of which one is a super state corresponding to collapsed version of larger states. For small values of n, we obtain closed-form results and policies in symbolic form (with arbitrary λ and $\mu_i$). Numerically the proposed approach is very efficient for a large class of routing problems with arbitrary admission costs and levels of offered load. Our numerical examples support the common claim that the first policy iteration step tends to yield the highest improvement in performance. In our example, it often yielded the near-optimal performance.

6. REFERENCES