PERFORMANCE-BASED ROUTING

AVISHAI MANDELBAUM AND PETAR MOMČILOVIĆ

ABSTRACT. In many-server systems with heterogeneous servers, the Fastest-Server-First (FSF) policy is known for its excellent performance. However, when service rates are unknown and/or time-varying, implementing the FSF policy is not straightforward. We thus propose a routing algorithm, Performance-Based Routing, that approximates the FSF policy: servers are ranked in a dynamic list, where the shorter the actual service times that a server exhibits – the closer the server is to the head of the list; a customer is then routed to the lowest-index (highest-in-the-list) idle server. It is argued that the algorithm is asymptotically equivalent to FSF.

1. INTRODUCTION

1.1. **Motivation.** The Fastest-Server-First (FSF) routing policy [2] assigns customers to a server with the highest service rate. Due to its excellent performance and intuitive nature, this policy serves as a benchmark in many-server queues, when the goal is to minimize customer waiting/sojourn times. The FSF policy belongs to a class of rank-based algorithms (also known as order-entry systems [14]): servers are represented by a ranked list, and a customer is routed to the idle server with the lowest index; in the FSF case, the servers are sorted according to their service rates.

Implementing FSF in practice is not always straightforward due to two factors: (i) server rates might not be known – only rate estimates can be obtained by considering samples of service times [4]; and (ii) server rates might vary over time. The algorithm we propose, Performance-Based Routing (PBR), overcomes these challenges by dynamically rearranging the server list and routing customers to the available server with the lowest index (highest rank). The algorithm reorganizes the list, based on observed service times, in such a way that (on average) the faster the server the closer it is to the beginning of the list. Like the well-known $c\mu$ (or G$c\mu$ [12, 11]) rule, the PBR algorithm does not require any knowledge about the arrival process – it performs well regardless of the arrival rate. In particular, it is robust across operational regimes, and it adapts to changing workloads. Finally, PBR is a low-complexity algorithm as each service completion triggers at most one transposition of server rankings in the list.

1.2. **Model and assumptions.** We consider a sequence of first-come-first-served queues indexed by the number of servers N. Customers arrive to a single queue with parallel servers and finite or infinite waiting room (inverted-V model). Arrivals to the Nth system form a Poisson process with rate λ^N; we omit the superscript N when the arrival rate does not vary with the system size. Servers are labeled by integers $\{1, 2, \ldots, N\}$. For the Nth system, service times are independent across servers, and for a given server, say i, the sequence of service durations is i.i.d. with elements equal in distribution to S_i^N; the random variable S_i^N is exponential, with the service rate given by $\mu_i^N = 1/\mathbb{E}S_i^N$. The parameters $\{\mu_i^N\}$ are deterministic and, without

Date: July 26, 2014.
2000 *Mathematics Subject Classification.* Primary: 68M20; Secondary: 90B22.
Key words and phrases. Multi-server queue, routing, transposition rule.

loss of generality, $\mu_1^N \geq \mu_2^N \geq \cdots \geq \mu_N^N$. Service rates are bounded, i.e., $\mu_i^N \leq \mu$, where $\mu < \infty$ does not change with N. The service capacity of the N-server system is thus given by

$$C^N = \sum_{i=1}^{N} \mu_i^N.$$

A routing policy specifies to which server a customer is routed, provided that there are idle servers (it is hence nonpreemptive). The FSF policy serves as our benchmark policy. Under FSF routing, a customer is routed to an idle server with the highest service rate. Note that this policy can be implemented only if the server rates are known to the router.

Our focus is on many-server asymptotics, namely $N \to \infty$. (Throughout the paper, we use the standard o, O and Θ asymptotic notation [6, Sect. 3.1].) The considered system is related to rank-based systems [14]. These are characterized by a vector $l = (l_1, \ldots, l_N)$, which is some permutation of $(1, 2, \ldots, N)$: a customer is routed to server l_i only if servers $l_1, \ldots, l_{i-1}$ are busy; for example, FSF routing corresponds to $l = (1, 2, \ldots, N)$, in view of our assumed order on μ_i^N. Roughly speaking, in a rank-based system, servers can be classified into three groups: (i) those that are busy with probability close to 1, (ii) those that are idle with probability close to 1, and (iii) those that are busy/idle a constant fraction of their time. Informally, in the efficiency-driven (ED) regime (load close to capacity in the sense that $\lambda^N \approx C^N - k$, for some fixed $k > 0$), $\Theta(1)$ servers are in the third group (servers with indices corresponding to the last $\Theta(1)$ dimensions of l), while all other servers are in the first group. On the other hand, in the quality-driven (QD) regime (load being a constant fraction of the capacity: $\lambda^N \approx \gamma C^N$ for some $\gamma \in (0, 1)$; a negligible fraction of customers experience delay), $\Theta(N)$ servers are in the first group (servers with indices $l_1, l_2, \ldots$), $\Theta(N)$ servers are in the second group (servers with indices $\ldots, l_{N-1}, l_N$), and $\Theta(\sqrt{N})$ servers are in the third group. Finally, in the quality-and-efficiency-driven (QED) regime (load and capacity relate via the square-root rule: $\lambda^N \approx C^N - \beta\sqrt{C^N}$; a constant fraction of customers experience delay), $\Theta(N)$ servers are in the first group (servers with indices $l_1, l_2, \ldots$), no servers are in the second group, and $\Theta(\sqrt{N})$ servers are in the third group (servers with indices $\ldots, l_{N-1}, l_N$).

Now, compare an FSF system to another rank-based system, characterized by a vector l. We note that the two systems can be asymptotically equivalent even if the server ordering for the first two groups, the very-busy and -idle servers, differs. By equivalence we mean, again informally, that the stationary numbers of customers in the two systems are equal, say on a diffusion scale. For example, let

$$\mu_i^N = \begin{cases} 3, & 1 \leq i \leq \lceil N/3 \rceil, \\ 2, & \lceil N/3 \rceil < i \leq \lceil 2N/3 \rceil, \\ 1, & \lceil 2N/3 \rceil < i \leq N, \end{cases}$$

and $\lambda^N = 11N/6$ (the first $5N/6$ servers are sufficient to keep the system stable). Then, the FSF system operates in the quality-driven regime ($\lambda^N \approx \gamma C^N$ with $\gamma = 11/12$). It can be shown that all servers with rates 3 and 2 are busy with probability close to 1, and that only servers with rate 1 can be idle for a non-negligible fraction of time. Moreover, a particular ordering of the servers with rates 3 and 2 does not play a role as the size of the system increases. For example, if $l = (\lceil N/3 \rceil + 1, \ldots, \lceil 2N/3 \rceil, 1, \ldots, \lceil N/3 \rceil, \lceil 2N/3 \rceil + 1, \ldots, N)$, then the system is asymptotically equivalent to the FSF system ($l = (1, 2, \ldots, N)$). On the other hand, when $\lambda^N = 2N/3$, the ordering of servers with rates 2 and 1 has asymptotically negligible effect, as long as fast servers (rate 3) correspond to the first indices of l, or equivalently,

$$\sum_{i=1}^{\lceil N/3 \rceil} \left(\mu_i^N - \mu_{l_i}^N\right) = N - \sum_{i=1}^{\lceil N/3 \rceil} \mu_{l_i}^N = 0.$$

The preceding equality arises from a particular choice of the input rate λ^N, as well as the structure of the sequence $\{\mu_i^N\}$. In order to avoid the dependency on λ^N and $\{\mu_i^N\}$ (that is, provide robustness across operational regimes), we introduce the following quantity that measures closeness to FSF: given a server ordering l, let

$$\Delta(l) = \max_{i\in\{1,\dots,N\}} \sum_{j=1}^{i} \left(\mu_j^N - \mu_{l_j}^N\right). \tag{1}$$

Note that $\Delta(l) \geq 0$ ($\sum_{j=1}^{i} \mu_j^N \geq \sum_{j=1}^{i} \mu_{l_j}^N$, $i = 1,\dots,N$), with equality if and only if l corresponds to an FSF system. Informally, when $\Delta(l)/\sqrt{N}$ vanishes, as $N \to \infty$, a rank-based QD or QED system defined by a vector l is asymptotically equivalent (in terms of the number of customers in the system) to the corresponding FSF system on the diffusion ($\sqrt{N}$) scale.

1.3. **Brief literature review.** Rank-based routing policies, defined by fixed vectors, were studied in [14]. In particular, the authors considered relative performance of two systems defined by two different vectors. A rank-based system can be viewed as an extension of the well-known M/M/∞ storage process [13, 5, 1]: it consists of an infinite number of i.i.d. servers; these are indexed by the natural numbers, and a customer is routed to the lowest-index idle server. An FSF system with random server rates was studied in [3]. The author established a central limit theorem for the number of customers in the system when the system is in the QED regime. Under this model, roughly speaking, only servers with the slowest rates experience idleness. An asymptotic optimality (in the QED regime) of the FSF policy was shown in [2]. In [4], the authors consider a many-server QED system. Service rates of servers are random and do not change over time, but they are unknown to the router. Before the system starts operating, the router obtains samples of service times (individual realizations, one service time per server) and, based on these observations, decides on a (fixed priority) routing policy. This sampling occurs only once, since server rates do not change over time. Due to the QED regime, it is sufficient to identify $\sqrt{N}$-order servers with server rates close to the minimum possible rate (since only those servers have non-negligible idle times) so that the system remains asymptotically optimal. The authors show that it is sufficient to sample $N^{1/2+\delta}$ servers, for some arbitrary $\delta > 0$.

2. Linear List

We use a linear list to describe the state of our system (servers), operating under PBR. Upon a service completion by a server in position $i \geq 2$ in the list, this server is moved forward by one position if the server in position $(i-1)$ is busy and *eligible* for a move; the latter server is moved back one position in that case, which entails that the servers in positions $(i-1)$ and i are transposed [10, Sect. 6.1]. Once a (busy) server is moved one position down in the list, it becomes *ineligible* for a move until the server in front of it becomes busy. A service completion by the first server in the list does not trigger a rearrangement of the list.

The idea behind PBR is to thrive to a list that is ordered based on service rates – the higher the service rate, the closer the server should be to the beginning of the list. The motivation for the rule according to which the list evolves is as follows.

- *Why transposition of busy servers?* Consider two busy servers located in adjacent positions of the list. If the server that is lower in the list completes service earlier than the higher one, we use this as an indication that the order of these servers should be reversed. We do however require that servers are transposed only if both of them are busy. This condition is required in order to avoid scenarios where adjacent servers are first transposed and then transposed again before the server that moved up in the list becomes busy again. (Note that a service completion by a server while the

adjacent server is idle does not convey any information about their relative rates.) A consequence of not implementing this rule is illustrated in Example 1 below.

- *Why the eligibility rule?* This rule is motivated by the light-load regime. The idea is to prevent a particular server from sliding down in the list "too quickly". To illustrate this point, consider a list of 4 busy servers sorted in the decreasing order of their service rates; if $\lambda \downarrow 0$, new arrivals are unlikely. Without the extra condition we impose, by the time all servers are idle, it is possible that the server initially in the first position (the fastest one) has moved to the last position (the server in the second position completes service first, then the one in the third position, and finally the one in the last position). With the eligibility condition, however, a server can drop only one position at a time. Indeed, as soon as the server initially in the first position moves to the second position, the server that moves to the first position is idle, and thus initially the highest server is ineligible to slide further in the list.

The state of the system at time t is described by a triple $(\mathcal{L}^N(t), \mathcal{B}^N(t), Q^N(t))$; the process $\{(\mathcal{L}^N(t), \mathcal{B}^N(t), Q^N(t)),\ t \geq 0\}$ is right-continuous. The vector $\mathcal{L}^N(t) \in \mathscr{L}^N$ represents the state of the list at time t, where $\mathscr{L}^N$ is the set of all permutations of the vector $(1, 2, \ldots, N)$. In particular, $\mathcal{L}^N(t) = (\mathcal{L}^N_1(t), \ldots, \mathcal{L}^N_N(t)) = (l_1, \ldots, l_N)$ indicates that, at time t, the server with index l_i is in the list position i. The vector $\mathcal{B}^N(t) \in \{0, 1, 2\}^N$ indicates the set of servers that are busy at time t. In particular, if $\mathcal{B}^N_i(t) = 1$, then the server in the ith position in the list is busy and eligible for a move at time t; when $\mathcal{B}^N_i(t) = 0$, the server is idle; when $\mathcal{B}^N_i(t) = 2$, the server is busy, but ineligible for a move. The number of customers awaiting service at time t is $Q^N(t)$. In a loss system, $Q^N(t) \equiv 0$ for all t (recall from Section 1.2 that the system can have either a finite or infinite buffer).

Example 1 (Motivation). Consider a two-server loss system ($N = 2$) with $\mu = \mu_1 \geq \mu_2 = 1$, and some $\lambda \in (0, \infty)$. (In this example we omit the superscript N in order to simplify the notation.) Then there exist eight possible states for the pair $(\mathcal{L}, \mathcal{B})$; let π be the stationary distribution of this pair. In a two-server system, the eligibility does not need to be considered explicitly, since it is implied by the "busy" rule. Straightforward calculations yield:

$$\frac{\pi((1,2),(1,1))}{\pi((2,1),(1,1))} = \frac{\mu_1}{\mu_2}, \qquad \frac{\pi((1,2),(1,0))}{\pi((2,1),(1,0))} = \frac{\lambda+\mu_1}{\lambda+\mu_2},$$
$$\frac{\pi((1,2),(0,1))}{\pi((2,1),(0,1))} = \frac{(\lambda+\mu_1)\mu_1}{(\lambda+\mu_2)\mu_2}, \qquad \frac{\pi((1,2),(0,0))}{\pi((2,1),(0,0))} = \frac{(\lambda+\mu_1)^2}{(\lambda+\mu_2)^2}.$$

Note that, in stationarity, the routing algorithm results in $\{\mathcal{L} = (1,2)\}$ being more probable than $\{\mathcal{L} = (2,1)\}$, regardless of the value of $\mathcal{B}$, i.e., PBR routing biases the list toward the state in which servers are sorted in a decreasing order of rates. Additional calculations yield that there exists a value of the arrival rate λ that results in minimum preference of the server ordering $(1,2)$ over $(2,1)$, as measured by the ratio $\mathbb{P}[\mathcal{L} = (1,2)]/\mathbb{P}[\mathcal{L} = (2,1)]$. In Figure 1, we plot the minimum value of the scaled ratio

$$\mu^{-1}\frac{\mathbb{P}[\mathcal{L} = (1,2)]}{\mathbb{P}[\mathcal{L} = (2,1)]},$$

as well as the corresponding value of λ that achieves that minimum. It can be verified that

$$\lim_{\mu\to\infty} \mu^{-1} \inf_{\lambda} \frac{\mathbb{P}[\mathcal{L} = (1,2)]}{\mathbb{P}[\mathcal{L} = (2,1)]} = 1/2.$$

Thus, even in the worst-case scenario (in terms of the arrival rate), $\{\mathcal{L} = (1,2)\}$ is more likely than $\{\mathcal{L} = (2,1)\}$.

Finally, this example also illustrates why we require $\mathcal{B}_1 = 1$ for the two servers to be transposed. To wit, consider the algorithm that transposes the order of servers (when the

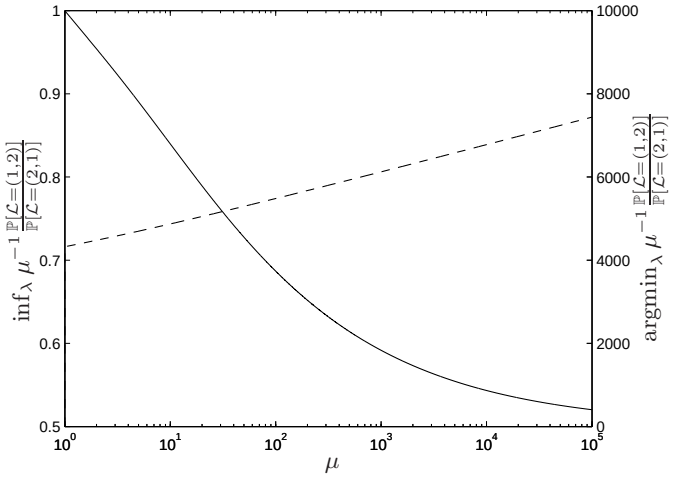

FIGURE 1. Illustration for Example 1. Depicted is the minimum value of the ratio $\mu^{-1}\mathbb{P}[\mathcal{L}=(1,2)]/\mathbb{P}[\mathcal{L}=(2,1)]$, as a function of the faster server speed μ (solid line); the corresponding value of the arrival rate λ that achieves this minimum is shown as well (dashed line).

second server in the list completes service) regardless of the value of $\mathcal{B}_1$; let $\tilde{\pi}$ the stationary distribution of $(\mathcal{L},\mathcal{B})$ under this modified algorithm (a server is moved up in the list by one position whenever the server completes a service). Then, we have

$$\frac{\tilde{\pi}((1,2),(1,1))}{\tilde{\pi}((2,1),(1,1))} = \frac{\mu_1}{\mu_2}\frac{\lambda(\lambda+\mu_1)+\mu_2(\lambda+\mu_2)}{\lambda(\lambda+\mu_2)+\mu_1(\lambda+\mu_1)}, \qquad \frac{\tilde{\pi}((1,2),(1,0))}{\tilde{\pi}((2,1),(1,0))} = \frac{\lambda+\mu_2}{\lambda+\mu_1},$$
$$\frac{\tilde{\pi}((1,2),(0,1))}{\tilde{\pi}((2,1),(0,1))} = \frac{(\lambda+\mu_1)\mu_1}{(\lambda+\mu_2)\mu_2}, \qquad \frac{\tilde{\pi}((1,2),(0,0))}{\tilde{\pi}((2,1),(0,0))} = 1.$$

Note that, in the limit as $\lambda \downarrow 0$, the two states of $\mathcal{L}$ are equally likely since $\mathbb{P}[\mathcal{B}=(0,0)] \to 1$ as $\lambda \downarrow 0$, i.e., the servers are unordered or, equivalently, no preference is given to faster service. □

In the following section, we analyze the PBR policy in two asymptotic regimes: saturated and light load. In these two regimes, analyses are feasible since one does not need to keep track of the vector $\mathcal{B}^N$ – in the saturated regime, $\mathcal{B}^N$ is a vector of ones, while it is a vector of zeros in the light-load regime. We now provide an example of a system that operates in an intermediate regime, to illustrate that the algorithm performs well in such a regime as well. This example also demonstrates that the algorithm adapts to time-varying conditions.

Example 2 (Quality-and-efficiency driven (QED) regime). Consider two infinite-buffer 100-server systems, where the service rate of server i is $\mu_i = 1-(i-1)/100$, $i=1,\ldots,100$. The first system operates under PBR routing, while the second system uses FSF routing. The two systems are subject to the same Poisson arrival stream of customers (arrival times, service requirements). Initially, at time $t=0$, servers in the list (for the first system) are ordered in an increasing order of their rates (the first server in the list is slowest); also, there are 60 customers in both systems – the 60 slowest and fastest servers are initially busy in the PBR and FSF systems, respectively. The arrival rate λ is taken to be

$$\lambda = \sum_{i=1}^{60} \mu_i = 42.3,$$

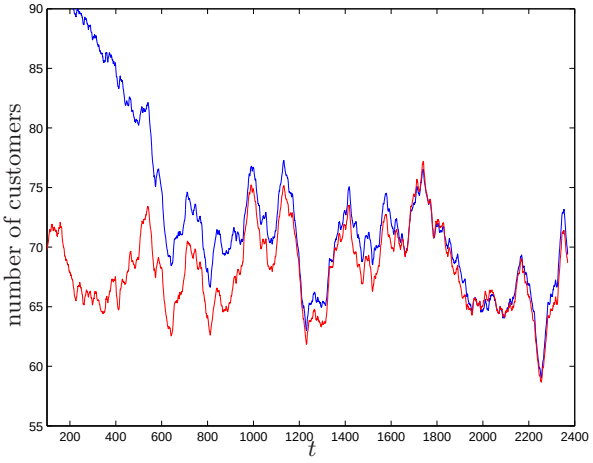

FIGURE 2. Illustration for Example 2. Performance of a PBR system (blue line) compared to the FSF system (red line) under an intermediate regime. The arrival and service times are equal in the two systems. At time $t = 0$, we started the PBR list in the reversed order (slowest at the top). Averages over the trailing 100 time units are shown, starting with $t = 100$.

i.e., the 60 fastest servers are sufficient to keep the both systems critically stable. The system capacities are $C = \sum_{i=1}^{100} \mu_i = 50.5 \approx \lambda + 1.26\sqrt{\lambda}$, i.e., the systems adhere to the so-called square-root staffing root and operate in the QED regime [9]. In Figure 2 we show typical sample paths of the numbers of customers in the systems, averaged over the trailing 100 time units. That is, we plot

$$\int_{t-100}^{t} \left(Q^N(u) + \sum_{i=1}^{N} 1_{\{\mathcal{B}_i^N(u) \neq 0\}} \right) du;$$

this averaging is done in order to make the plot readable by reducing the short-term variability present in both systems. The blue line corresponds to the PBR system, and the red one to the FSF system. Observe that after an initial period where the list is reorganized, the performance of the PBR system is very close to the performance of the FSF system. □

3. Main Results

3.1. **Saturated regime.** In this subsection, we assume an saturated system, that is, $\lambda^N \equiv \infty$. In that regime, an arrival occurs immediately after a service completion and, hence, all work conserving routing policies are equivalent in terms of routing decisions, since service completions occur one at a time. However, the list management algorithm results in a particular server ordering (ranking); we argue that PBR (transposition rule) achieves a desirable server ordering in terms of low $\Delta(\mathcal{L}^N)$ (compared to $\sqrt{N}$; see (1)). Let $\mathcal{L}^N = (\mathcal{L}_1^N, \ldots, \mathcal{L}_N^N)$ be a random vector with its distribution equal to the stationary distribution of $\{\mathcal{L}^N(t), t \geq 0\}$. The process $\{\mathcal{L}^N(t), t \geq 0\}$ is a reversible Markov process, and it is straightforward to verify that its stationary distribution is given, for $(l_1, \ldots, l_N) \in \mathscr{L}^N$, by

$$\mathbb{P}[\mathcal{L}^N = (l_1, \ldots, l_N)] = \frac{1}{\eta^N} \prod_{i=1}^{N} \left(\mu_{l_i}^N\right)^{-i}, \tag{2}$$

where η^N is the normalization constant:

$$\eta^N = \sum_{l \in \mathscr{L}^N} \prod_{i=1}^{N} \left(\mu_{l_i}^N\right)^{-i}.$$

Equivalently, the distribution of $\mathcal{L}^N$ satisfies, for $(l_1, \ldots, l_N) \in \mathscr{L}^N$,

$$\frac{\mathbb{P}[\mathcal{L}^N = (l_1, \ldots, l_{k+1}, l_k, \ldots, l_N)]}{\mathbb{P}[\mathcal{L}^N = (l_1, \ldots, l_k, l_{k+1}, \ldots, l_N)]} = \frac{\mu_{l_{k+1}}^N}{\mu_{l_k}^N}. \tag{3}$$

Note that, in the stationary regime, a most likely state of the process $\{\mathcal{L}^N(t), t \geq 0\}$ is $(1, 2, \ldots, N)$, namely the servers are arranged according to their service rates; the probability of this most likely list state is given by

$$\frac{1}{\eta^N} \prod_{i=1}^{N} \left(\mu_i^N\right)^{-i}.$$

The following theorem is our first result. It quantifies the quality of server ordering when PBR (transposition rule) is used. Informally, when the system operates in the QD or QED regime, the saturated regime is relevant in describing the server ordering in the subset of servers that are busy with probability 1.

Theorem 1 (Saturated regime)**.** *Consider an N-server system operating under PBR in the saturated regime ($\lambda^N \equiv \infty$). Let $\{a_N\}$ be any monotonic sequence of reals such that, as $N \to \infty$, $a_N \to \infty$. Then, as $N \to \infty$,*

$$\frac{1}{a_N \log N} \Delta(\mathcal{L}^N) \xrightarrow{\mathbb{P}} 0. \tag{4}$$

Proof. See Section 4.1. □

Remark 1. The limit in the statement of the theorem holds for any set of service rates such that $\mu \geq \mu_i^N \geq \mu_{i+1}^N$, for $i = 1, \ldots, N-1$. Clearly, if $\mu_1^N = \mu_N^N$, then the left-hand side of (4) is equal to 0 for all N (see (1)). When the service rates $\{\mu_i^N\}$ are of a specific form, more explicit bounds could be derived. For example, if $\mu = \mu_1^N = \ldots = \mu_{\lfloor N/2 \rfloor}^N$ and $\alpha\mu = \mu_{\lfloor N/2 \rfloor + 1}^N = \ldots = \mu_N^N$, for some $\alpha \in (0, 1)$ that does not change with N, then results from [8] yield, for $c > 0$,

$$\mathbb{P}\left[\Delta(\mathcal{L}^N) > c\mu\right] = \mathbb{P}\left[\sum_{j=1}^{\lfloor N/2 \rfloor} \left(\mu_j^N - \mu_{\mathcal{L}_j^N}^N\right) > c\mu\right] \leq \frac{\alpha^{c/(1-\alpha)}}{1-\alpha},$$

where the bound does not depend on N; thus, in this specific example, $\Delta(\mathcal{L}^N)/a_N \xrightarrow{\mathbb{P}} 0$, as $N \to \infty$, for any monotonic sequence $\{a_N\}$ such that $a_N \to \infty$, as $N \to \infty$. □

Example 3 (Steady-state performance). Consider a 100-server system ($N = 100$) in the saturated regime. The server rates are given by $\mu_i^N = 1 - 0.01(i-1)$, $i = 1, \ldots, N$, i.e., the server speeds decrease linearly from 1 to 0. Initially (at time $t = 0$) the list is ordered: $\mathcal{L}_i^N(0) = i$ for $i = 1, \ldots, N$. Next, for $n \geq 0$, we define

$$\sigma_n = \max_{T_{n-1} \leq t < T_n} \Delta(\mathcal{L}^N(t)),$$

where T_n is the time when the system completes $10^5 n$ service requests ($T_0 \equiv 0$). In Figure 3, we show a typical sample path of the discrete-time process $\{\sigma_n, n \geq 0\}$. The figure suggests that the list does not deviate significantly from its steady state even during long time intervals, as measured by $\Delta(\mathcal{L}^N(t))$. □

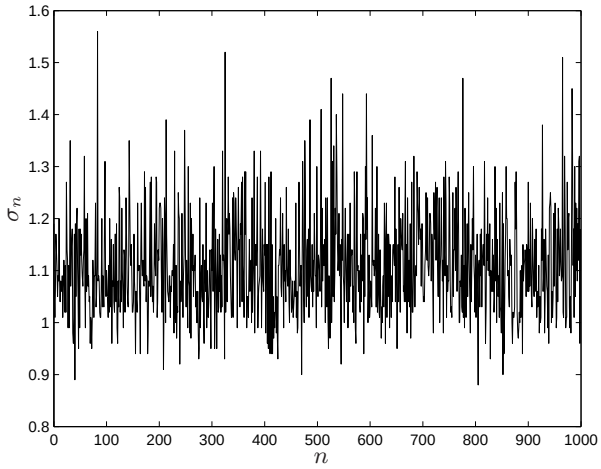

FIGURE 3. Illustration for Example 3. In terms of $\Delta(\mathcal{L}^N(t))$, the list does not deviate significantly from its steady state. Each point represents the worst (highest) $\Delta(\mathcal{L}^N(t))$ over an interval that is comparable to the chain's mixing time (see Example 4).

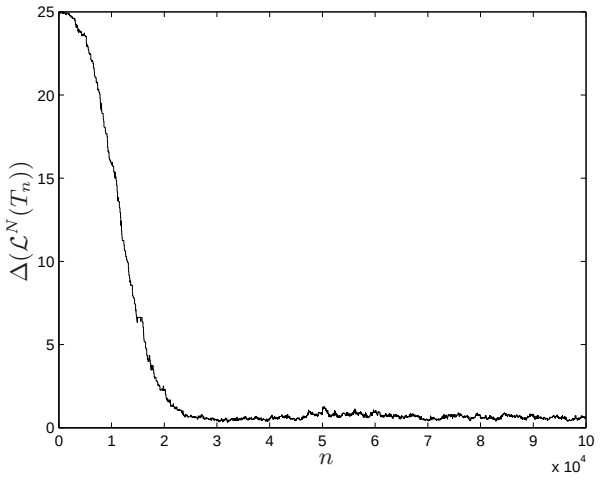

FIGURE 4. Illustration for Example 4. Initially, the list is in the reversed order. As time progresses (service completions accumulate), faster servers are moved towards the beginning of the list, resulting in lower values of $\Delta(\mathcal{L}^N(T_n))$.

Example 4 (Mixing time). Consider the system described in the previous example, with the difference that initially (at time $t = 0$) the list is in the reversed order, i.e., $\mathcal{L}_i^N(0) = 101 - i$ for $i = 1, \ldots, 100$. This represents the worst-case scenario in terms of mixing time. Redefine T_n as the time when the system completes n service requests ($T_0 \equiv 0$). In Figure 4, we plot a typical sample path of $\Delta(\mathcal{L}^N(T_n))$, as a function of n. Note that the minimum number of transpositions required for the list to become ordered is 4950 in this case. As can be seen in Figure 4, the list approaches its steady state after $25 \cdot 10^3$ transpositions. □

3.2. **Light-load regime.** In the saturated regime, there is no need to specify the routing policy in a finite ($N < \infty$) system, since at most one server becomes available at a time. However, in the case when multiple servers can be idle simultaneously, a routing algorithm

is required. We consider PBR routing: a customer is assigned to the server with the index $\min\{i : \mathcal{B}_i^N = 0\}$; that server plausibly has the highest service rate. Indeed, the higher the service rate, the lower the index of the server should be.

The following theorem characterizes the server ordering under PBR routing in the light-load regime ($\lambda \downarrow 0$). The proof is based on a time-scale decomposition. In particular, when $\lambda \downarrow 0$, customers arrive to a server in position i with rate $\Theta(\lambda^i)$. Hence, from the perspective of the server in the ith position, the list of servers in positions $1, \dots, i-1$ is in the steady state, since it operates on a faster time scale. We use this property to obtain a set of asymptotic equations for the stationary probabilities of a system with N servers. Informally, when the system operates in the QD regime, the light-load regime is relevant in describing the server ordering in the subset of servers that are idle with probability 1.

Theorem 2 (Light-load regime)**.** *Consider an N-server system operating under PBR in the light-load regime ($\lambda \downarrow 0$). Let $\{a_N\}$ be any monotonic sequence of reals such that $a_N \to \infty$, as $N \to \infty$. Then, for any $\varepsilon > 0$, we have*

$$\lim_{N\to\infty} \lim_{\lambda\downarrow 0} \mathbb{P}\left[\frac{1}{a_N \log N}\Delta(\mathcal{L}^N) > \varepsilon\right] = 0.$$

Proof. See Section 4.2. □

4. Proofs

4.1. **Proof of Theorem 1.** The proof is based on (2), namely that relative probabilities of list states are known (see (3)). We start with introducing relevant notation. For $l \in \mathscr{L}^N$, let

$$\sigma_i^N(l) = \sum_{j=1}^{i} \left(\mu_i^N - \mu_{l_j}^N\right) \quad \text{and} \quad \mu^N(l) = \prod_{j=1}^{N} \left(\mu_{l_j}^N\right)^{-j}. \tag{5}$$

Define $\mathscr{L}_{i,0}^N \subseteq \mathscr{L}^N$ to be the set of all list states (permutations) such that sum rate of the first i servers in the list is maximal:

$$\mathscr{L}_{i,0}^N = \left\{l \in \mathscr{L}^N : \sigma_i^N(l) = 0\right\};$$

the size of this set is given by

$$|\mathscr{L}_{i,0}^N| = \binom{m_i}{\hat{m}_i} i!(N-i)!,$$

where $m_i = \{\#j : \mu_j^N = \mu_i^N\}$ and $\hat{m}_i = \{\#j \le i : \mu_j^N = \mu_i^N\}$. Furthermore, for $i \in \{1, \dots, N\}$ we introduce the following additional $i \wedge (N-i)$ sets

$$\mathscr{L}_{i,k}^N = \left\{ l \in \mathscr{L}^N : \min_{h \in \mathscr{L}_{i,0}^N} \sum_{j=1}^{i} 1_{\{l_j \le i,\, h_j \le i\}} = i - k \right\},$$

where $k = 1, \dots, i \wedge (N-i)$ and $1_{\{\cdot\}}$ is the indicator function. The set $\mathscr{L}_{i,k}^N$ consists of all permutations such that, by having exactly k servers from the first i positions in the list exchange with k servers located after position i, one can obtain a permutation from $\mathscr{L}_{i,0}^N$. The above definitions imply

$$\mathscr{L}^N = \bigcup_{k=0}^{i\wedge(N-i)} \mathscr{L}_{i,k}^N. \tag{6}$$

For example, if $N = 3$ and $\mu_1 = 3$, $\mu_2 = 2$, $\mu_3 = 1$, then $\mathscr{L}_{2,0}^3 = \{(1,2,3), (2,1,3)\}$, $\mathscr{L}_{2,1}^3 = \{(1,3,2), (3,1,2), (2,3,1), (3,2,1)\}$ and $\mathscr{L}^3 = \mathscr{L}_{2,0}^3 \cup \mathscr{L}_{2,1}^3$.

Now, for $l \in \mathscr{L}^N$, let $l^{\leftarrow} \in \mathscr{L}^N$ be such that $l^{\leftarrow}_{l_i} = i$, $i = 1, \ldots, N$. For every l there exists a unique $l^{\leftarrow}$, and $l^{\leftarrow}_i$ represents the position of server i when the list is in state l. The set $\mathscr{T}_i^N(l)$, $l \in \mathscr{L}^N$, is defined as follows:

$$\mathscr{T}_i^N(l) = \left\{ h \in \mathscr{L}^N : h^{\leftarrow}_{l_j} < h^{\leftarrow}_{l_{j+1}},\ j \neq i, N \right\};$$

in words, $\mathscr{T}_i^N(l)$ contains all permutations that maintain the same relative order of the first i and the last $N-i$ servers as in l; the size of $\mathscr{T}_i^N(l)$ is

$$|\mathscr{T}_i^N(l)| = \binom{N}{i}.$$

For example, if $N = 4$ and $l = (3,2,4,1)$ then $l^{\leftarrow} = (4,2,1,3)$, $|\mathscr{T}_2^4(l)| = 6$ and $\mathscr{T}_2^4(l) = \{(3,2,4,1),\ (3,4,2,1),\ (3,4,1,2),\ (4,3,2,1),\ (4,3,1,2), (4,1,3,2)\}$. Finally, we note that

$$\mathscr{L}^N = \bigcup_{l \in \mathscr{L}^N_{i,0}} \mathscr{T}_i^N(l). \tag{7}$$

Next, we present three preliminary lemmas.

Lemma 1. *The following bound holds:*

$$\max_{l \in \mathscr{L}^N_{i,0}} |\mathscr{T}_i^N(l) \cap \mathscr{L}^N_{i,k}| \leq \binom{i}{k}\binom{N-i}{k}.$$

Remark 2. The statement of Lemma 1 holds with equality when $\mu^N_{i+1} < \mu^N_i$. □

Proof. The definitions of $\mathscr{L}^N_{i,k}$ and $\mathscr{T}_i^N(l)$ imply that, for any $l \in \mathscr{L}^N_{i,0}$,

$$\mathscr{T}_i^N(l) \cap \mathscr{L}^N_{i,k} \subseteq \mathscr{T}^N_{i,k}(l) \equiv \left\{ h \in \mathscr{T}_i^N(l) : \sum_{j=0}^{k-1} 1_{\{h^{\leftarrow}_{l_{i-j}} > i\}} = \sum_{j=1}^{k} 1_{\{h^{\leftarrow}_{l_{i+j}} \leq i\}} = k \right\},$$

and, hence, $|\mathscr{T}_i^N(l) \cap \mathscr{L}^N_{i,k}| \leq |\mathscr{T}^N_{i,k}(l)|$. The statement of the lemma follows from this inequality and

$$|\mathscr{T}^N_{i,k}(l)| = \binom{i}{k}\binom{N-i}{k};$$

the equality is due to the fact that there are $\binom{i}{k}$ ways to place servers $l_{i+1}, \ldots, l_{i+k}$ in the first i positions in the list, and there are $\binom{N-i}{k}$ ways to place servers $l_{i-k+1}, \ldots, l_i$ in the last $(N-i)$ positions in the list. □

Lemma 2. *The following bound holds:*

$$\max_{l \in \mathscr{L}^N_{i,0}} \max_{h \in \mathscr{T}_i^N(l) \cap \mathscr{L}^N_{i,k}} \frac{\mu^N(h)}{\mu^N(l)} 1_{\{\sigma_i^N(h) \geq \mu c\}} \leq e^{-ck}.$$

Proof. First, note that if $l \in \mathscr{L}^N_{i,0}$ then $\mu^N_{l_j} \geq \mu^N_{l_k}$, for any $j \leq i$ and $k > i$, and, therefore (2) implies

$$\begin{aligned} \max_{h \in \mathscr{T}_i^N(l) \cap \mathscr{L}^N_{i,k}} \frac{\mu^N(h)}{\mu^N(l)} &= \left(\frac{\mu^N_{l_{i+1}} \mu^N_{l_{i+2}} \cdots \mu^N_{l_{i+k}}}{\mu^N_{l_{i-k+1}} \mu^N_{l_{i-k+2}} \cdots \mu^N_{l_i}} \right)^k \\ &= \prod_{j=1}^{k} \left(1 - x_j c \frac{\mu}{\mu^N_{l_{i-k+j}}} \right)^k \leq \prod_{j=1}^{k} (1 - x_j c)^k, \end{aligned} \tag{8}$$

where

$$x_j \equiv \frac{1}{c\mu}\left(\mu^N_{l_{i-k+j}} - \mu^N_{l_{i+j}}\right)$$

and the last inequality is due to $\mu^N_{l_{i-k+j}} \le \mu$. Second, $l \in \mathscr{L}^N_{i,0}$ and $h \in \mathscr{T}_i^N(l) \cap \mathscr{L}^N_{i,k}$ imply

$$\begin{aligned}\sigma_i^N(h) &= \sum_{j=1}^{i}\left(\mu^N_{l_j} - \mu^N_{h_j}\right)\\ &\le \sum_{j=1}^{k}\left(\mu^N_{l_{i-k+j}} - \mu^N_{l_{i+j}}\right) = c\mu\sum_{j=1}^{k} x_j. \end{aligned} \tag{9}$$

Third, (8) and (9) yield the statement of the lemma since, for any $l \in \mathscr{L}^N_{i,0}$:

$$\begin{aligned}\max_{h\in\mathscr{T}_i^N(l)\cap\mathscr{L}^N_{i,k}} \frac{\mu^N(h)}{\mu^N(l)} 1_{\{\sigma_i^N(h)\ge c\mu\}} &\le \max_{\{y_j\ge 0\}:\sum_{j=1}^k y_j\ge 1} \prod_{j=1}^{k}\left((1-y_jc)^+\right)^k\\ &= \left((1-c/k)^+\right)^{k^2}\\ &\le e^{-ck},\end{aligned}$$

where the last inequality follows from $(1-x)^+ \le e^{-x}$, for all $x \ge 0$, and $(\cdot)^+$ denotes the positive part. □

Lemma 3. *The following bound holds:*

$$\mathbb{P}\left[\sigma_i^N(\mathcal{L}^N) \ge \mu c\right] \le \sum_{k=1}^{i\wedge(N-i)} \binom{i}{k}\binom{N-i}{k} e^{-ck}.$$

Proof. Equations (2) and (5) result in

$$\begin{aligned}\frac{\mathbb{P}\left[\sigma_i^N(\mathcal{L}^N)\ge\mu c,\, \mathcal{L}^N\in\mathscr{L}^N_{i,k}\right]}{\mathbb{P}\left[\mathcal{L}^N\in\mathscr{L}^N_{i,0}\right]} &= \frac{\sum_{l\in\mathscr{L}^N_{i,k}}\mu^N(l)1_{\{\sigma_i^N(l)\ge\mu c\}}}{\sum_{l\in\mathscr{L}^N_{i,0}}\mu^N(l)}\\ &\le \frac{\sum_{l\in\mathscr{L}^N_{i,0}}\sum_{h\in\mathscr{T}_i^N(l)\cap\mathscr{L}^N_{i,k}}\mu^N(h)1_{\{\sigma_i^N(h)\ge\mu c\}}}{\sum_{l\in\mathscr{L}^N_{i,0}}\mu^N(l)},\end{aligned}$$

where the inequality is due to (7). The preceding further implies

$$\begin{aligned}\mathbb{P}\left[\sigma_i^N(\mathcal{L}^N)\ge\mu c,\, \mathcal{L}^N\in\mathscr{L}^N_{i,k}\right] &\le \max_{l\in\mathscr{L}^N_{i,0}} \frac{\sum_{h\in\mathscr{T}_i^N(l)\cap\mathscr{L}^N_{i,k}}\mu^N(h)1_{\{\sigma_i^N(h)\ge\mu c\}}}{\mu^N(l)}\\ &\le \max_{l\in\mathscr{L}^N_{i,0}}\left\{|\mathscr{T}_i^N(l)\cap\mathscr{L}^N_{i,k}| \max_{h\in\mathscr{T}_i^N(l)\cap\mathscr{L}^N_{i,k}}\frac{\mu^N(h)}{\mu^N(l)}1_{\{\sigma_i^N(h)\ge\mu c\}}\right\}\\ &\le \binom{i}{k}\binom{N-i}{k}e^{-ck},\end{aligned} \tag{10}$$

where the last inequality follows from Lemmas 1 and 2. Equality (6) and (10) yield the statement of the lemma:

$$\mathbb{P}\left[\sigma_i^N(\mathcal{L}^N) \geq \mu c\right] \leq \sum_{k=1}^{i\wedge(N-i)} \mathbb{P}\left[\sigma_i^N(\mathcal{L}^N) \geq \mu c,\, \mathcal{L}^N \in \mathscr{L}_{i,k}^N\right]$$
$$\leq \sum_{k=1}^{i\wedge(N-i)} \binom{i}{k}\binom{N-i}{k} e^{-ck}. \qquad \square$$

Finally, we conclude this section with the proof of Theorem 1.

Proof of Theorem 1. The union bound and Lemma 3 imply

$$\mathbb{P}\left[\max_{i\in\{1,\ldots,N\}} \sigma_i^N(\mathcal{L}^N) > \varepsilon\, a_N \log N\right] \leq \sum_{i=1}^{N} \mathbb{P}\left[\sigma_i^N(\mathcal{L}^N) > \varepsilon\, a_N \log N\right]$$
$$\leq \sum_{i=1}^{N} \sum_{k=1}^{i\wedge(N-i)} \binom{i}{k}\binom{N-i}{k} e^{-\varepsilon\mu^{-1}k a_N \log N}$$
$$\leq N^2 \max_{1\leq k\leq \lfloor N/2\rfloor} N^{2k} e^{-\varepsilon\mu^{-1}k a_N \log N}$$
$$= \max_{1\leq k\leq \lfloor N/2\rfloor} N^{k(2-\varepsilon\mu^{-1}a_N)+2}, \tag{11}$$

where the last inequality is due to

$$\binom{i}{k}\binom{N-i}{k} \leq \binom{N}{2k} \leq N^{2k}.$$

The statement of the theorem follows from (11). $\square$

4.2. **Proof of Theorem 2.** The basic idea of the proof is to obtain an asymptotic relation between the stationary probabilities of different list states, similar to (2). For two real-valued functions $f(x)$ and $g(x)$, we use the notation $f(x) \sim g(x)$, as $x \downarrow 0$, to denote $f(x)/g(x) \to 1$, as $x \downarrow 0$. Next, we present two preliminary lemmas.

Lemma 4. (Time-scale decomposition) *Consider an N-server system with server speeds $\mu \geq \mu_1^N \geq \mu_2^N \geq \cdots \geq \mu_N^N \geq \delta > 0$ operating under the PBR policy. Then, for any $l = (l_1, \ldots, l_N) \in \mathscr{L}^N$, as $\lambda \downarrow 0$,*

$$\mathbb{P}\left[\mathcal{L}^N = l\right] \sim \mathbb{P}\left[\mathcal{L}_N^N = l_N\right]\, \mathbb{P}\left[\mathcal{L}^{N-1}(l_N) = (l_1, \ldots, l_{N-1})\right], \tag{12}$$

where $\mathcal{L}^{N-1}(l_N)$ is the stationary list-order vector corresponding to the system with server rates $(\mu_1^N, \ldots, \mu_{l_N-1}^N, \mu_{l_N+1}^N, \ldots, \mu_N^N)$.

Proof. Consider a time-embedded Markov chain $\{\mathcal{H}^N(n),\, n \in \mathbb{N}\}$ defined by $\mathcal{H}^N(n) = \mathcal{L}^N(t_n)$, where t_n is the time when the nth idle period (all servers are idle) starts, i.e., $t_1 = \inf\{t > 0 : \sum \mathcal{B}_i^N(t) = 0\}$ and, for $n \geq 1$,

$$t_{n+1} = \inf\left\{t > t_n : \sum_{i=1}^{N} \mathcal{B}_i^N(t) = 0,\, \sup_{t_n \leq s < t} \sum_{i=1}^{N} \mathcal{B}_i^N(s) > 0\right\};$$

let $\mathcal{H}^N$ be a random variable with the distribution equal to the stationary distribution of $\{\mathcal{H}^N(n),\, n \in \mathbb{N}\}$. Then, for all $l \in \mathscr{L}^N$,

$$\mathbb{P}[\mathcal{H}^N = l] \sim \mathbb{P}[\mathcal{L}^N = l], \tag{13}$$

as $\lambda \downarrow 0$, since the sequence of idle periods is i.i.d. with expectation equal to $1/\lambda$ and $\mathbb{P}[\sum \mathcal{B}_i^N = 0] \to 1$, as $\lambda \downarrow 0$. The transition probabilities $p^N(l,h)$, $l,h \in \mathscr{L}^N$, of the chain $\{\mathcal{H}^N(n),\, n \in \mathbb{N}\}$ satisfy $p^N(l,h) = O(\lambda^{\max\{i:\, l_i \neq h_i\}-1})$, as $\lambda \downarrow 0$; this is due to the fact that at least $\max\{i : l_i \neq h_i\}$ arrivals are needed in the original chain to occur during a single busy period (at least one server busy) in order for the transition to occur in the time-embedded chain. More precisely, as $\lambda \downarrow 0$,

$$p^N(l,h) = \Theta(\lambda^{d(l,h)-1}),$$

where $d(l,h)$ is the minimum number of arrivals required to change the state of the list from l to h. For example, $d((2,1,3,4),(1,2,4,3)) = 4$, $d((2,1,3,4),(2,3,4,1)) = 7$ and $d((1,2,3,4),(4,3,2,1)) = 14$.

In view of the preceding, $\{\mathcal{H}^N(n),\, n \in \mathbb{N}\}$ is a multi-level nearly completely decomposable Markov chain [7, Sect. 1.5], with the last level corresponding to the state of the last position in the list. Hence, (12) holds with $\mathcal{L}$ replaced with $\mathcal{H}$. Recalling (13) completes the proof. □

Lemma 5. (Stationary distribution) *Let $(l_1,\ldots,l_{N-2},i,j) \in \mathscr{L}^N$. For the stationary distribution of the list-order vector $\mathcal{L}^N$ we have, as $\lambda \downarrow 0$,*

$$\mathbb{P}\left[\mathcal{L}^N = (l_1,\ldots,l_{N-2},i,j)\right] \frac{\mu_j^N}{\sum_{k=1}^{N-2}\mu_{l_k}^N + \mu_i^N} \sim \mathbb{P}\left[\mathcal{L}^N = (l_1,\ldots,l_{N-2},j,i)\right] \frac{\mu_i^N}{\sum_{k=1}^{N-2}\mu_{l_k}^N + \mu_j^N}. \tag{14}$$

Remark 3. Lemma 4 and Lemma 5 provide a set of equations that determine the distribution of $\mathcal{L}^N$. That is, these lemmas establish a light-load analog of (3). Let $l,h \in \mathscr{L}^N$ be such that, for some $n < N$, we have $l_n = h_{n+1} < l_{n+1} = h_n$ and $l_i = h_i$, for $i \neq n, n+1$. Then, as $\lambda \downarrow 0$,

$$\mathbb{P}\left[\mathcal{L}^N = l\right] \frac{\mu_{l_{n+1}}^N}{\sum_{k=1}^{n}\mu_{l_k}^N} \sim \mathbb{P}\left[\mathcal{L}^N = h\right] \frac{\mu_{h_{n+1}}^N}{\sum_{k=1}^{n}\mu_{h_k}^N}.$$

For example, when $N = 3$, combining the two lemmas yields, for $(i,j,k) \in \mathscr{L}^N$, as $\lambda \downarrow 0$,

$$\mathbb{P}\left[\mathcal{L}^N = (i,j,k)\right] \frac{\mu_j^N}{\mu_i^N} \sim \mathbb{P}\left[\mathcal{L}^N = (j,i,k)\right] \frac{\mu_i^N}{\mu_j^N}.$$

Moreover, Lemma 4 and Lemma 5 imply that the "quality" of stationary server ordering in the limit, as $\lambda \downarrow 0$, is no worse than the ordering in the case $\lambda \to \infty$. In particular, the following inequality holds:

$$\lim_{\lambda \downarrow 0} \frac{\mathbb{P}[\mathcal{L}^N = l]}{\mathbb{P}[\mathcal{L}^N = h]} \leq \frac{\mu_{h_n}^N}{\mu_{l_n}^N}. \tag{15}$$

□

Proof. Consider the Markov chain $\{\mathcal{H}^N(n),\, n \in \mathbb{N}\}$ introduced in the proof of Lemma 4, as well as its stationary distribution. In view of (13), a time-scale decomposition (see Lemma 4) applies to this Markov chain as well. Suppose that $\{\mathcal{H}^N(n),\, n \in \mathbb{N}\}$ is in state l at some time n (recall that $\mathcal{H}^N(n) = \mathcal{L}^N(t_n)$, where t_n is the time when the nth idle period starts). The probability that the server in the last position in the list becomes busy before the next idle period starts (at time t_{n+1}) is given by, as $\lambda \downarrow 0$,

$$\prod_{k=1}^{N-1} \frac{\lambda}{\lambda + \sum_{n=1}^{k}\mu_{l_n}^N} \cdot (1 + o(\lambda)) \sim \lambda^{N-1} \prod_{k=1}^{N-1} \frac{1}{\sum_{n=1}^{k}\mu_{l_n}^N},$$

since at least $(N-1)$ arrivals are needed once the last idle period is concluded; note that

$$\frac{\lambda}{\lambda + \sum_{n=1}^{k}\mu_{l_n}^N}$$

is the probability that an arrival occurs before a service completion in one of the first k servers in the list. Consequently, given that $\mathcal{H}^N(n) = l$, the probability that $\mathcal{H}^N_N(n) \neq \mathcal{H}^N_N(n+1)$ (there is a change in the last position in the list between two idle periods) is given by

$$\prod_{k=1}^{N-1} \frac{\lambda}{\lambda + \sum_{n=1}^{k} \mu^N_{l_n}} \cdot \frac{\mu^N_{l_N}}{\mu^N_{l_{N-1}} + \mu^N_{l_N}} \left(1 + o(\lambda)\right),$$

as $\lambda \downarrow 0$; this is because the server in the last position can move one position forward if and only if the server in position $(N-1)$ does not complete service earlier. Indeed, if the server in position $(N-1)$ completes service before the server in position N, then it will remain in its position as an idle server or it will be replaced by a server ineligible for a move – in either case, the last server will remain in its position. Then, the global balance equations for the sets $\{l \in \mathscr{L}^N : l_N = i\}$ are as follows:

$$\sum_{l \in \mathscr{L}^N : l_N = i} \mathbb{P}\left[\mathcal{H}^N = l\right] \prod_{k=1}^{N-1} \frac{1}{\sum_{n=1}^{k} \mu^N_{l_n}} \frac{\mu^N_i}{\mu^N_{l_{N-1}} + \mu^N_i} \sim \sum_{l \in \mathscr{L}^N : l_{N-1} = i} \mathbb{P}\left[\mathcal{H}^N = l\right] \prod_{k=1}^{N-1} \frac{1}{\sum_{n=1}^{k} \mu^N_{l_n}} \frac{\mu^N_{l_N}}{\mu^N_{l_N} + \mu^N_i},$$

as $\lambda \downarrow 0$. By considering the indices of the servers in the last two positions in the list and applying Lemma 4, the previous equation can be rewritten in the following form:

$$\sum_{j \neq i} \sum_{l \in \mathscr{L}^N : l_{N-1} = j, l_N = i} \mathbb{P}\left[(\mathcal{H}^N_1, \ldots, \mathcal{H}^N_{N-2}) = (l_1, \ldots, l_{N-2})\right] \frac{1}{\mu^N_i + \mu^N_j} \prod_{k=1}^{N-2} \frac{1}{\sum_{n=1}^{k} \mu^N_{l_n}} \times$$
$$\times \left(\mathbb{P}[(\mathcal{H}^N_{N-1}, \mathcal{H}^N_N) = (j,i)] \frac{\mu^N_i}{\sum_{k=1}^{N-2} \mu^N_{l_k} + \mu^N_j} - \mathbb{P}[(\mathcal{H}^N_{N-1}, \mathcal{H}^N_N) = (i,j)] \frac{\mu^N_j}{\sum_{k=1}^{N-2} \mu^N_{l_k} + \mu^N_j} \right) \sim 0, \tag{16}$$

as $\lambda \downarrow 0$. Now, assume that, as $\lambda \downarrow 0$,

$$\mathbb{P}\left[\mathcal{H}^N = (l_1, \ldots, l_{N-2}, i, j)\right] \frac{\mu^N_j}{\sum_{k=1}^{N-2} \mu^N_{l_k} + \mu^N_i} \sim \mathbb{P}\left[\mathcal{H}^N = (l_1, \ldots, l_{N-2}, j, i)\right] \frac{\mu^N_i}{\sum_{k=1}^{N-2} \mu^N_{l_k} + \mu^N_j}, \tag{17}$$

for $(l_1, \ldots, l_{N-2}, i, j) \in \mathscr{L}^N$ and $N \leq K-1$, for some K. Then, (17) also holds for $N = K$. Indeed, in view of Lemma 4 and the inductive assumption, (17) defines a set of probabilities for $\mathcal{L}^K$ that solve (16) (the terms in parentheses in (16) vanish, as $\lambda \downarrow 0$). Relations (13) and (17) yield the statement of the lemma. □

Finally, we present the proof of Theorem 2.

Proof of Theorem 2. It is sufficient to consider the case $\mu^N_N > 0$ since, otherwise, servers with $\mu^N_i = 0$ eventually end up at the end of the list.

The proof is very similar to the proof of Theorem 1; it will thus help to recall the definitions introduced there. For $l \in \mathscr{L}^N_{i,0}$, as in the proof of Lemma 2, we have

$$\lim_{\lambda \downarrow 0} \max_{h \in \mathscr{T}^N_i(l) \cap \mathscr{L}^N_{i,k}} \frac{\mathbb{P}[\mathcal{L}^N = h]}{\mathbb{P}[\mathcal{L}^N = l]} \leq \left(\frac{\mu^N_{l_{i+1}} \mu^N_{l_{i+2}} \cdots \mu^N_{l_{i+k}}}{\mu^N_{l_{i-k+1}} \mu^N_{l_{i-k+2}} \cdots \mu^N_{l_i}} \right)^k,$$

where the inequality is due to Lemma 5 (see (15)). Therefore, a statement analogous to the statement of Lemma 2 holds:

$$\lim_{\lambda\downarrow 0} \max_{l\in\mathscr{L}^N_{i,0}} \max_{h\in\mathscr{T}^N_i(l)\cap\mathscr{L}^N_{i,k}} \frac{\mathbb{P}[\mathcal{L}^N=h]}{\mathbb{P}[\mathcal{L}^N=l]} 1_{\{\sigma^N_i(h)\geq\mu c\}} \leq e^{-ck}.$$

The rest of the proof can be obtained by following the same steps as in the proof of Theorem 1. □

Acknowledgments

The work of A.M. has been partially supported by BSF Grants 2005175 and 2008480, ISF Grant 1357/08 and by the Technion funds for promotion of research and sponsored research. Some of the research was funded by and carried out while A.M. was visiting the Statistics and Applied Mathematical Sciences Institute (SAMSI) of the NSF; the Department of Statistics and Operations Research (STOR), the University of North Carolina at Chapel Hill; the Department of Information, Operations and Management Sciences (IOMS), Leonard N. Stern School of Business, New York University; and the Department of Statistics, The Wharton School, University of Pennsylvania – the wonderful hospitality of all four institutions is gratefully acknowledged and truly appreciated.

The work of P. M. was supported in part by NSF Grant CNS-0643213.

References

[1] D. Aldous. Some interesting processes arising as heavy traffic limits in an M/M/∞ storage process. *Stochastic Process. Appl.*, 22(2):291–313, 1986. 1.3

[2] M. Armony. Dynamic routing in large-scale service systems with heterogeneous servers. *Queueing Syst. Theory Appl.*, 51(3-4):287–329, 2005. 1.1, 1.3

[3] R. Atar. Central limit theorem for a many-server queue with random service rates. *Ann. Appl. Probab.*, 18(4):1548–1568, 2008. 1.3

[4] R. Atar and A. Shwartz. Efficient routing in heavy traffic under partial sampling of service times. *Math. Oper. Res.*, 33(4):899–909, 2008. 1.1, 1.3

[5] E. Coffman, T. Kadota, and L. Shepp. Stochastic model of fragmentation in dynamic storage allocation. *SIAM J. Comput.*, 14(2):416–425, 1985. 1.3

[6] T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2nd edition, 2001. 1.2

[7] P. J. Courtois. *Decomposability: Queueing and Computer System Applications*. Academic Press, 1977. 4.2

[8] D. Gamarnik and P. Momčilović. A transposition rule analysis based on a particle process. *J. Appl. Probab.*, 42(1):234–246, 2005. 1

[9] S. Halfin and W. Whitt. Heavy-traffic limits for queues with many exponential servers. *Oper. Res.*, 29(3):567–588, 1981. 2

[10] D. Knuth. *The Art of Computer Programming*, volume 3. Sorting and Searching. Addison-Wesley, 2nd edition, 1998. 2

[11] A. Mandelbaum and A. Stolyar. Scheduling flexible servers with convex delay costs: Heavy-traffic optimality of the generalized $c\mu$-rule. *Oper. Res.*, 52(6):836–855, 2004. 1.1

[12] J. Van Mieghem. Dynamic scheduling with convex delay costs: The generalized $c\mu$ rule. *Ann. Appl. Probab.*, 5(3):809–833, 1995. 1.1

[13] G. F. Newell. *The M/M/∞ service system with ranked servers in heavy traffic*, volume 231 of *Lecture Notes in Econ. and Math. Systems*. Springer-Verlag, New York, NY, 1984. 1.3

[14] J.G. Shanthikumar and D. Yao. Comparing ordered-entry queues with heteroneneous servers. *Queueing Syst. Theory Appl.*, 2(3):235–244, 1987. 1.1, 1.2, 1.3

Faculty of Industrial Engineering and Management, Technion, Haifa 3200, Israel
E-mail address: avim@tx.technion.ac.il

Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611, U.S.A.
E-mail address: petar@ise.ufl.edu