Operations Research Letters 42 (2014) 418–423

Contents lists available at ScienceDirect

Operations Research Letters

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

Performance-based routing

Avishai Mandelbaum [a], Petar Momčilović [b,*]

[a] Faculty of Industrial Engineering and Management, Technion, Haifa 3200, Israel

[b] Department of Industrial and Systems Engineering, University of Florida, Gainesville, FL 32611, USA

ARTICLE INFO

Article history:
Received 18 February 2014
Received in revised form
10 July 2014
Accepted 11 July 2014
Available online 17 July 2014

Keywords:
Multi-server queue
Routing
Transposition rule

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 FSF routing is not straightforward. We analyze an algorithm that approximates FSF routing: 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.

1. Introduction

Motivation. The Fastest-Server-First (FSF) routing policy [1] 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 [10]): 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 [3]; 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$c\mu$ [8,7]) 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.

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 loss of generality, $\mu_1^N \geq \mu_2^N \geq \cdots \geq \mu_N^N$. Service rates are bounded ($\mu_i^N \leq \mu$, where $\mu < \infty$ does not change with N) and do not depend on the routing policy (we do not model systems where servers can reduce their service rate in order to reduce the number of customers they serve). 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. 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 $N \to \infty$. The considered system is related to rank-based systems [10]. 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

* Corresponding author.
E-mail addresses: momcilovic@ufl.edu, petar@ise.ufl.edu (P. Momčilović).

http://dx.doi.org/10.1016/j.orl.2014.07.003
0167-6377/

(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 $\mu_i^N = 3$ for $1 \le i \le \lceil N/3 \rceil$, $\mu_i^N = 2$ for $\lceil N/3 \rceil < i \le \lceil 2N/3 \rceil$, $\mu_i^N = 1$ for $\lceil 2N/3 \rceil < i \le N$, 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

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

Note that $\Delta(l) \ge 0$ ($\sum_{j=1}^{i} \mu_j^N \ge \sum_{j=1}^{i} \mu_{l_j}^N$, $i = 1, \ldots, N$), 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.

Brief literature review. Rank-based routing policies, defined by fixed vectors, were studied in [10]. 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 $M/M/\infty$ storage process [9]: 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 [2]. The author established a central limit theorem for the number of customers in the system when the system is in the QED regime. An asymptotic optimality (in the QED regime) of the FSF policy was shown in [1]. In [3], 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 $i \ge 2$ 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 [6, Section 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. (A service completion by a server while the adjacent server is idle does not convey any information about their relative rates.)
- *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 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, $\mathcal{L}^N(t) = (\mathcal{L}_1^N(t), \ldots, \mathcal{L}_N^N(t)) = (l_1, \ldots, l_N)$ 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 $\mathcal{B}_i^N(t) = 1$, then the server in the ith position in the list is busy and eligible for a move at time t; when $\mathcal{B}_i^N(t) = 0$, the server is idle; when $\mathcal{B}_i^N(t) = 2$, the server is busy, but ineligible