J. Appl. Prob. **53,** 1111–1124 (2016)
*doi:*10.1017/jpr.2016.68

UNIVERSALITY OF LOAD BALANCING SCHEMES ON THE DIFFUSION SCALE

DEBANKUR MUKHERJEE,* ** *Eindhoven University of Technology*
SEM C. BORST,* *Eindhoven University of Technology and Nokia Bell Labs*
JOHAN S. H. VAN LEEUWAARDEN,* *Eindhoven University of Technology*
PHILIP A. WHITING,*** *Macquarie University*

Abstract

We consider a system of N parallel queues with identical exponential service rates and a single dispatcher where tasks arrive as a Poisson process. When a task arrives, the dispatcher always assigns it to an idle server, if there is any, and to a server with the shortest queue among d randomly selected servers otherwise ($1 \leq d \leq N$). This load balancing scheme subsumes the so-called join-the-idle queue policy ($d = 1$) and the celebrated join-the-shortest queue policy ($d = N$) as two crucial special cases. We develop a stochastic coupling construction to obtain the diffusion limit of the queue process in the Halfin–Whitt heavy-traffic regime, and establish that it does not depend on the value of d, implying that assigning tasks to idle servers is sufficient for diffusion level optimality.

Keywords: Join-the-idle-queue policy; join-the-shortest-queue policy; load balancing; power of two; routeing; sample path comparison; stochastic coupling; supermarket model

2010 Mathematics Subject Classification: Primary 60K25; 68M20
Secondary 60K30; 90B18; 90B36

1. Introduction

In this paper we establish a universality property for a broad class of load balancing schemes in a many-server heavy-traffic regime. While the specific features of load balancing policies may considerably differ, the principal purpose is to distribute service requests or tasks among servers or geographically distributed nodes in parallel-processing systems. Well-designed load balancing schemes provide an effective mechanism for improving relevant performance metrics experienced by users while achieving high resource utilization levels. The analysis and design of load balancing policies has attracted strong renewed interest in the last several years, mainly motivated by significant challenges involved in assigning tasks (e.g. file transfers, compute jobs, data base look-ups) to servers in large-scale data centers.

Load balancing schemes can be broadly categorized as static (open loop), dynamic (closed loop), or some intermediate blend, depending on the amount of real-time feedback or state information (e.g. queue lengths or load measurements) that is used in assigning tasks. Within the category of dynamic policies, we can further distinguish between push-based and pull-based approaches, depending on whether the initiative resides with a dispatcher actively collecting feedback from the servers, or with the servers advertizing their availability or load status.

Received 15 October 2015; revision received 29 February 2016.
* Postal address: Eindhoven University of Technology, PO Box 513, 5600 MB Eindhoven, Netherlands.
** Email address: d.mukherjee@tue.nl
*** Postal address: Department of Engineering, Macquarie University, North Ryde, NSW 2109, Australia.

The use of state information naturally allows dynamic policies to achieve better performance and greater resource pooling gains, but also involves higher implementation complexity and potentially substantial communication overhead. The latter issue is particularly pertinent in large-scale data centers, which deploy thousands of servers and handle massive demands, with service requests coming in at huge rates.

In the present paper we focus on a basic scenario of N parallel queues with identical servers, exponentially distributed service requirements, and a service discipline at each individual server that is oblivious to the actual service requirements (e.g. first-come–first-served). In this canonical case, the so-called join-the-shortest-queue (JSQ) policy has several strong optimality properties, and, in particular, minimizes the overall mean delay among the class of nonanticipating load balancing policies that do not have any advance knowledge of the service requirements [3], [16], [18]. (Relaxing any of the three abovementioned assumptions tends to break the optimality properties of the JSQ policy, and renders the delay-minimizing policy quite complex or even counterintuitive; see, for instance, [5], [7], and [17].)

In order to implement the JSQ policy, a dispatcher requires instantaneous knowledge of the queue lengths at all the servers, which may give rise to a substantial communication burden, and may not be scalable in scenarios with large numbers of servers. The latter issue has motivated consideration of so-called JSQ(d) policies, where the dispatcher assigns an incoming task to a server with the shortest queue among d servers selected uniformly at random. Mean-field limit theorems in [9] and [15] indicate that even a value as small as $d = 2$ yields significant performance improvements in a many-server regime, in the sense that the tail of the queue length distribution at each individual server falls off much more rapidly compared to a strictly random assignment policy ($d = 1$). This is commonly referred to as the 'power-of-two' effect. While these results were originally proved for exponential service requirement distributions, they have recently been extended to general service requirement distributions in [2].

In this paper we consider a related but different family of load balancing schemes termed JIQ(d), where the dispatcher always assigns an incoming task to an idle server, if there is any, and to a server with the shortest queue among d uniformly at random selected servers otherwise. Observe that the JIQ(N) scheme coincides with the ordinary JSQ policy, while the JIQ(1) scheme corresponds to the so-called join-the-idle-queue (JIQ) policy considered in [1], [8], and [12]. The latter policy offers particularly attractive properties, both from a scalability perspective and from a performance viewpoint. Since only knowledge of the empty queues is required, it suffices for servers to send an 'invite' notice to the dispatcher whenever they become idle. This generates at most one message per task and ensures low communication overhead even in large-scale systems with many servers. At the same time, fluid-limit theorems in [12] indicate that, for any fixed subcritical load per server, the equilibrium probability of a task experiencing a wait because no idle server is available, asymptotically vanishes in a regime where the number of servers grows large.

We consider a regime where the number of servers N grows large, but additionally assume that the capacity slack per server diminishes as $\beta/\sqrt{N}$, i.e. the load per server approaches unity as $1 - \beta/\sqrt{N}$, with $\beta > 0$ some positive coefficient. In terms of the aggregate traffic load and total service capacity, this scaling corresponds to the so-called Halfin–Whitt heavy-traffic regime which was introduced in the seminal paper [6] and has been extensively studied since. The setup in [6], as well as the numerous model extensions in the literature, predominantly concerned a setting with a single centralized queue and server pool, rather than a scenario with parallel queues. To the best of our knowledge, the only exception is a recent study of Eschenfeldt and Gamarnik [4], who considered a parallel-server system with the ordinary JSQ

https://doi.org/10.1017/jpr.2016.68 Published online by Cambridge University Press

policy, and showed that in the Halfin–Whitt regime the diffusion-scaled system occupancy state weakly converges to a two-dimensional reflected Ornstein–Uhlenbeck process.

We exploit a stochastic coupling construction to extend the latter result to the entire class of JIQ(d) policies. We specifically establish that the diffusion limit, rather surprisingly, does not depend on the value of d at all, so that in particular the JIQ and JSQ policies yield the same diffusion limit. The latter property implies that in a many-server heavy-traffic regime, ensuring that tasks are assigned to idle servers whenever possible, e.g. using a low-overhead invite mechanism, suffices to achieve optimality at the diffusion level, and not just at the fluid level as proved by Stolyar [12] for the under-loaded scenario. It further suggests that using any additional queue length information beyond the knowledge of empty queues yields only limited performance gains in large-scale systems in the Halfin–Whitt heavy-traffic regime.

The remainder of the paper is organized as follows. In Section 2 we present a detailed model description and formulate the main result. In Section 3 we develop a stochastic coupling construction to compare the system occupancy state under various task assignment policies. We then combine in Section 4 the stochastic comparison results with some of the derivations in [4] to obtain the common diffusion limit and finally make a few concluding remarks in Section 5.

2. Model description

Consider a system with N parallel queues with independent and identical servers having unit-exponential service rates and a single *dispatcher*. Tasks arrive at the dispatcher as a Poisson process of rate λ_N, and are instantaneously forwarded to one of the servers. Tasks can be queued at the various servers, possibly subject to a buffer capacity limit as further described below, but *cannot* be queued at the dispatcher. The dispatcher always assigns an incoming task to an idle server, if there is any, and to a server with the shortest queue among d uniformly at random selected servers otherwise ($1 \leq d \leq N$), ties being broken arbitrarily. The buffer capacity at each of the servers is $b \geq 2$ (possibly infinite), and when a task is assigned to a server with b pending tasks, it is instantly discarded.

As mentioned earlier, the above-described scheme coincides with the ordinary JSQ policy when $d = N$, and corresponds to the JIQ policy considered in [1], [8], and [12] when $d = 1$.

Under the JSQ policy, the dispatcher always assigns an incoming task to the server with the minimum queue length. As stated in the introduction, the JSQ policy has several strong optimality properties in the symmetric Markovian scenario under consideration. In order to implement the JSQ policy however, a dispatcher requires instantaneous knowledge of the queue lengths at all the servers, which may give rise to a substantial communication burden, and may not be scalable in scenarios with large numbers of servers. In a recent study Eschenfeldt and Gamarnik [4] characterized the diffusion limit of the system occupancy state in the Halfin–Whitt heavy-traffic regime.

Under the JIQ policy, the dispatcher assigns an incoming task to an idle server, if there is any, or to a uniformly at random selected server otherwise. This scheme is of particular interest because of its low communication overhead, and can be implemented as follows. When a server becomes idle, it sends an *invite* message to the dispatcher declaring that it is vacant. Whenever a task arrives, the dispatcher looks at its list of invite messages. If there are any messages in the list then it selects one arbitrarily, assigns the task to the corresponding server, and discards the selected invite message. Otherwise, the dispatcher assigns the task uniformly at random to one of the servers. In this way the number of messages exchanged per task is at most 1, reducing the communication overhead and ensuring scalability. Stolyar [12] recently proved that the probability that there are invite messages approaches 1, and, hence, the fraction of tasks that

https://doi.org/10.1017/jpr.2016.68 Published online by Cambridge University Press

incur a nonzero wait tends to 0, in a fluid regime where the number of servers and total arrival rate grow large in proportion with $\lambda_N/N \to \lambda < 1$ as $N \to \infty$.

In the present paper we consider the Halfin–Whitt heavy-traffic regime where the arrival rate increases with the number of servers as $\lambda_N = N - \beta\sqrt{N}$ for some $\beta > 0$. We denote the class of above-described policies by $\Pi^{(N)}(d)$, where the superscript N indicates that the diversity parameter d is allowed to depend on the number of servers. For any policy $\Pi \in \Pi^{(N)}(d)$ and buffer size b, let $\boldsymbol{Q}^\Pi = (Q_1^\Pi, Q_2^\Pi, \ldots, Q_b^\Pi)$, where Q_i^Π is the number of servers with a queue length greater than or equal to $i = 1, \ldots, b$, including the possible task in service. Also, let $\boldsymbol{X}^\Pi = (X_1^\Pi, X_2^\Pi, \ldots, X_b^\Pi)$ be a properly centered and scaled version of the vector $\boldsymbol{Q}^\Pi$, with $X_1^\Pi = (Q_1^\Pi - N)/\sqrt{N}$ and $X_i^\Pi = Q_i^\Pi/\sqrt{N}$ for $i = 2, \ldots, b$. The reason why Q_1^Π is centered around N while Q_i^Π, $i = 2, \ldots, b$, are not is because the fraction of servers with exactly one task tends to 1 as N grows large, as we will see. In the case of a finite buffer size $b < \infty$, when a task is discarded, we call it an *overflow* event, and we denote by $L^\Pi(t)$ the total number of overflow events under policy Π up to time t.

The next theorem states our main result. In the rest of the paper let D be the set of all right-continuous functions from $[0, \infty)$ to $\mathbb{R}$ having left limits, and let '$\xrightarrow{\mathrm{D}}$' denote convergence in distribution.

Theorem 1. *For any policy* $\Pi \in \Pi^{(N)}(d)$, *if, for* $i = 1, 2, \ldots, X_i^\Pi(0) \xrightarrow{\mathrm{D}} X_i(0)$ *in* $\mathbb{R}$ *as* $N \to \infty$ *with* $X_i(0) = 0$ *for* $i \geq 3$, *then the processes* $\{X_i^\Pi(t)\}_{t\geq 0} \xrightarrow{\mathrm{D}} \{X_i(t)\}_{t\geq 0}$ *in* D, *where* $X_i(t) \equiv 0$ *for* $i \geq 3$ *and* $(X_1(t), X_2(t))$ *are unique solutions in* $D \times D$ *of the stochastic integral equations*

$$
\begin{aligned}
X_1(t) &= X_1(0) + \sqrt{2}W(t) - \beta t + \int_0^t (-X_1(s) + X_2(s))\,\mathrm{d}s - U_1(t),\\
X_2(t) &= X_2(0) + U_1(t) + \int_0^t (-X_2(s))\,\mathrm{d}s,
\end{aligned}
\tag{1}
$$

where W *is a standard Brownian motion and* U_1 *is the unique nondecreasing nonnegative process in* D *satisfying* $\int_0^\infty \mathbf{1}[X_1(t) < 0]\,\mathrm{d}U_1(t) = 0$.

The above result is proved in [4] for the ordinary JSQ policy. Our contribution is to develop a stochastic ordering construction and establish that, somewhat remarkably, the diffusion limit is the same for any policy in $\Pi^{(N)}(d)$. In particular, the JIQ and JSQ policies yield the same diffusion limit. The latter property implies that in the Halfin–Whitt heavy-traffic regime, assigning tasks to idle servers, e.g. through a lightweight invite mechanism, suffices to achieve optimality at the diffusion level. It further suggests that using any additional queue length information beyond the knowledge of empty queues yields only limited performance gains in large-scale systems in the Halfin–Whitt heavy-traffic regime.

Remark 1. Note that, as in [4], we assume the convergence of the initial state, which implies that the process has to start from a state in which the number of vacant servers as well as the number of servers with two tasks scale with $\sqrt{N}$, and the number of servers with three or more tasks is $o(\sqrt{N})$.

3. Coupling and stochastic ordering

In this section we prove several stochastic comparison results for the system occupancy state under various load balancing schemes for a fixed number of queues N (and, hence, we shall

https://doi.org/10.1017/jpr.2016.68 Published online by Cambridge University Press

often omit the superscript N in this section). These stochastic ordering results will be leveraged in the next section to prove the main result stated in Theorem 1.

In order to bring out the full strength of the stochastic comparison results, we will in fact consider a broader class of load balancing schemes $\Pi^{(N)} := \{\Pi(d_0, d_1, \ldots, d_{b-1})\colon d_0 = N,\ 1 \le d_i \le N,\ 1 \le i \le b-1,\ b \ge 2\}$, and show that Theorem 1 actually holds for this entire class. In the scheme $\Pi(d_0, d_1, \ldots, d_{b-1})$, the dispatcher assigns an incoming task to the server with the minimum queue length among d_k (possibly a function of N) servers selected uniformly at random when the minimum queue length across the system is k, $k = 0, 1, \ldots, b-1$. As before, b represents the buffer size, and when a task is assigned to a server with b outstanding tasks, it is instantly discarded.

3.1. Stack formation and deterministic ordering

Let us consider the servers arranged in nondecreasing order of their queue lengths. Each server along with its queue can be thought of as a stack of items. The ensemble of stacks then represent the empirical cumulative distribution function (CDF) of the queue length distribution, and the ith horizontal bar corresponds to Q_i^{Π} (for the concerned policy Π). The items are added to and removed from the various stacks according to some rule. Before proceeding to the coupling argument, we first state and prove a deterministic comparison result under the above setting.

Consider two ensembles A and B with the same total number of stacks. The stacks in ensemble A have a maximum capacity of b items and those in ensemble B have a maximum capacity of b' items with $b \le b'$. For two such ensembles, a step is said to follow $\mathrm{Rule}(k, l, l_A, l_B)$ if either addition or removal of an item in both ensembles is done in that step as follows.

(i) *Removal.*

An item is removed (if any) from the kth stack from both ensembles or an item is removed from some stack in ensemble A but no removal is done in ensemble B.

(ii) *Addition.*

(ii.a) *System A.* If the minimum stack height is less than $b-1$ then the item is added to the lth stack. Otherwise, the item is added to the l_Ath stack. If the item lands on a stack with height b then it is dropped.

(ii.b) *System B.* If the minimum stack height is less than $b-1$ then the item is added to the lth stack. Otherwise, if the minimum stack height is precisely equal to $b-1$, the item is added to the l_Bth stack. When the minimum stack height in the system is at least b the item can be sent to any stack. If the item lands on a stack with height b' then it is dropped.

Then we have the following result.

Proposition 1. *Consider two ensembles A and B as described above with the total number of stacks being N, stack capacities respectively being b and b', $b \le b'$, and with $\boldsymbol{Q}^A \le \boldsymbol{Q}^B$ componentwise, i.e. $Q_i^A \le Q_i^B$ for all $i \ge 1$. The componentwise ordering is preserved if at any step* $\mathrm{Rule}(k, l, l_A, l_B)$ *is followed with $l_A \ge l_B$ and either $l = 1$ or $l \ge l_B$.*

Before diving deeper into the proof of this proposition, let us discuss the high-level intuition behind it. First observe that if $\boldsymbol{Q}^A \le \boldsymbol{Q}^B$, and an item is added (removed) to (from) the stack with the same index in both ensembles, then the componentwise ordering will be preserved.

https://doi.org/10.1017/jpr.2016.68 Published online by Cambridge University Press

Hence, the preservation of ordering at the time of removal, and at the time of addition when, in both ensembles, the minimum stack height is less than $b-1$, is fairly straightforward.

Now, in other cases of addition, since in ensemble A the stack capacity is b $(\leq b')$, if the minimum stack height in ensemble B is at least b, the ordering is preserved trivially. This leaves us with only the case when the minimum stack height in ensemble B is precisely equal to $b-1$. In this case, when the minimum stack height in ensemble A is also precisely equal to $b-1$, the preservation of the ordering follows from the assumption that $l_A \geq l_B$, which ensures that if in ensemble A the item is added to some stack with $b-1$ items (and, hence, increases Q_b^A), then the same will be done in ensemble B whenever $Q_b^A = Q_b^B$. Otherwise, if the minimum stack height in ensemble A is less than b, then assuming that either $l=1$ (i.e. the item will be sent to the minimum queue) or $l \geq l_B$ (i.e. an increase in Q_b^A implies an increase in Q_b^B) ensures the preservation of ordering.

Proof of Proposition 1. Suppose that after following $\mathrm{Rule}(k, l, l_A, l_B)$ the updated lengths of the horizontal bars of ensemble Π are denoted by $(\tilde{Q}_1^\Pi, \tilde{Q}_2^\Pi, \ldots)$, $\Pi = A, B$. We need to show that $\tilde{Q}_i^A \leq \tilde{Q}_i^B$ for all $i \geq 1$.

For ensemble Π, let us define $I_\Pi(c) := \max\{i : Q_i^\Pi \geq N-c+1\}$, $c = 1, \ldots, N$, $\Pi = A, B$. Define $I_\Pi(c)$ to be 0 if Q_1^Π is (and, hence, all the Q_i^Π values are) less than $N-c+1$. Note that $I_A(c) \leq I_B(c)$ for all $c = 1, 2, \ldots, N$ because of the initial ordering.

Now if the rule produces a removal of an item then the updated ensemble will have the values

$$\tilde{Q}_i^\Pi = \begin{cases} Q_i^\Pi - 1 & \text{for } i = I_\Pi(k), \\ Q_i^\Pi & \text{otherwise}, \end{cases}$$

if $I_\Pi(k) \geq 1$; otherwise, all the Q_i^Π values remain unchanged. For example, in Figure 1, $b = 5$, $N = 10$, and at the time of removal $k = 7$. For this configuration, $I_\Pi(7) = 4$ since $Q_4^\Pi = 5 \geq 10-7+1 = 4$, but $Q_5^\Pi = 2 < 4$. Hence, Q_4^Π is reduced and all the other values remain unchanged. Note that the specific label of the servers does not matter here. So after the removal/addition of an item we consider the configuration as a whole by rearranging it again in nondecreasing order of the queue lengths.

Since in both A and B the values of Q_i remain unchanged except for $i = I_A(k)$ and $I_B(k)$, it suffices to prove the preservation of the ordering for these two specific values of i. Now, for $i = I_A(k)$,

$$\tilde{Q}_i^A = Q_i^A - 1 \leq Q_i^B - 1 \leq \tilde{Q}_i^B.$$

If $I_B(k) = I_A(k)$ then we are done by the previous step. If $I_B(k) > I_A(k)$ then, from the definition of $I_A(k)$, observe that $I_B(k) \notin \{i : Q_i^A \geq N-k+1\}$ and, hence, $Q_i^A < N-k+1$

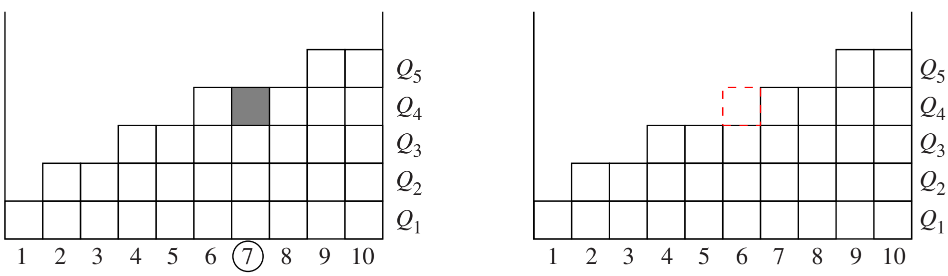

FIGURE 1: Removal of an item from the ensemble.

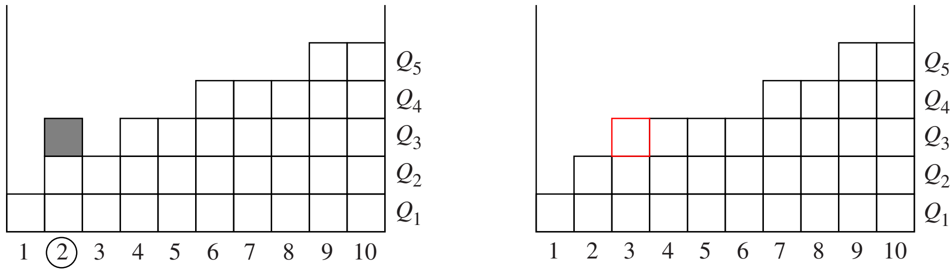

FIGURE 2: Addition of an item to the ensemble.

for $i = I_B(k)$. Therefore, for $i = I_B(k)$,

$$\tilde{Q}_i^A \le N - k \le Q_i^B - 1 = \tilde{Q}_i^B.$$

On the other hand, if the rule produces the addition of an item to stack l then the values will be updated as

$$\tilde{Q}_i^\Pi = \begin{cases} Q_i^\Pi + 1 & \text{for } i = I_\Pi(l) + 1, \\ Q_i^\Pi & \text{otherwise,} \end{cases}$$

if $I_\Pi(l) < b_\Pi$, with b_Π the stack capacity of the corresponding system; otherwise, the values remain unchanged. In Figure 2 we have $l = 2$ and, for that particular configuration, $I_\Pi(2) = 2$. Hence, Q_3^Π is incremented by one and the other variables remain fixed.

It is therefore enough to consider the ith horizontal bars for $i = (I_A(l) + 1), (I_B(l) + 1)$ when $I_A(l) < b$. According to the addition rule there are several cases which we now consider one by one.

- First we consider the case when in both ensembles the minimum stack height is less than $b-1$. Then, by part (ii) of the rule, both incoming items are added to the lth stack. When considering ensemble B, we may neglect the case in which $I_B(l) \ge b$ since then the value at $I_B(l)+1$ does not matter. Thus, assume that $I_B(l) \le b-1$ and set $i = I_B(l)+1$ so that
$$\tilde{Q}_i^B = Q_i^B + 1 \ge Q_i^A + 1 \ge \tilde{Q}_i^A.$$
If $I_A(l) = I_B(l)$ then we are done by the previous case. If $I_A(l) + 1 \le I_B(l)$ then it follows from the definition that $Q_i^A < N - l + 1$ and $Q_i^B \ge N - l + 1$ for $i = I_A(l) + 1$. Hence,
$$\tilde{Q}_i^A = Q_i^A + 1 \le N - l + 1 \le Q_i^B \le \tilde{Q}_i^B.$$
- If the minimum stack height in A is less than $b - 1$ and that in B is precisely $b - 1$, then according to the rule, the incoming item is added to the lth stack in A and the l_Bth stack in B. We show here that the componentwise ordering will be preserved if either $l = 1$ or $l \ge l_B$. Observe that if $l = 1$ then $I_A(l) < b - 1$, which implies that $I_A(l) + 1 \le b - 1$. But, since the minimum stack height in B is $b - 1$ for all $i \le b - 1$ and, in particular, for $i = I_A(l) + 1$, $\tilde{Q}_i^B = N \ge \tilde{Q}_i^A$. Now we consider the case when $l \ge l_B$. Also, observe that the fact that the minimum stack height in B is $b - 1$ implies that $I_B(l_B) \ge b-1 \ge I_A(l_A)$ (since if $I_A(l) = b$ then nothing will be changed and so we do not need to consider this case). Then again if $I_A(l) = I_B(l_B)$, we are done. Therefore, suppose that $I_A(l) < I_B(l_B)$, which implies that $I_A(l) + 1 \le I_B(l_B)$. By definition, for

https://doi.org/10.1017/jpr.2016.68 Published online by Cambridge University Press

$i = I_A(l)+1$, we have $Q_i^A < N-l+1$ and $Q_i^B \geq N-l_B+1 \geq N-l+1$. Combining these two inequalities yields

$$\tilde{Q}_i^A = Q_i^A + 1 \leq N - l + 1 \leq Q_i^B = \tilde{Q}_i^B.$$

- If the minimum stack height in both ensembles is $b-1$ then recall that the incoming item is added to the l_Ath stack in A and to the l_Bth stack in B with $l_A \geq l_B$. Arguing similarly as in the previous case, we can conclude that the inequality is preserved.

- Finally, if the minimum stack height in B is larger than or equal to b, then the preservation of the inequality is trivial.

Hence, the proof of the proposition is complete. □

3.2. The coupling construction

We now construct a coupling between two systems A and B following any two schemes, say, $\Pi_A = \Pi(l_0, l_1, \dots, l_{b-1})$ and $\Pi_B = \Pi(d_0, d_1, \dots, d_{b'-1})$ in $\Pi^{(N)}$, respectively, and combine it with Proposition 1 to get the desired stochastic ordering results.

For the arrival process, we couple the two systems as follows. First we synchronize the arrival epochs of the two systems. Now assume that in systems A and B the minimum queue lengths are k and m, respectively, $k \leq b-1$ and $m \leq b'-1$. Therefore, when a task arrives, the dispatchers in A and B have to select l_k and d_m servers, respectively, and then have to send the task to the one having the minimum queue length among the respectively selected servers. Since the servers are being selected uniformly at random, we can assume, without loss of generality, as in the stack construction, that the servers are arranged in nondecreasing order of their queue lengths and are indexed in increasing order. Hence, observe that, when a few server indices are selected, the server having the minimum of those indices will be the server with the minimum queue length among these. Hence, in this case the dispatchers in A and B select l_k and d_m random numbers (without replacement) from $\{1, 2, \dots, N\}$ and then send the incoming task to the servers having indices to be the minimum of those selected numbers. To couple the decisions of the two systems, at each arrival epoch a single random permutation of $\{1, 2, \dots, N\}$ is drawn, denoted by $\Sigma^{(N)} := (\sigma_1, \sigma_2, \dots, \sigma_N)$. Define $\sigma_{(i)} := \min_{j \leq i} \sigma_j$. Then observe that system A sends the task to the server with the index $\sigma_{(l_k)}$ and system B sends the task to the server with the index $\sigma_{(d_m)}$. Since at each arrival epoch both systems use a common random permutation, they take decisions in a coupled manner.

For the potential departure process, couple the service completion times of the kth queue in both scenarios, $k = 1, 2, \dots, N$. More precisely, for the potential departure process, assume that we have a single synchronized $\exp(N)$ clock independent of arrival epochs for both systems. Now, when this clock rings, a number k is uniformly selected from $\{1, 2, \dots, N\}$ and a potential departure occurs from the kth queue in both systems. If at a potential departure epoch an empty queue is selected then we do nothing. In this way the two schemes, considered independently, still evolve according to their appropriate statistical laws.

Loosely speaking, our next result is based upon the following intuition. Suppose that we have two systems A and B with two different schemes Π_A and Π_B having buffer sizes b and b' ($b \leq b'$), respectively. Also suppose that, for these two system, initially, $Q_i^A \leq Q_i^B$ for all $i = 1, \dots, b$. Below we develop some intuition as to under what conditions the initial ordering of the Q_i-values will be preserved after one arrival or departure.

For the departure process, if we ensure that departures will occur from the kth largest queue in both systems for some $k \in \{1, 2, \dots, N\}$ (ties are broken in any way), then observe that the ordering will be preserved after one departure.

In the case of the arrival process, assume that, when the minimum queue length in both systems is less than $b-1$, the incoming task is sent to the server with the same index. In that case it can be seen that the Q_i-values in A and B will preserve their ordering after the arrival as well. Next consider the case when the minimum queue length in both systems is precisely $b-1$. Now, in A, an incoming task can either be rejected (and will not change the Q-values at all) or be accepted (and $Q_b^{\Pi_A}$ will increase by 1). Here we ensure that if the incoming task is accepted in A then it is accepted in B as well unless $Q_b^{\Pi_A} < Q_b^{\Pi_B}$, in which case it is clear that the initial ordering will be preserved after the arrival. Finally, if the minimum queue length in A is less than $b-1$ and that in B is precisely $b-1$, then the way to ensure the inequality is either by making the scheme Π_A send the incoming task to the server with minimum queue length (and, hence, it will only increase the value of $Q_i^{\Pi_A}$ for some $i<b$, leaving other values unchanged) or by letting the selected server in Π_A have a smaller queue length than the selected server in Π_B. The former case corresponds to the condition $d=N$ and the latter corresponds to the condition $d \le d_{b-1}$, either of which has to be satisfied, in order to ensure the preservation of the ordering. This whole idea is formalized below.

Proposition 2. *For two schemes* $\Pi_A = \Pi(l_0, l_1, \dots, l_{b-1})$ *and* $\Pi_B = \Pi(d_0, d_1, \dots, d_{b'-1})$ *with* $b \le b'$, *assume that* $l_0 = \cdots = l_{b-2} = d_0 = \cdots = d_{b-2} = d$, $l_{b-1} \le d_{b-1}$, *and either* $d = N$ *or* $d \le d_{b-1}$. *Then*

(i) $\{Q_i^{\Pi_A}(t)\}_{t\ge 0} \le_{\mathrm{st}} \{Q_i^{\Pi_B}(t)\}_{t\ge 0}$ *for* $i = 1, 2, \dots, b$,

(ii) $\{\sum_{i=1}^{b} Q_i^{\Pi_A}(t) + L^{\Pi_A}(t)\}_{t\ge 0} \ge_{\mathrm{st}} \{\sum_{i=1}^{b'} Q_i^{\Pi_B}(t) + L^{\Pi_B}(t)\}_{t\ge 0}$,

(iii) $\{\Delta(t)\}_{t\ge 0} \ge \{\sum_{i=b+1}^{b'} Q_i^{\Pi_B}(t)\}_{t\ge 0}$ *almost surely under the coupling defined above,*

for any fixed $N \in \mathbb{N}$, *where* $\Delta(t) := L^{\Pi_A}(t) - L^{\Pi_B}(t)$, *provided that at time* $t = 0$ *the above ordering holds.*

Proof. To prove the stochastic ordering, we use the coupling of the schemes as described above and show that the ordering holds for the entire sample path. That is, the two processes arising from the above pair of schemes will be defined on a common probability space and it will then be shown that the ordering is maintained almost surely over all time.

Note that we shall consider only the event times $0 = t_0 < t_1 < \cdots$, i.e. the time epochs when arrivals or potential service completions occur and apply forward induction to show that the ordering is preserved. By assumption, the orderings hold at time $t_0 = 0$.

(i) The main idea of the proof is to use the coupling and show that at each event time the joint process of the two schemes follows a rule $\mathrm{Rule}(k, l, l_A, l_B)$ described in Subsection 3.1, with some random k, l, l_A, and l_B such that $l_A \ge l_B$ and either $l = 1$ or $l \ge l_B$, and apply Proposition 1. We now identify the rule at event time t_1 and verify that the conditions of Proposition 1 hold. If the event time t_1 is a potential departure epoch then, according to the coupling similarly as in the stack formation, a random $k \in \{1, 2, \dots, N\}$ will be chosen in both systems for a potential departure. Now assume that t_1 is an arrival epoch. In that case if the minimum queue length in both systems is less than $b-1$ then both schemes Π_A and Π_B will send the arriving task to the $\sigma_{(d)}$th queue. If the minimum queue length in scheme Π_A is $b-1$ then the incoming task is sent to the $\sigma_{(l_{b-1})}$th queue, and if in scheme Π_B the minimum queue length is $b-1$ then the incoming task is sent to the $\sigma_{(d_{b-1})}$th queue, where we recall that $(\sigma_1, \sigma_2, \dots, \sigma_N)$ is a random permutation of $\{1, 2, \dots, N\}$. Therefore, observe that at each step $\mathrm{Rule}(\sigma_{(d)}, k, \sigma_{(l_{b-1})}, \sigma_{(d_{b-1})})$ is followed.

Now to check the conditions, first observe that

$$\sigma_{(l_{b-1})} = \min_{i \le l_{b-1}} \sigma_i \ge \min_{i \le d_{b-1}} \sigma_i = \sigma_{(d_{b-1})},$$

where the second inequality is due to the assumption that $l_{b-1} \le d_{b-1}$. In addition, we have assumed that either $d = N$ or $d \le d_{b-1}$. If $d = N$ then the dispatcher sends the incoming task to the server with the minimum queue length, which is the same as sending to stack 1 as in Proposition 1. On the other hand, $d \le d_{b-1}$ implies that

$$\sigma_{(d)} = \min_{i \le d} \sigma_i \ge \min_{i \le d_{b-1}} \sigma_i = \sigma_{(d_{b-1})}.$$

Therefore, assertion (i) follows from Proposition 1.

(ii) We again apply forward induction. Assume that the ordering holds at time t_0. If the next event time is an arrival epoch then observe that both sides of the inequality in (ii) will increase, since if the incoming task is accepted then the Q-values will increase and if it is rejected then the L-value will increase.

On the other hand, if the next event time is a potential departure epoch then it suffices to show that if the left-hand side decreases then the right-hand side decreases as well. Indeed, from assertion (i) we know that $Q_1^{\Pi_A} \le Q_1^{\Pi_B}$ and, hence, we can see that if there is a departure from Π_A (i.e. the kth queue of Π_A is nonempty) then there will be a departure from Π_B (i.e. the kth queue of Π_B will be nonempty) as well.

(iii) Assertion (iii) follows directly from (i) and (ii). □

3.3. Discussion

It is worth emphasizing that Proposition 2(i) is fundamentally different from the stochastic majorization results for the ordinary JSQ policy, and below we contrast our methodology with some existing literature. As noted earlier, the ensemble of stacks, arranged in nondecreasing order, represents the empirical CDF of the queue length distribution at the various servers. Specifically, if we randomly select one of the servers then the probability that the queue length at that server is greater than or equal to i at time t under policy Π equals $(1/N)\mathbb{E}Q_i^{\Pi}(t)$. Thus, assertion (i) of Proposition 2 implies that if we select one of the servers at random then its queue length is stochastically larger under policy Π_B than under policy Π_A.

The latter property does generally *not* hold when we compare the ordinary JSQ policy with an alternative load balancing policy. Indeed, the class of load balancing schemes $\tilde{\Pi}^{(N)}$ (for the Nth system say) considered in [14] consists of all the schemes that have instantaneous queue length information for all the servers and that have to send an incoming task to some server if there is at least some place available anywhere in the whole system. This means that a scheme can only discard an incoming task if the system is completely full. Observe that *only* the JSQ policy lies both in the class $\Pi^{(N)}$ (defined in Section 3) and the class $\tilde{\Pi}^{(N)}$, because any scheme in $\Pi^{(N)}$ other than JSQ may reject an incoming task in some situations, where there might be some place available in the system. In this setup Towsley *et al.* [14] showed that, for any scheme $\Pi \in \tilde{\Pi}^{(N)}$ and all $t \ge 0$,

$$\sum_{i=1}^{k} Y_{(i)}^{\mathrm{JSQ}}(t) \le_{\mathrm{st}} \sum_{i=1}^{k} Y_{(i)}^{\Pi}(t) \quad \text{for } k = 1, 2, \ldots, N, \tag{2}$$

$$\{L^{\mathrm{JSQ}}(t)\}_{t\ge0} \le_{\mathrm{st}} \{L^{\Pi}(t)\}_{t\ge0},$$

where $Y_{(i)}^{\Pi}(t)$ is the ith largest queue length at time t in the system following scheme Π and $L^{\Pi}(t)$ is the total number of overflow events under policy Π up to time t, as defined in Section 2. Observe that $Y_{(i)}^{\Pi}$ can be visualized as the ith largest vertical bar (or stack) as described in Subsection 3.1. Thus, (2) says that the sum of the lengths of the k largest *vertical* stacks in a system following any scheme $\Pi \in \tilde{\Pi}^{(N)}$ is stochastically larger than or equal to that following the scheme JSQ for any $k = 1, 2, \dots, N$. Mathematically, this ordering can be written as

$$\sum_{i=1}^{b} \min\{k, Q_i^{\mathrm{JSQ}}(t)\} \leq_{\mathrm{st}} \sum_{i=1}^{b} \min\{k, Q_i^{\Pi}(t)\}$$

for all $k = 1, \dots, N$. In contrast, Proposition 2 shows that the length of the ith largest *horizontal* bar in the system following some scheme Π_A is stochastically smaller than that following some other scheme Π_B if some conditions are satisfied. Also, observe that the ordering between each of the horizontal bars (i.e. the Q_i) implies the ordering between the sums of the k largest vertical stacks, but not the other way around. Furthermore, it should be stressed that, in crude terms, JSQ in our class $\Pi^{(N)}$, plays the role of upper bound, whereas what (2) implies is almost the opposite in nature to the conditions we require.

While in [14] no policies with admission control (where the dispatcher can discard an incoming task even if the system is not full) were considered, in a later paper [11] and also in [13] the class was extended to a class $\hat{\Pi}^{(N)}$ consisting of all the policies that have information about instantaneous queue lengths available, and that can either send an incoming task to some server with available space or can reject an incoming task even if the system is not full. We can see that $\hat{\Pi}^{(N)}$ contains both $\tilde{\Pi}^{(N)}$ and $\Pi^{(N)}$ as subclasses. But then for such a class with admission control, Sparaggis *et al.* [11] noted that a stochastic ordering result like (2) cannot possibly hold. Instead, what was shown in [13] is that, for all $t \geq 0$,

$$\sum_{i=1}^{k} Y_{(i)}^{\mathrm{JSQ}}(t) + L^{\mathrm{JSQ}}(t) \leq_{\mathrm{st}} \sum_{i=1}^{k} Y_{(i)}^{\Pi}(t) + L^{\Pi}(t) \quad \text{for all } k \in \{1, 2, \dots, N\}. \tag{3}$$

Note that the ordering in (3) is the same in spirit as the ordering in Proposition 2(ii) and the inequalities in (3) are in the language of [13, Definition 14.4], *weak submajorization by* p, where $p = L^{\Pi}(t) - L^{\mathrm{JSQ}}(t)$. But in this case also our inequalities in Proposition 2(i) imply something completely orthogonal to what is implied by (3). In other words, the stochastic ordering results in Proposition 2 provide both upper and lower bounds for the occupancy state of one scheme with respect to another and are stronger than the stochastic majorization properties for the JSQ policy existing in the literature. Hence, we also needed to exploit a different proof methodology than the majorization framework developed in [11], [13], and [14].

4. Convergence on the diffusion scale

In this section we leverage the stochastic ordering established in Proposition 2 to prove the main result stated in Theorem 1. All the inequalities below are stated as almost-sure statements with respect to the common probability space constructed under the associated coupling. We shall use this joint probability space to make the probability statements about the marginals.

Proof of Theorem 1. Let $\Pi = \Pi(N, d_1, \dots, d_{b-1})$ be a load balancing scheme in the class $\Pi^{(N)}$. Denote by Π_1 the scheme $\Pi(N, d_1)$ with buffer size $b = 2$, and let Π_2 denote the JIQ policy $\Pi(N, 1)$ with buffer size $b = 2$.

https://doi.org/10.1017/jpr.2016.68 Published online by Cambridge University Press

Observe that from Proposition 2 we have, under the coupling defined in Subsection 3.2,

$$\begin{aligned} |Q_i^{\Pi}(t) - Q_i^{\Pi_2}(t)| &\le |Q_i^{\Pi}(t) - Q_i^{\Pi_1}(t)| + |Q_i^{\Pi_1}(t) - Q_i^{\Pi_2}(t)| \\ &\le |L^{\Pi_1}(t) - L^{\Pi}(t)| + |L^{\Pi_2}(t) - L^{\Pi_1}(t)| \\ &\le 2L^{\Pi_2}(t) \end{aligned} \tag{4}$$

for all $i \ge 1$ and $t \ge 0$ with the understanding that $Q_j(t) = 0$ for all $j > b$ for a scheme with buffer b. The third inequality above is due to Proposition 2(iii), which in particular says that $\{L^{\Pi_2}(t)\}_{t\ge 0} \ge \{L^{\Pi_1}(t)\}_{t\ge 0} \ge \{L^{\Pi}(t)\}_{t\ge 0}$ almost surely under the coupling. Now we have the following lemma which we will prove below.

Lemma 1. *For all $t \ge 0$, under the assumption of Theorem 1, $\{L^{\Pi_2}(t)\}_{N\ge 1}$ forms a tight sequence.*

Since $L^{\Pi_2}(t)$ is nondecreasing in t, the above lemma in particular implies that

$$\sup_{t\in[0,T]} \frac{L^{\Pi_2}(t)}{\sqrt{N}} \xrightarrow{\mathbb{P}} 0. \tag{5}$$

For any scheme $\Pi \in \Pi^{(N)}$, from (4) we know that

$$\{Q_i^{\Pi_2}(t) - 2L^{\Pi_2}(t)\}_{t\ge 0} \le \{Q_i^{\Pi}(t)\}_{t\ge 0} \le \{Q_i^{\Pi_2}(t) + 2L^{\Pi_2}(t)\}_{t\ge 0}.$$

Combining (4) and (5) shows that if the weak limits under the $\sqrt{N}$ scaling exist with respect to the Skorokhod J_1-topology, they must be the same for all the schemes in the class $\Pi^{(N)}$. Also, from Theorem 2 of [4] we know that the weak limit for $\Pi(N, N)$ exists and the common weak limit for the first two components can be described by the unique solution in $D \times D$ of the stochastic differential equations in (1). Hence, the proof of Theorem 1 is complete. □

Proof of Lemma 1. First we consider the evolution of $L^{\Pi_2}(t)$ as the following unit-jump counting process. A task arrival occurs at rate λ_N at the dispatcher, and if $Q_1^{\Pi_1} = N$ then it sends it to a server chosen uniformly at random. If the chosen server has queue length 2 then L^{Π_2} is increased by 1. It is easy to observe that this evolution can be equivalently described as follows. If $Q_1^{\Pi_2}(t) = N$ then each of the servers having queue length 2 starts increasing L^{Π_2} by 1 at rate λ_N/N. From this description we have

$$L^{\Pi_2}(t) = A\bigg(\int_0^t \frac{\lambda_N}{N} Q_2^{\Pi_2}(s)\mathbf{1}[Q_1^{\Pi_2}(s) = N]\,\mathrm{d}s\bigg)$$

with $A(\cdot)$ the unit-rate Poisson process. Now, using Proposition 2, it follows that $\mathbf{1}[Q_1^{\Pi_2}(s) = N] \le \mathbf{1}[Q_1^{\Pi_3}(s) = N]$ and $Q_2^{\Pi_2}(s) \le Q_2^{\Pi_3}(s)$, where $\Pi_3 = \Pi(N, N)$. Therefore, it is enough to prove the stochastic boundedness [10, Definition 5.4] of the sequence

$$\Gamma^{(N)}(t) := A\bigg(\int_0^t \frac{\lambda_N}{N} Q_2^{\Pi_3}(s)\mathbf{1}[Q_1^{\Pi_3}(s) = N]\,\mathrm{d}s\bigg).$$

To prove this, we shall use the martingale techniques described, for instance, in [10]. Define the filtration $\mathcal{F} \equiv \{\mathcal{F}_t : t \ge 0\}$, where, for $t \ge 0$,

$$\mathcal{F}_t := \sigma\bigg(Q^{\Pi_3}(0), A\bigg(\int_0^t \frac{\lambda_N}{N} Q_2^{\Pi_3}(s)\mathbf{1}[Q_1^{\Pi_3}(s) = N]\,\mathrm{d}s\bigg), Q_1^{\Pi_3}(s), Q_2^{\Pi_3}(s) : 0 \le s \le t\bigg).$$

Then, using a random time change of a unit-rate Poisson process [10, Lemma 3.2] and similar arguments to those in [10, Lemma 3.4], we have the next lemma.

Lemma 2. *With respect to the filtration $\mathcal{F}$,*

$$M^{(N)}(t) := A\bigg(\int_0^t \frac{\lambda_N}{N} Q_2^{\Pi_3}(s)\mathbf{1}[Q_1^{\Pi_3}(s) = N]\,\mathrm{d}s\bigg) - \int_0^t \frac{\lambda_N}{N} Q_2^{\Pi_3}(s)\mathbf{1}[Q_1^{\Pi_3}(s) = N]\,\mathrm{d}s$$

is a square-integrable martingale with $\mathcal{F}$-compensator

$$I(t) = \int_0^t \frac{\lambda_N}{N} Q_2^{\Pi_3}(s)\mathbf{1}[Q_1^{\Pi_3}(s) = N]\,\mathrm{d}s.$$

Moreover, the predictable quadratic variation process is given by $\langle M^{(N)}\rangle(t) = I(t)$.

Now we apply Lemma 5.8 of [10] which gives a stochastic boundedness criterion for square-integrable martingales.

Lemma 3. ([10, Lemma 5.8].) *Suppose that, for each $N \geq 1$, $M^{(N)} \equiv \{M^{(N)}(t)\colon t \geq 0\}$ is a square-integrable martingale (with respect to a specified filtration) with predictable quadratic variation process $\langle M^{(N)}\rangle \equiv \{\langle M^{(N)}\rangle(t)\colon t \geq 0\}$. If the sequence of random variables $\{\langle M^{(N)}\rangle(T)\colon N \geq 1\}$ is stochastically bounded in $\mathbb{R}$ for each $T > 0$, then the sequence of stochastic processes $\{M^{(N)}\colon N \geq 1\}$ is stochastically bounded in D.*

Therefore, it only remains to show the stochastic boundedness of $\{\langle M^{(N)}\rangle(T)\colon N \geq 1\}$ for each $T > 0$. Fix a $T > 0$, and observe that

$$\begin{aligned}\langle M^{(N)}\rangle(T) &= \frac{\lambda_N}{N}\int_0^T \frac{Q_2^{\Pi_3}(s)}{\sqrt{N}}\mathbf{1}[Q_1^{\Pi_3}(s) = N]\,\mathrm{d}s \\ &\leq \bigg[\sup_{t\in[0,T]} \frac{Q_2^{\Pi_3}(s)}{\sqrt{N}}\bigg]\int_0^T \frac{1}{\sqrt{N}}\mathbf{1}[Q_1^{\Pi_3}(s) = N]\lambda_N\,\mathrm{d}s. \qquad (6)\end{aligned}$$

From [4] we know that $\sup_{t\in[0,T]} Q_2^{\Pi_3}(t)/\sqrt{N}$ and $\int_0^T (1/\sqrt{N})\mathbf{1}[Q_1^{\Pi_3}(s) = N]\,\mathrm{d}A(\lambda_N s)$ are both tight. Moreover, since $\int_0^T (1/\sqrt{N})\mathbf{1}[Q_1^{\Pi_3}(s) = N]\lambda_N\,\mathrm{d}s$ is the intensity function of the stochastic integral $\int_0^T (1/\sqrt{N})\mathbf{1}[Q_1^{\Pi_3}(s) = N]\,\mathrm{d}A(\lambda_N s)$, which is a tight sequence, we have the following lemma.

Lemma 4. *For all fixed $T \geq 0$, $\int_0^T (1/\sqrt{N})\mathbf{1}[Q_1^{\Pi_3}(s) = N]\lambda_N\,\mathrm{d}s$ is tight as a sequence in N.*

Hence, both terms on the right-hand side of (6) are stochastically bounded and the resulting stochastic bound on $\langle M^{(N)}\rangle(T)$ completes the proof of Lemma 1. □

5. Conclusion

In the present paper we have considered a system with symmetric Markovian parallel queues and a single dispatcher. We established the diffusion limit of the queue process in the Halfin–Whitt regime for a wide class of load balancing schemes which always assign an incoming task to an idle server, if there is any. The results imply that assigning tasks to idle servers whenever possible is sufficient to achieve diffusion level optimality. Thus, using more fine-grained queue state information will increase the communication burden and potentially impact the scalability in large-scale deployments without significantly improving the performance.

In ongoing work we are aiming to extend the analysis to the stationary distribution of the queue process, and, in particular, to quantify the performance deviation from a system with

https://doi.org/10.1017/jpr.2016.68 Published online by Cambridge University Press

a single centralized queue. It would also be interesting to generalize the results to scenarios where the individual nodes have general state-dependent service rates rather than constant service rates.

Acknowledgements

This research was financially supported by an ERC Starting Grant and by The Netherlands Organization for Scientific Research (NWO) through the TOP-GO grant 613.001.012 and Gravitation Networks grant 024.002.003. Dr. Whiting was supported in part by the Australian Research grant DP-1592400 and in part by a Macquarie University Vice-Chancellor Innovation Fellowship.

References

[1] Badonnel, R. and Burgess, M. (2008). Dynamic pull-based load balancing for autonomic servers. In *Network Operations and Management Symposium*, NOMS 2008, pp. 751–754.
[2] Bramson, M., Lu, Y. and Prabhakar, B. (2012). Asymptotic independence of queues under randomized load balancing. *Queueing Systems* **71,** 247–292.
[3] Ephremides, A., Varaiya, P. and Walrand, J. (1980). A simple dynamic routing problem. *IEEE Trans. Automatic Control* **25,** 690–693.
[4] Eschenfeldt, P. and Gamarnik, D. (2015). Join the shortest queue with many servers. The heavy traffic asymptotics. Preprint. Available at https://arxiv.org/abs/1502.00999v2.
[5] Gupta, V., Harchol-Balter, M., Sigman, K. and Whitt, W. (2007). Analysis of join-the-shortest-queue routing for web server farms. *Performance Evaluation* **64,** 1062–1081.
[6] Halfin, S. and Whitt, W. (1981). Heavy-traffic limits for queues with many exponential servers. *Operat. Res.* **29,** 567–588.
[7] Jonckheere, M. (2006). Insensitive versus efficient dynamic load balancing in networks without blocking. *Queueing Systems* **54,** 193–202.
[8] Lu, Y. *et al.* (2011). Join-idle-queue: a novel load balancing algorithm for dynamically scalable web services. *Performance Evaluation* **68,** 1056–1071.
[9] Mitzenmacher, M. (2001). The power of two choices in randomized load balancing. *IEEE Trans. Parallel Distributed Systems* **12,** 1094–1104.
[10] Pang, G., Talreja, R. and Whitt, W. (2007). Martingale proofs of many-server heavy-traffic limits for Markovian queues. *Prob. Surveys* **4,** 193–267.
[11] Sparaggis, P. D., Towsley, D. and Cassandras, C. G. (1994). Sample path criteria for weak majorization. *Adv. Appl. Prob.* **26,** 155–171.
[12] Stolyar, A. L. (2015). Pull-based load distribution in large-scale heterogeneous service systems. *Queueing Systems* **80,** 341–361.
[13] Towsley, D. (1995). Application of majorization to control problems in queueing systems. In *Scheduling Theory and Its Applications*, eds P. Chrétienne *et al.*, John Wiley, Chichester.
[14] Towsley, D., Sparaggis, P. D. and Cassandras, C. G. (1992). Optimal routing and buffer allocation for a class of finite capacity queueing systems. *IEEE Trans. Automatic Control* **37,** 1446–1451.
[15] Vvedenskaya, N. D., Dobrushin, R. L. and Karpelevich, F. I. (1996). Queueing system with selection of the shortest of two queues: an asymptotic approach. *Prob. Peredachi Inf.* **32,** 20–34.
[16] Weber, R. R. (1978). On the optimal assignment of customers to parallel servers. *J. Appl. Prob.* **15,** 406–413.
[17] Whitt, W. (1986). Deciding which queue to join: some counterexamples. *Operat. Res.* **34,** 55–62.
[18] Winston, W. (1977). Optimality of the shortest line discipline. *J. Appl. Prob.* **14,** 181–189.