Scalable load balancing in networked systems

Scalable Load Balancing in Networked Systems: Universality Properties and Stochastic Coupling Methods

Mark van der Boor[1], Sem C. Borst[1,2], Johan S.H. van Leeuwaarden[1], and Debankur Mukherjee[1]

[1]*Eindhoven University of Technology, The Netherlands*
[2]*Nokia Bell Labs, Murray Hill, NJ, USA*

December 25, 2017

Abstract

We present an overview of scalable load balancing algorithms which provide favorable delay performance in large-scale systems, and yet only require minimal implementation overhead. Aimed at a broad audience, the paper starts with an introduction to the basic load balancing scenario – referred to as the *supermarket model* – consisting of a single dispatcher where tasks arrive that must immediately be forwarded to one of N single-server queues. The supermarket model is a dynamic counterpart of the classical balls-and-bins setup where balls must be sequentially distributed across bins.

A popular class of load balancing algorithms are so-called power-of-d or JSQ(d) policies, where an incoming task is assigned to a server with the shortest queue among d servers selected uniformly at random. As the name reflects, this class includes the celebrated Join-the-Shortest-Queue (JSQ) policy as a special case ($d = N$), which has strong stochastic optimality properties and yields a mean waiting time that *vanishes* as N grows large for any fixed subcritical load. However, a nominal implementation of the JSQ policy involves a prohibitive communication burden in large-scale deployments. In contrast, a simple random assignment policy ($d = 1$) does not entail any communication overhead, but the mean waiting time remains constant as N grows large for any fixed positive load.

In order to examine the fundamental trade-off between delay performance and implementation overhead, we consider an asymptotic regime where the diversity parameter $d(N)$ depends on N. We investigate what growth rate of $d(N)$ is required to match the optimal performance of the JSQ policy on fluid and diffusion scale, and achieve a vanishing waiting time in the limit. The results demonstrate that the asymptotics for the JSQ($d(N)$) policy are insensitive to the exact growth rate of $d(N)$, as long as the latter is sufficiently fast, implying that the optimality of the JSQ policy can asymptotically be preserved while dramatically reducing the communication overhead.

Stochastic coupling techniques play an instrumental role in establishing the asymptotic optimality and universality properties, and augmentations of the coupling constructions allow these properties to be extended to infinite-server settings and network scenarios. We additionally show how the communication overhead can be reduced yet further by the so-called Join-the-Idle-Queue (JIQ) scheme, leveraging memory at the dispatcher to keep track of idle servers.

1 Introduction

In the present paper we review scalable load balancing algorithms (LBAs) which achieve excellent delay performance in large-scale systems and yet only involve low implementation overhead. LBAs play a critical role in distributing service requests or tasks (e.g. compute jobs, data base look-ups, file transfers) among servers or distributed resources in parallel-processing systems. The analysis and design of LBAs has attracted strong attention in recent years, mainly spurred by crucial scalability challenges arising in cloud networks and data centers with massive numbers of servers.

LBAs can be broadly categorized as static, dynamic, or some intermediate blend, depending on the amount of feedback or state information (e.g. congestion levels) that is used in allocating tasks. The use of state information naturally allows dynamic policies to achieve better delay performance, but also involves higher implementation complexity and a substantial communication burden. The latter issue is particularly pertinent in cloud networks and data centers with immense numbers of servers handling a huge influx of service requests. In order to capture the large-scale context, we examine scalability properties through the prism of asymptotic scalings where the system size grows large, and identify LBAs which strike an optimal balance between delay performance and implementation overhead in that regime.

The most basic load balancing scenario consists of N identical parallel servers and a dispatcher where tasks arrive that must immediately be forwarded to one of the servers. Tasks are assumed to have unit-mean exponentially distributed service requirements, and the service discipline at each server is supposed to be oblivious to the actual service requirements. In this canonical setup, the celebrated Join-the-Shortest-Queue (JSQ) policy has several strong stochastic optimality properties. In particular, the JSQ policy achieves the minimum mean overall delay among all non-anticipating policies that do not have any advance knowledge of the service requirements [9, 45]. In order to implement the JSQ policy however, a dispatcher requires instantaneous knowledge of all the queue lengths, which may involve a prohibitive communication burden with a large number of servers N.

This poor scalability has motivated consideration of JSQ(d) policies, where an incoming task is assigned to a server with the shortest queue among $d \geq 2$ servers selected uniformly at random. Note that this involves exchange of $2d$ messages per task, irrespective of the number of servers N. Results in Mitzenmacher [26] and Vvedenskaya *et al.* [43] indicate that even sampling as few as $d = 2$ servers yields significant performance enhancements over purely random assignment ($d = 1$) as N grows large, which is commonly referred to as the "power-of-two" or "power-of-choice" effect. Specifically, when tasks arrive at rate λN, the queue length distribution at each individual server exhibits super-exponential decay for any fixed $\lambda < 1$ as N grows large, compared to exponential decay for purely random assignment.

As illustrated by the above, the diversity parameter d induces a fundamental trade-off between the amount of communication overhead and the delay performance. Specifically, a random assignment policy does not entail any communication burden, but the mean waiting time remains *constant* as N grows large for any fixed $\lambda > 0$. In contrast, a nominal implementation of the JSQ policy (without maintaining state information at the dispatcher) involves $2N$ messages per task, but the mean waiting time *vanishes* as N grows large for any fixed $\lambda < 1$. Although JSQ(d) policies with $d \geq 2$ yield major performance improvements over purely random assignment while reducing the communication burden by a factor O(N) compared to the JSQ policy, the mean waiting time *does not vanish* in the limit. Thus, no fixed value of d will provide asymptotically optimal delay performance. This is evidenced by results of Gamarnik *et al.* [14] indicating that in the absence of any memory at the dispatcher the communication overhead per task *must increase* with N in order for any scheme to achieve a zero mean waiting time in the limit.

We will explore the intrinsic trade-off between delay performance and communication overhead as governed by the diversity parameter d, in conjunction with the relative load λ. The latter trade-off is examined in an asymptotic regime where not only the overall task arrival rate is assumed to grow with N, but also the diversity parameter is allowed to depend on N. We write $\lambda(N)$ and $d(N)$, respectively, to explicitly reflect that, and investigate what growth rate of $d(N)$ is required, depending on the scaling behavior of $\lambda(N)$, in order to achieve a zero mean waiting time in the limit. We establish that the fluid-

scale and diffusion-scale limiting processes are insensitive to the exact growth rate of $d(N)$, as long as the latter is sufficiently fast, and in particular coincide with the limiting processes for the JSQ policy. This reflects a remarkable universality property and demonstrates that the optimality of the JSQ policy can asymptotically be preserved while dramatically lowering the communication overhead.

We will extend the above-mentioned universality properties to network scenarios where the N servers are assumed to be inter-connected by some underlying graph topology G_N. Tasks arrive at the various servers as independent Poisson processes of rate λ, and each incoming task is assigned to whichever server has the shortest queue among the one where it appears and its neighbors in G_N. In case G_N is a clique, each incoming task is assigned to the server with the shortest queue across the entire system, and the behavior is equivalent to that under the JSQ policy. The above-mentioned stochastic optimality properties of the JSQ policy thus imply that the queue length process in a clique will be 'better' than in an arbitrary graph G_N. We will establish sufficient conditions for the fluid-scaled and diffucion-scaled versions of the queue length process in an arbitrary graph to be equivalent to the limiting processes in a clique as $N \to \infty$. The conditions reflect similar universality properties as described above, and in particular demonstrate that the optimality of a clique can asymptotically be preserved while markedly reducing the number of connections, provided the graph G_N is suitably random.

While a zero waiting time can be achieved in the limit by sampling only $d(N) = o(N)$ servers, the amount of communication overhead in terms of $d(N)$ must still grow with N. This may be explained from the fact that a large number of servers need to be sampled for each incoming task to ensure that at least one of them is found idle with high probability. As alluded to above, this can be avoided by introducing memory at the dispatcher, in particular maintaining a record of vacant servers, and assigning tasks to idle servers, if there are any. This so-called Join-the-Idle-Queue (JIQ) scheme [5, 22] has gained huge popularity recently, and can be implemented through a simple token-based mechanism generating at most one message per task. As established by Stolyar [37], the fluid-scaled queue length process under the JIQ scheme is equivalent to that under the JSQ policy as $N \to \infty$, and this result can be shown to extend the diffusion-scaled queue length process. Thus, the use of memory allows the JIQ scheme to achieve asymptotically optimal delay performance with minimal communication overhead. In particular, ensuring that tasks are assigned to idle servers whenever available is sufficient to achieve asymptotic optimality, and using any additional queue length information yields no meaningful performance benefits on the fluid or diffusion levels.

Stochastic coupling techniques play an instrumental role in the proofs of the above-described universality and asymptotic optimality properties. A direct analysis of the queue length processes under a JSQ($d(N)$) policy, in a load balancing graph G_N, or under the JIQ scheme is confronted with unsurmountable obstacles. As an alternative route, we leverage novel stochastic coupling constructions to relate the relevant queue length processes to the corresponding processes under a JSQ policy, and show that the deviation between these two is asymptotically negligible under mild assumptions on $d(N)$ or G_N.

While the stochastic coupling schemes provide a remarkably effective and overarching approach, they defy a systematic recipe and involve some degree of ingenuity and customization. Indeed, the specific coupling arguments that we develop are not only different from those that were originally used in establishing the stochastic optimality properties of the JSQ policy, but also differ in critical ways between a JSQ($d(N)$) policy, a load balancing graph G_N, and the JIQ scheme. Yet different coupling constructions are devised for model variants with infinite-server dynamics that we will discuss in Section 4.

The remainder of the paper is organized as follows. In Section 2 we discuss a wide spectrum of LBAs and evaluate their scalability properties. In Section 3 we introduce some useful preliminaries, review fluid and diffusion limits for the JSQ policy as well as JSQ(d) policies with a fixed value of d, and explore the trade-off between delay performance and communication overhead as function of the diversity parameter d. In particular, we establish asymptotic universality properties for JSQ(d) policies, which are extended to systems with server pools and network scenarios in Sections 4 and 5, respectively. In Section 6 we establish asymptotic optimality properties for the JIQ scheme. We discuss somewhat related redundancy policies and alternative scaling regimes and performance metrics in Section 7.

2 Scalability spectrum

In this section we review a wide spectrum of LBAs and examine their scalability properties in terms of the delay performance vis-a-vis the associated implementation overhead in large-scale systems.

2.1 Basic model

Throughout this section and most of the paper, we focus on a basic scenario with N parallel single-server infinite-buffer queues and a single dispatcher where tasks arrive as a Poisson process of rate $\lambda(N)$, as depicted in Figure 2. Arriving tasks cannot be queued at the dispatcher, and must immediately be forwarded to one of the servers. This canonical setup is commonly dubbed the *supermarket model*. Tasks are assumed to have unit-mean exponentially distributed service requirements, and the service discipline at each server is supposed to be oblivious to the actual service requirements.

In Section 4 we consider some model variants with N server pools and possibly finite buffers and in Section 5 we will treat network generalizations of the above model.

2.2 Asymptotic scaling regimes

An exact analysis of the delay performance is quite involved, if not intractable, for all but the simplest LBAs. Numerical evaluation or simulation are not straightforward either, especially for high load levels and large system sizes. A common approach is therefore to consider various limit regimes, which not only provide mathematical tractability and illuminate the fundamental behavior, but are also natural in view of the typical conditions in which cloud networks and data centers operate. One can distinguish several asymptotic scalings that have been used for these purposes: (i) In the classical heavy-traffic regime, $\lambda(N) = \lambda N$ with a fixed number of servers N and a relative load λ that tends to one in the limit. (ii) In the conventional large-capacity or many-server regime, the relative load $\lambda(N)/N$ approaches a constant $\lambda < 1$ as the number of servers N grows large. (iii) The popular Halfin-Whitt regime [17] combines heavy traffic with a large capacity, with

$$\frac{N - \lambda(N)}{\sqrt{N}} \to \beta > 0 \text{ as } N \to \infty, \tag{2.1}$$

so the relative capacity slack behaves as $\beta/\sqrt{N}$ as the number of servers N grows large. (iv) The so-called non-degenerate slow-down regime [2] involves $N - \lambda(N) \to \gamma > 0$, so the relative capacity slack shrinks as γ/N as the number of servers N grows large.

The term non-degenerate slow-down refers to the fact that in the context of a centralized multi-server queue, the mean waiting time in regime (iv) tends to a strictly positive constant as $N \to \infty$, and is thus of similar magnitude as the mean service requirement. In contrast, in regimes (ii) and (iii), the mean waiting time decays exponentially fast in N or is of the order $1/\sqrt{N}$, respectively, as $N \to \infty$, while in regime (i) the mean waiting time grows arbitrarily large relative to the mean service requirement.

In the present paper we will focus on scalings (ii) and (iii), and occasionally also refer to these as fluid and diffusion scalings, since it is natural to analyze the relevant queue length process on fluid scale $(1/N)$ and diffusion scale $(1/\sqrt{N})$ in these regimes, respectively. We will not provide a detailed account of scalings (i) and (iv), which do not capture the large-scale perspective and do not allow for low delays, respectively, but we will briefly revisit these regimes in Section 7.

2.3 Random assignment: N independent M/M/1 queues

One of the most basic LBAs is to assign each arriving task to a server selected uniformly at random. In that case, the various queues collectively behave as N independent M/M/1 queues, each with arrival rate $\lambda(N)/N$ and unit service rate. In particular, at each of the queues, the total number of tasks in stationarity has a geometric distribution with parameter $\lambda(N)/N$. By virtue of the PASTA property, the probability

that an arriving task incurs a non-zero waiting time is $\lambda(N)/N$. The mean number of waiting tasks (excluding the possible task in service) at each of the queues is $\frac{\lambda(N)^2}{N(N-\lambda(N))}$, so the total mean number of waiting tasks is $\frac{\lambda(N)^2}{N-\lambda(N)}$, which by Little's law implies that the mean waiting time of a task is $\frac{\lambda(N)}{N-\lambda(N)}$. In particular, when $\lambda(N) = N\lambda$, the probability that a task incurs a non-zero waiting time is λ, and the mean waiting time of a task is $\frac{\lambda}{1-\lambda}$, independent of N, reflecting the independence of the various queues.

A slightly better LBA is to assign tasks to the servers in a Round-Robin manner, dispatching every N-th task to the same server. In the large-capacity regime where $\lambda(N) = N\lambda$, the inter-arrival time of tasks at each given queue will then converge to a constant $1/\lambda$ as $N \to \infty$. Thus each of the queues will behave as an D/M/1 queue in the limit, and the probability of a non-zero waiting time and the mean waiting time will be somewhat lower than under purely random assignment. However, both the probability of a non-zero waiting time and the mean waiting time will still tend to strictly positive values and not vanish as $N \to \infty$.

2.4 Join-the-Shortest Queue (JSQ)

Under the Join-the-Shortest-Queue (JSQ) policy, each arriving task is assigned to the server with the currently shortest queue (ties are broken arbitrarily). In the basic model described above, the JSQ policy has several strong stochastic optimality properties, and yields the 'most balanced and smallest' queue process among all non-anticipating policies that do not have any advance knowledge of the service requirements [9, 45]. Specifically, the JSQ policy minimizes the joint queue length vector in a stochastic majorization sense, and in particular stochastically minimizes the total number of tasks in the system, and hence the mean overall delay. In order to implement the JSQ policy however, a dispatcher requires instantaneous knowledge of the queue lengths at all the servers. A nominal implementation would involve exchange of $2N$ messages per task, and thus yield a prohibitive communication burden in large-scale systems.

2.5 Join-the-Smallest-Workload (JSW): centralized M/M/N queue

Under the Join-the-Smallest-Workload (JSW) policy, each arriving task is assigned to the server with the currently smallest workload. Note that this is an anticipating policy, since it requires advance knowledge of the service requirements of all the tasks in the system. Further observe that this policy (myopically) minimizes the waiting time for each incoming task, and mimics the operation of a centralized N-server queue with a FCFS discipline. The equivalence with a centralized N-server queue yields a strong optimality property of the JSW policy: The vector of joint workloads at the various servers observed by each incoming task is smaller in the Schur convex sense than under any alternative admissible policy [13].

The equivalence with a centralized FCFS queue means that there cannot be any idle servers while tasks are waiting. In our setting with Poisson arrivals and exponential service requirements, it can therefore be shown that the total number of tasks under the JSW policy is stochastically smaller than under the JSQ policy. At the same time, it means that the total number of tasks under the JSW policy behaves as a birth-death process, which renders it far more tractable than the JSQ policy. Specifically, given that all the servers are busy, the total number of waiting tasks is geometrically distributed with parameter $\lambda(N)/N$. Thus the total mean number of waiting tasks is $\Pi_W(N, \lambda(N))\frac{\lambda(N)}{N-\lambda(N)}$, and the mean waiting time is $\Pi_W(N, \lambda(N))\frac{1}{N-\lambda(N)}$, with $\Pi_W(N, \lambda(N)$ denoting the probability of all servers being occupied and a task incurring a non-zero waiting time. This immediately shows that the mean waiting time is smaller by at least a factor $\lambda(N)$ than for the random assignment policy considered in Subsection 2.3.

In the large-capacity regime $\lambda(N) = N\lambda$, it can be shown that the probability $\Pi_W(N, \lambda(N))$ of a non-zero waiting time decays exponentially fast in N, and hence so does the mean waiting time. In the Halfin-Whitt heavy-traffic regime (2.1), the probability $\Pi_W(N, \lambda(N))$ of a non-zero waiting time converges to a finite constant $\Pi^\star_W(\beta)$, implying that the mean waiting time of a task is of the order $1/\sqrt{N}$, and thus vanishes as $N \to \infty$.

2.6 Power-of-d load balancing (JSQ(d))

As mentioned above, the achilles heel of the JSQ policy is its excessive communication overhead in large-scale systems. This poor scalability has motivated consideration of so-called JSQ(d) policies, where an incoming task is assigned to a server with the shortest queue among d servers selected uniformly at random. Results in Mitzenmacher [26] and Vvedenskaya *et al.* [43] indicate that even sampling as few as $d = 2$ servers yields significant performance enhancements over purely random assignment ($d = 1$) as $N \to \infty$. Specifically, in the fluid regime where $\lambda(N) = \lambda N$, the probability that there are i or more tasks at a given queue is proportional to $\lambda^{\frac{d^i-1}{d-1}}$ as $N \to \infty$, and thus exhibits super-exponential decay as opposed to exponential decay for the random assignment policy considered in Subsection 2.3.

As illustrated by the above, the diversity parameter d induces a fundamental trade-off between the amount of communication overhead and the performance in terms of queue lengths and delays. A rudimentary implementation of the JSQ policy ($d = N$, without replacement) involves $O(N)$ communication overhead per task, but it can be shown that the probability of a non-zero waiting time and the mean waiting *vanish* as $N \to \infty$, just like in a centralized queue. Although JSQ(d) policies with a fixed parameter $d \geq 2$ yield major performance improvements over purely random assignment while reducing the communication burden by a factor O(N) compared to the JSQ policy, the probability of a non-zero waiting time and the mean waiting time *do not vanish* as $N \to \infty$.

In Subsection 3.5 we will explore the intrinsic trade-off between delay performance and communication overhead as function of the diversity parameter d, in conjunction with the relative load. We will examine an asymptotic regime where not only the total task arrival rate $\lambda(N)$ is assumed to grow with N, but also the diversity parameter is allowed to depend on N. As will be demonstrated, the optimality of the JSQ policy ($d(N) = N$) can be preserved, and in particular a vanishing waiting time can be achieved in the limit as $N \to \infty$, even when $d(N) = o(N)$, thus dramatically lowering the communication overhead.

2.7 Token-based strategies: Join-the-Idle-Queue (JIQ)

While a zero waiting time can be achieved in the limit by sampling only $d(N) = o(N)$ servers, the amount of communication overhead in terms of $d(N)$ must still grow with N. This can be countered by introducing memory at the dispatcher, in particular maintaining a record of vacant servers, and assigning tasks to idle servers as long as there are any, or to a uniformly at random selected server otherwise. This so-called Join-the-Idle-Queue (JIQ) scheme [5, 22] has received keen interest recently, and can be implemented through a simple token-based mechanism. Specifically, idle servers send tokens to the dispatcher to advertise their availability, and when a task arrives and the dispatcher has tokens available, it assigns the task to one of the corresponding servers (and disposes of the token). Note that a server only issues a token when a task completion leaves its queue empty, thus generating at most one message per task. Surprisingly, the mean waiting time and the probability of a non-zero waiting time vanish under the JIQ scheme in both the fluid and diffusion regimes, as we will further discuss in Section 6. Thus, the use of memory allows the JIQ scheme to achieve asymptotically optimal delay performance with minimal communication overhead.

2.8 Performance comparison

We now present some simulation experiments that we have conducted to compare the above-described LBAs in terms of delay performance. Specifically, we evaluate the mean waiting time and the probability of a non-zero waiting time in both a fluid regime ($\lambda(N) = 0.9N$) and a diffusion regime ($\lambda(N) = N - \sqrt{N}$). The results are shown in Figure 1. We are especially interested in distinguishing two classes of LBAs – ones delivering a mean waiting time and probability of a non-zero waiting time that vanish asymptotically, and ones that fail to do so – and relating that dichotomy to the associated overhead.

JSQ, JIQ, and JSW. JSQ, JIQ and JSW evidently have a vanishing waiting time in both the fluid and the diffusion regime as discussed in Subsections 2.4, 2.5 and 2.7. The optimality of JSW as mentioned in

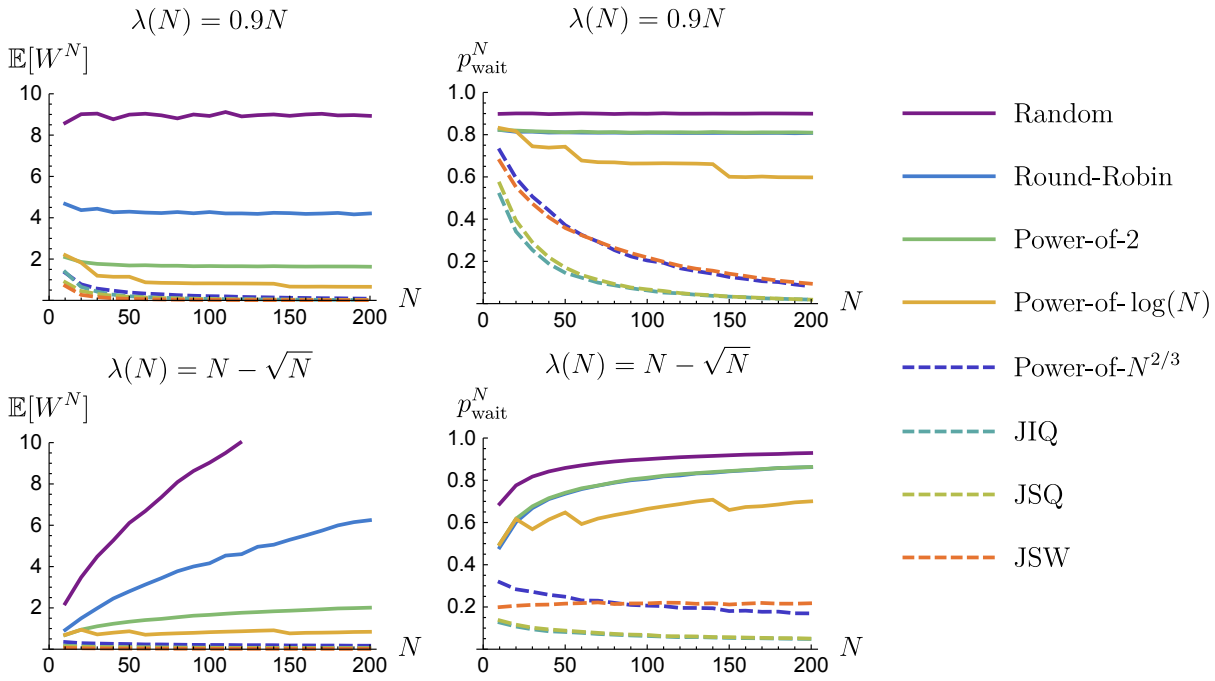


Figure 1: Simulation results for mean waiting time $\mathbb{E}[W^N]$ and probability of a non-zero waiting time p^N_{wait}, for both a fluid regime and a diffusion regime.

Subsection 2.5 can also be clearly observed.

However, there is a significant difference between JSW and JSQ/JIQ in the diffusion regime. We observe that the probability of a non-zero waiting time *approaches a positive constant* for JSW, while it *vanishes* for JSQ/JIQ. In other words, the mean of all positive waiting times is of a larger order of magnitude in JSQ/JIQ compared to JSW. Intuitively, this is clear since in JSQ/JIQ, when a task is placed in a queue, it waits for at least a residual service time. In JSW, which is equivalent to the M/M/N queue, a task that cannot start service immediately, joins a queue that is collectively drained by all the N servers

Random and Round-Robin. The mean waiting time does not vanish for Random and Round-Robin in the fluid regime, as already mentioned in Subsection 2.3. Moreover, the mean waiting time grows without bound in the diffusion regime for these two schemes. This is because the system can still be decomposed, and the loads of the individual M/M/1 and D/M/1 queues tend to 1.

JSQ(d) policies. Three versions of JSQ(d) are included in the figures; $d(N) = 2 \not\to \infty$, $d(N) = \lfloor \log(N) \rfloor \to \infty$ and $d(N) = N^{2/3}$ for which $\frac{d(N)}{\sqrt{N}\log(N)} \to \infty$. Note that the graph for $d(N) = \lfloor \log(N) \rfloor$ shows sudden jumps when $d(N)$ increases by 1. The variants for which $d(N) \to \infty$ have a vanishing waiting time in the fluid regime, while $d = 2$ does not. The latter observation is a manifestation of the results of Gamarnik *et al.* [14] mentioned in the introduction, since JSQ(d) uses no memory and the overhead per task does not increase with N. Furthermore, it follows that JSQ(d) policies outperform Random and Round-Robin, while JSQ/JIQ/JSW are better in terms of mean waiting time.

In order to succinctly capture the results and observed dichotomy in Figure 1, we provide an overview of the delay performance of the various LBAs and the associated overhead in Table 1, where $q_i^\star$ denotes the stationary fraction of servers with i or more tasks.

Scheme	Queue length	Waiting time (fixed $\lambda < 1$)	Waiting time ($1 - \lambda \sim 1/\sqrt{N}$)	Overhead per task
Random	$q_i^\star = \lambda^i$	$\frac{\lambda}{1-\lambda}$	$\Theta(\sqrt{N})$	0
JSQ(d)	$q_i^\star = \lambda^{\frac{d^i-1}{d-1}}$	$\Theta(1)$	$\Omega(\log N)$	$2d$
$d(N) \to \infty$	same as JSQ	same as JSQ	??	$2d(N)$
$\frac{d(N)}{\sqrt{N}\log(N)} \to \infty$	same as JSQ	same as JSQ	same as JSQ	$2d(N)$
JSQ	$q_1^\star = \lambda$, $q_2^\star = \mathrm{o}(1)$	$\mathrm{o}(1)$	$\Theta(1/\sqrt{N})$	$2N$
JIQ	same as JSQ	same as JSQ	same as JSQ	≤ 1

Table 1: Queue length distribution, waiting times and communication overhead for various LBAs.

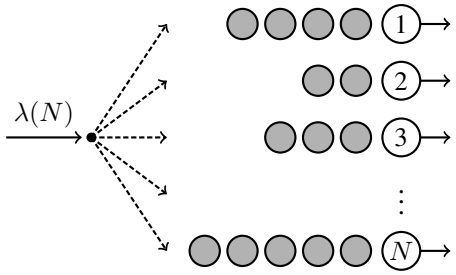


Figure 2: Tasks arrive at the dispatcher as a Poisson process of rate $\lambda(N)$, and are forwarded to one of the N servers according to some specific load balancing algorithm.

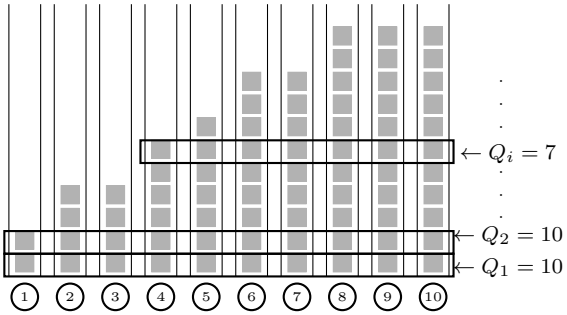


Figure 3: The value of Q_i represents the width of the i-th row, when the servers are arranged in non-descending order of their queue lengths.

3 JSQ(d) policies and universality properties

In this section we first introduce some useful preliminary concepts, then review fluid and diffusion limits for the JSQ policy as well as JSQ(d) policies with a fixed value of d, and finally discuss universality properties when the diversity parameter $d(N)$ is being scaled with N.

As described in the previous section, we focus on a basic scenario where all the servers are homogeneous, the service requirements are exponentially distributed, and the service discipline at each server is oblivious of the actual service requirements. In order to obtain a Markovian state description, it therefore suffices to only track the number of tasks, and in fact we do not need to keep record of the number of tasks at each individual server, but only count the number of servers with a given number of tasks. Specifically, we represent the state of the system by a vector $\mathbf{Q}(t) := (Q_1(t), Q_2(t), \dots)$, with $Q_i(t)$ denoting the number of servers with i or more tasks at time t, including the possible task in service, $i = 1, 2 \dots$. Note that if we represent the queues at the various servers as (vertical) stacks, and arrange these from left to right in non-descending order, then the value of Q_i corresponds to the width of the i-th (horizontal) row, as depicted in the schematic diagram in Figure 3.

In order to examine the asymptotic behavior when the number of servers N grows large, we consider a sequence of systems indexed by N, and attach a superscript N to the associated state variables.

The fluid-scaled occupancy state is denoted by $\mathbf{q}^N(t) := (q_1^N(t), q_2^N(t), \dots)$, with $q_i^N(t) = Q_i^N(t)/N$ representing the fraction of servers in the N-th system with i or more tasks as time t, $i = 1, 2, \dots$. Let $\mathcal{S} = \{\mathbf{q} \in [0,1]^\infty : q_i \leq q_{i-1} \forall i = 2, 3, \dots\}$ be the set of all possible fluid-scaled states. Whenever we

consider fluid limits, we assume the sequence of initial states is such that $\mathbf{q}^N(0) \to \mathbf{q}^\infty \in \mathcal{S}$ as $N \to \infty$.

The diffusion-scaled occupancy state is defined as $\bar{\mathbf{Q}}^N(t) = (\bar{Q}_1^N(t), \bar{Q}_2^N(t), \dots)$, with

$$\bar{Q}_1^N(t) = -\frac{N - Q_1^N(t)}{\sqrt{N}}, \qquad \bar{Q}_i^N(t) = \frac{Q_i^N(t)}{\sqrt{N}}, \quad i = 2, 3, \dots. \tag{3.1}$$

Note that $-\bar{Q}_1^N(t)$ corresponds to the number of vacant servers, normalized by $\sqrt{N}$. The reason why $Q_1^N(t)$ is centered around N while $Q_i^N(t)$, $i = 2, 3, \dots$, are not, is because for the scalable LBAs that we pursue, the fraction of servers with exactly one task tends to one, whereas the fraction of servers with two or more tasks tends to zero as $N \to \infty$.

3.1 Fluid limit for JSQ(d) policies

We first consider the fluid limit for JSQ(d) policies with an arbitrary but fixed value of d as characterized by Mitzenmacher [26] and Vvedenskaya *et al.* [43].

The sequence of processes $\{\mathbf{q}^N(t)\}_{t\geq 0}$ has a weak limit $\{\mathbf{q}(t)\}_{t\geq 0}$ that satisfies the system of differential equations

$$\frac{\mathrm{d}q_i(t)}{\mathrm{d}t} = \lambda[(q_{i-1}(t))^d - (q_i(t))^d] - [q_i(t) - q_{i+1}(t)], \quad i = 1, 2, \dots. \tag{3.2}$$

The fluid-limit equations may be interpreted as follows. The first term represents the rate of increase in the fraction of servers with i or more tasks due to arriving tasks that are assigned to a server with exactly $i-1$ tasks. Note that the latter occurs in fluid state $\mathbf{q} \in \mathcal{S}$ with probability $q_{i-1}^d - q_i^d$, i.e., the probability that all d sampled servers have $i-1$ or more tasks, but not all of them have i or more tasks. The second term corresponds to the rate of decrease in the fraction of servers with i or more tasks due to service completions from servers with exactly i tasks, and the latter rate is given by $q_i - q_{i+1}$.

The unique fixed point of (3.2) for any $d \geq 2$ is obtained as

$$q_i^\star = \lambda^{\frac{d^i-1}{d-1}}, \quad i = 1, 2, \dots. \tag{3.3}$$

It can be shown that the fixed point is asymptotically stable in the sense that $\mathbf{q}(t) \to \mathbf{q}^\star$ as $t \to \infty$ for any initial fluid state $\mathbf{q}^\infty$ with $\sum_{i=1}^\infty q_i^\infty < \infty$. The fixed point reveals that the stationary queue length distribution at each individual server exhibits super-exponential decay as $N \to \infty$, as opposed to exponential decay for a random assignment policy. It is worth observing that this involves an interchange of the many-server ($N \to \infty$) and stationary ($t \to \infty$) limits. The justification is provided by the asymptotic stability of the fixed point along with a few further technical conditions.

3.2 Fluid limit for JSQ policy

We now turn to the fluid limit for the ordinary JSQ policy, which rather surprisingly was not rigorously established until fairly recently in [31], leveraging martingale functional limit theorems and time-scale separation arguments [18].

In order to state the fluid limit starting from an arbitrary fluid-scaled occupancy state, we first introduce some additional notation. For any fluid state $\mathbf{q} \in \mathcal{S}$, denote by $m(\mathbf{q}) = \min\{i : q_{i+1} < 1\}$ the minimum queue length among all servers. Now if $m(\mathbf{q}) = 0$, then define $p_0(m(\mathbf{q})) = 1$ and $p_i(m(\mathbf{q})) = 0$ for all $i = 1, 2, \dots$. Otherwise, in case $m(\mathbf{q}) > 0$, define

$$p_i(\mathbf{q}) = \begin{cases} \min\left\{(1 - q_{m(\mathbf{q})+1})/\lambda, 1\right\} & \text{for} \quad i = m(\mathbf{q}) - 1, \\ 1 - p_{m(\mathbf{q})-1}(\mathbf{q}) & \text{for} \quad i = m(\mathbf{q}), \end{cases} \tag{3.4}$$

and $p_i(\mathbf{q}) = 0$ otherwise. The coefficient $p_i(\mathbf{q})$ represents the instantaneous fraction of incoming tasks assigned to servers with a queue length of exactly i in the fluid state $\mathbf{q} \in \mathcal{S}$.

Any weak limit of the sequence of processes $\{\mathbf{q}^N(t)\}_{t\geq 0}$ is given by the deterministic system $\{\mathbf{q}(t)\}_{t\geq 0}$ satisfying the following system of differential equations

$$\frac{\mathrm{d}^+ q_i(t)}{\mathrm{d}t} = \lambda p_{i-1}(\mathbf{q}(t)) - (q_i(t) - q_{i+1}(t)), \quad i = 1, 2, \ldots, \tag{3.5}$$

where d^+/dt denotes the right-derivative.

The unique fixed point $\mathbf{q}^\star = (q_1^\star, q_2^\star, \ldots)$ of the dynamical system in (3.5) is given by

$$q_i^\star = \begin{cases} \lambda, & i = 1, \\ 0, & i = 2, 3, \ldots. \end{cases} \tag{3.6}$$

The fixed point in (3.6), in conjunction with an interchange of limits argument, indicates that in stationarity the fraction of servers with a queue length of two or larger under the JSQ policy is negligible as $N \to \infty$.

3.3 Diffusion limit for JSQ policy

We next describe the diffusion limit for the JSQ policy in the Halfin-Whitt heavy-traffic regime (2.1), as recently derived by Eschenfeldt & Gamarnik [10].

For suitable initial conditions, the sequence of processes $\left\{\bar{\mathbf{Q}}^N(t)\right\}_{t\geq 0}$ as in (3.1) *converges weakly to the limit $\left\{\bar{\mathbf{Q}}(t)\right\}_{t\geq 0}$, where $(\bar{Q}_1(t), \bar{Q}_2(t), \ldots)$ is the unique solution to the following system of SDEs*

$$\begin{aligned} \mathrm{d}\bar{Q}_1(t) &= \sqrt{2}\mathrm{d}W(t) - \beta \mathrm{d}t - \bar{Q}_1(t)\mathrm{d}t + \bar{Q}_2(t)\mathrm{d}t - \mathrm{d}U_1(t), \\ \mathrm{d}\bar{Q}_2(t) &= \mathrm{d}U_1(t) - (\bar{Q}_2(t) - \bar{Q}_3(t))\mathrm{d}t, \\ \mathrm{d}\bar{Q}_i(t) &= -(\bar{Q}_i(t) - \bar{Q}_{i+1}(t))\mathrm{d}t, \quad i \geq 3, \end{aligned} \tag{3.7}$$

for $t \geq 0$, where $W(\cdot)$ is the standard Brownian motion and $U_1(\cdot)$ is the unique nondecreasing nonnegative process satisfying $\int_0^\infty \mathbb{1}_{[\bar{Q}_1(t)<0]}\mathrm{d}U_1(t) = 0$.

The above diffusion limit implies that the mean waiting time under the JSQ policy is of a similar order $O(1/\sqrt{N})$ as in the corresponding centralized M/M/N queue. Hence, we conclude that despite the distributed queueing operation a suitable load balancing policy can deliver a similar combination of excellent service quality and high resource utilization in the Halfin-Whitt regime (2.1) as in a centralized queueing arrangement. It it important though to observe a subtle but fundamental difference in the distributional properties due to the distributed versus centralized queueing operation. In the ordinary M/M/N queue a fraction $\Pi_W^\star(\beta)$ of the customers incur a non-zero waiting time as $N \to \infty$, but a non-zero waiting time is only of length $1/(\beta\sqrt{N})$ in expectation. In contrast, under the JSQ policy, the fraction of tasks that experience a non-zero waiting time is only of the order $O(1/\sqrt{N})$. However, such tasks will have to wait for the duration of a residual service time, yielding a waiting time of the order $O(1)$.

3.4 Heavy-traffic limits for JSQ(d) policies

Finally, we briefly discuss the behavior of JSQ(d) policies for fixed d in a heavy-traffic regime where $(N - \lambda(N))/\eta(N) \to \beta > 0$ as $N \to \infty$ with $\eta(N)$ a positive function diverging to infinity. Note that the case $\eta(N) = \sqrt{N}$ corresponds to the Halfin-Whitt heavy-traffic regime (2.1). While a complete characterization of the occupancy process for fixed d has remained elusive so far, significant partial results were recently obtained by Eschenfeldt & Gamarnik [11]. In order to describe the transient asymptotics, we introduce the following rescaled processes $\bar{Q}_i^N(t) := (N - Q_i^N(t))/\eta(N)$, $i = 1, 2, \ldots$.

Then, *for suitable initial states, on any finite time interval, $\{\bar{\mathbf{Q}}^N(t)\}_{t\geq 0}$ converges weakly to a deterministic system $\{\bar{\mathbf{Q}}(t)\}_{t\geq 0}$ that satisfies the following system of ODEs*

$$\frac{\mathrm{d}\bar{Q}_i(t)}{\mathrm{d}t} = -d[\bar{Q}_i(t) - \bar{Q}_{i-1}(t)] - [\bar{Q}_i(t) - \bar{Q}_{i+1}(t)], \quad i = 1, 2, \ldots, \tag{3.8}$$

with the convention that $\bar{Q}_0(t) \equiv 0$.

It is noteworthy that the scaled occupancy process loses its diffusive behavior for fixed d. It is further shown in [11] that with high probability the steady-state fraction of queues with length at least $\log_d(N/\eta(N)) - \omega(1)$ tasks approaches unity, which in turn implies that with high probability the steady-state delay is *at least* $\log_d(N/\eta(N)) - O(1)$ as $N \to \infty$. The diffusion approximation of the JSQ(d) policy in the Halfin-Whitt regime (2.1), starting from a different initial scaling, has been studied by Budhiraja & Friedlander [8]. Recently, Ying [47] introduced a broad framework involving Stein's method to analyze the rate of convergence of the scaled steady-state occupancy process of the JSQ(2) policy when $\eta(N) = N^\alpha$ with $\alpha > 0.8$. The results in [47] establish that in steady state, most of the queues are of size $\log_2(N/\eta(N)) + O(1)$, and thus the steady-state delay is of order $\log_2(N/\eta(N))$.

3.5 Universality properties

We now further explore the trade-off between delay performance and communication overhead as a function of the diversity parameter d, in conjunction with the relative load. The latter trade-off will be examined in an asymptotic regime where not only the total task arrival rate $\lambda(N)$ grows with N, but also the diversity parameter depends on N, and we write $d(N)$, to explicitly reflect that. We will specifically investigate what growth rate of $d(N)$ is required, depending on the scaling behavior of $\lambda(N)$, in order to asymptotically match the optimal performance of the JSQ policy and achieve a zero mean waiting time in the limit. The results presented in this subsection are based on [31], unless specified otherwise.

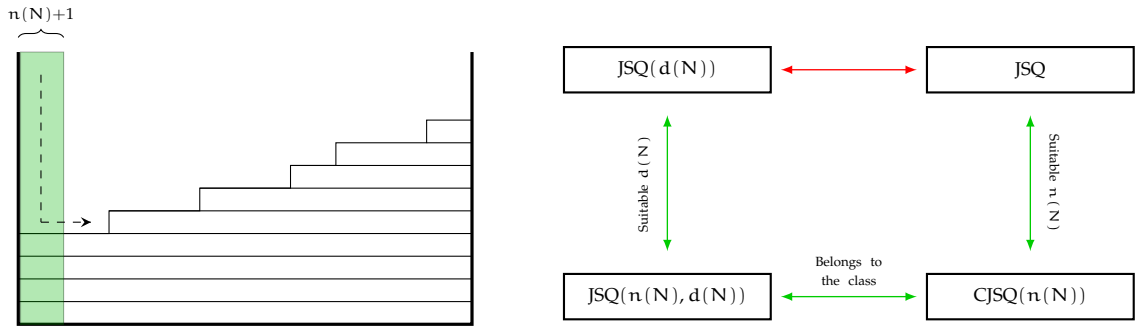
Theorem 3.1. (Universality fluid limit for JSQ($d(N)$)) *If* $d(N) \to \infty$ *as* $N \to \infty$*, then the fluid limit of the JSQ*$(d(N))$ *scheme coincides with that of the ordinary JSQ policy given by the dynamical system in* (3.5)*. Consequently, the stationary occupancy states converge to the unique fixed point in* (3.6)*.*

Theorem 3.2. (Universality diffusion limit for JSQ($d(N)$)) *If* $d(N)/(\sqrt{N}\log N) \to \infty$*, then for suitable initial conditions the weak limit of the sequence of processes* $\left\{\bar{\mathbf{Q}}^{d(N)}(t)\right\}_{t\geq 0}$ *coincides with that of the ordinary JSQ policy, and in particular, is given by the system of SDEs in* (3.7)*.*

The above universality properties indicate that the JSQ overhead can be lowered by almost a factor O(N) and O($\sqrt{N}/\log N$) while retaining fluid- and diffusion-level optimality, respectively. In other words, Theorems 3.1 and 3.2 thus reveal that it is sufficient for $d(N)$ to grow at any rate and faster than $\sqrt{N}\log N$ in order to observe similar scaling benefits as in a corresponding centralized M/M/N queue on fluid scale and diffusion scale, respectively. The stated conditions are in fact close to necessary, in the sense that if $d(N)$ is uniformly bounded and $d(N)/(\sqrt{N}\log N) \to 0$ as $N \to \infty$, then the fluid-limit and diffusion-limit paths of the system occupancy process under the JSQ($d(N)$) scheme differ from those under the ordinary JSQ policy, respectively. In particular, if $d(N)$ is uniformly bounded, the mean steady-state delay does not vanish asymptotically as $N \to \infty$.

High-level proof idea. The proofs of both Theorems 3.1 and 3.2 rely on a stochastic coupling construction to bound the difference in the queue length processes between the JSQ policy and a scheme with an arbitrary value of $d(N)$. This S-coupling ('S' stands for server-based) is then exploited to obtain the fluid and diffusion limits of the JSQ($d(N)$) policy under the conditions stated in Theorems 3.1 and 3.2.

A direct comparison between the JSQ($d(N)$) scheme and the ordinary JSQ policy is not straightforward, which is why the CJSQ($n(N)$) class of schemes is introduced as an intermediate scenario to establish the universality result. Just like the JSQ($d(N)$) scheme, the schemes in the class CJSQ($n(N)$) may be thought of as "sloppy" versions of the JSQ policy, in the sense that tasks are not necessarily assigned to a server with the shortest queue length but to one of the $n(N)+1$ lowest ordered servers, as graphically illustrated in Figure 4a. In particular, for $n(N) = 0$, the class only includes the ordinary JSQ policy. Note that the JSQ($d(N)$) scheme is guaranteed to identify the lowest ordered server, but only among a randomly sampled subset of $d(N)$ servers. In contrast, a scheme in the CJSQ($n(N)$) class only



(a) CJSQ$(n(N))$ scheme

(b) Asymptotic equivalence relations

Figure 4: (a) High-level view of the CJSQ$(n(N))$ class of schemes, where as in Figure 3, the servers are arranged in nondecreasing order of their queue lengths, and the arrival must be assigned through the left tunnel. (b) The equivalence structure is depicted for various intermediate load balancing schemes to facilitate the comparison between the JSQ$(d(N))$ scheme and the ordinary JSQ policy.

guarantees that one of the $n(N)+1$ lowest ordered servers is selected, but across the entire pool of N servers. It may be shown that for sufficiently small $n(N)$, any scheme from the class CJSQ$(n(N))$ is still 'close' to the ordinary JSQ policy. It can further be proved that for sufficiently large $d(N)$ relative to $n(N)$ we can construct a scheme called JSQ$(n(N), d(N))$, belonging to the CJSQ$(n(N))$ class, which differs 'negligibly' from the JSQ$(d(N))$ scheme. Therefore, for a 'suitable' choice of $d(N)$ the idea is to produce a 'suitable' $n(N)$. This proof strategy is schematically represented in Figure 4b.

In order to prove the stochastic comparisons among the various schemes, the many-server system is described as an ensemble of stacks, in a way that two different ensembles can be ordered. This stack formulation has also been considered in the literature for establishing the stochastic optimality properties of the JSQ policy [36, 39, 40]. However, it is only through the stack arguments developed in [31] that the comparison results can be extended to any scheme from the class CJSQ.

4 Blocking and infinite-server dynamics

The basic scenario that we have focused on so far involved single-server queues. In this section we turn attention to a system with parallel server pools, each with B servers, where B can possibly be infinite. As before, tasks must immediately be forwarded to one of the server pools, but also directly start execution or be discarded otherwise. The execution times are assumed to be exponentially distributed, and do not depend on the number of other tasks receiving service simultaneously. The current scenario will be referred to as 'infinite-server dynamics', in contrast to the earlier single-server queueing dynamics.

As it turns out, the JSQ policy has similar stochastic optimality properties as in the case of single-server queues, and in particular stochastically minimizes the cumulative number of discarded tasks [19, 24, 25, 35]. However, the JSQ policy also suffers from a similar scalability issue due to the excessive communication overhead in large-scale systems, which can be mitigated through JSQ(d) policies. Results of Turner [41] and recent papers by Karthik *et al.* [20], Mukhopadhyay *et al.* [32, 33], and Xie *et al.* [46] indicate that JSQ(d) policies provide similar "power-of-choice" gains for loss probabilities. It may be shown though that the optimal performance of the JSQ policy cannot be matched for any fixed value of d.

Motivated by these observations, we explore the trade-off between performance and communication overhead for infinite-server dynamics. We will demonstrate that the optimal performance of the JSQ policy can be asymptotically retained while drastically reducing the communication burden, mirroring the universality properties described in Section 3.5 for single-server queues. The results presented in the

remainder of the section are extracted from [29], unless indicated otherwise.

4.1 Fluid limit for JSQ policy

As in Subsection 3.2, for any fluid state $\mathbf{q} \in \mathcal{S}$, denote by $m(\mathbf{q}) = \min\{i : q_{i+1} < 1\}$ the minimum queue length among all servers. Now if $m(\mathbf{q}) = 0$, then define $p_0(m(\mathbf{q})) = 1$ and $p_i(m(\mathbf{q})) = 0$ for all $i = 1, 2, \ldots$. Otherwise, in case $m(\mathbf{q}) > 0$, define

$$p_i(\mathbf{q}) = \begin{cases} \min\left\{m(\mathbf{q})(1 - q_{m(\mathbf{q})+1})/\lambda, 1\right\} & \text{for} \quad i = m(\mathbf{q}) - 1, \\ 1 - p_{m(\mathbf{q})-1}(\mathbf{q}) & \text{for} \quad i = m(\mathbf{q}), \end{cases} \tag{4.1}$$

and $p_i(\mathbf{q}) = 0$ otherwise. As before, the coefficient $p_i(\mathbf{q})$ represents the instantaneous fraction of incoming tasks assigned to servers with a queue length of exactly i in the fluid state $\mathbf{q} \in \mathcal{S}$.

Any weak limit of the sequence of processes $\{\mathbf{q}^N(t)\}_{t\geq 0}$ is given by the deterministic system $\{\mathbf{q}(t)\}_{t\geq 0}$ satisfying the following of differential equations

$$\frac{\mathrm{d}^+ q_i(t)}{\mathrm{d}t} = \lambda p_{i-1}(\mathbf{q}(t)) - i(q_i(t) - q_{i+1}(t)), \quad i = 1, 2, \ldots, \tag{4.2}$$

where d^+/dt denotes the right-derivative.

Equations (4.1) and (4.2) are to be contrasted with Equations (3.4) and (3.5). While the form of (4.1) and the evolution equations (4.2) of the limiting dynamical system remains similar to that of (3.4) and (3.5), respectively, an additional factor $m(\mathbf{q})$ appears in (4.1) and the rate of decrease in (4.2) now becomes $i(q_i - q_{i+1})$, reflecting the infinite-server dynamics.

Let $K := \lfloor \lambda \rfloor$ and $f := \lambda - K$ denote the integral and fractional parts of λ, respectively. It is easily verified that, assuming $\lambda < B$, the unique fixed point of the dynamical system in (4.2) is given by

$$q_i^\star = \begin{cases} 1 & i = 1, \ldots, K \\ f & i = K + 1 \\ 0 & i = K + 2, \ldots, B, \end{cases} \tag{4.3}$$

and thus $\sum_{i=1}^{B} q_i^\star = \lambda$. This is consistent with the results in Mukhopadhyay *et al.* [32, 33] and Xie *et al.* [46] for fixed d, where taking $d \to \infty$ yields the same fixed point. The fixed point in (4.3), in conjunction with an interchange of limits argument, indicates that in stationarity the fraction of server pools with at least $K + 2$ and at most $K - 1$ active tasks is negligible as $N \to \infty$.

4.2 Diffusion limit for JSQ policy

As it turns out, the diffusion-limit results may be qualitatively different, depending on whether $f = 0$ or $f > 0$, and we will distinguish between these two cases accordingly. Observe that for any assignment scheme, in the absence of overflow events, the total number of active tasks evolves as the number of jobs in an M/M/∞ system, for which the diffusion limit is well-known. For the JSQ policy, it can be established that the total number of server pools with $K - 2$ or less and $K + 2$ or more tasks is negligible on the diffusion scale. If $f > 0$, the number of server pools with $K - 1$ tasks is negligible as well, and the dynamics of the number of server pools with K or $K + 1$ tasks can then be derived from the known diffusion limit of the total number of tasks mentioned above. In contrast, if $f = 0$, the number of server pools with $K - 1$ tasks is not negligible on the diffusion scale, and the limiting behavior is qualitatively different, but can still be characterized. We refer to [29] for further details.

4.3 Universality of JSQ(d) policies in infinite-server scenario

As in Subsection 3.5, we now further explore the trade-off between performance and communication overhead as a function of the diversity parameter $d(N)$, in conjunction with the relative load. We will specifically investigate what growth rate of $d(N)$ is required, depending on the scaling behavior of $\lambda(N)$, in order to asymptotically match the optimal performance of the JSQ policy.

Theorem 4.1. (Universality fluid limit for JSQ($d(N)$)) *If $d(N) \to \infty$ as $N \to \infty$, then the fluid limit of the JSQ($d(N)$) scheme coincides with that of the ordinary JSQ policy given by the dynamical system in* (4.2)*. Consequently, the stationary occupancy states converge to the unique fixed point in* (4.3)*.*

In order to state the universality result on diffusion scale, define in case $f > 0$, $f(N) := \lambda(N) - K(N)$,

$$\bar{Q}_i^{d(N)}(t) := \frac{N - Q_i^{d(N)}(t)}{\sqrt{N}} \ (i \leq K),\ \bar{Q}_{K+1}^{d(N)}(t) := \frac{Q_{K+1}^{d(N)}(t) - f(N)}{\sqrt{N}},\ \bar{Q}_i^{d(N)}(t) := \frac{Q_i^{d(N)}(t)}{\sqrt{N}} \geq 0 \ (i \geq K+2),$$

and otherwise, if $f = 0$, assume $(KN - \lambda(N))/\sqrt{N} \to \beta \in \mathbb{R}$ as $N \to \infty$, and define

$$\hat{Q}_{K-1}^{d(N)}(t) := \sum_{i=1}^{K-1} \frac{N - Q_i^{d(N)}(t)}{\sqrt{N}},\ \hat{Q}_K^{d(N)}(t) := \frac{N - Q_K^{d(N)}(t)}{\sqrt{N}},\ \hat{Q}_i^{d(N)}(t) := \frac{Q_i^{d(N)}(t)}{\sqrt{N}} \geq 0 \ (i \geq K+1).$$

Theorem 4.2 (Universality diffusion limit for JSQ($d(N)$))**.** *Assume $d(N)/(\sqrt{N}\log N) \to \infty$. Under suitable initial conditions*
(i) *If $f > 0$, then $\bar{Q}_i^{d(N)}(\cdot)$ converges to the zero process for $i \neq K+1$, and $\bar{Q}_{K+1}^{d(N)}(\cdot)$ converges weakly to the Ornstein-Uhlenbeck process satisfying the SDE $d\bar{Q}_{K+1}(t) = -\bar{Q}_{K+1}(t)dt + \sqrt{2\lambda}dW(t)$, where $W(\cdot)$ is the standard Brownian motion.*
(ii) *If $f = 0$, then $\hat{Q}_{K-1}^{d(N)}(\cdot)$ converges weakly to the zero process, and $(\hat{Q}_K^{d(N)}(\cdot), \hat{Q}_{K+1}^{d(N)}(\cdot))$ converges weakly to $(\hat{Q}_K(\cdot), \hat{Q}_{K+1}(\cdot))$, described by the unique solution of the following system of SDEs:*

$$\begin{aligned} d\hat{Q}_K(t) &= \sqrt{2K}dW(t) - (\hat{Q}_K(t) + K\hat{Q}_{K+1}(t))dt + \beta dt + dV_1(t) \\ d\hat{Q}_{K+1}(t) &= dV_1(t) - (K+1)\hat{Q}_{K+1}(t)dt, \end{aligned}$$

where $W(\cdot)$ is the standard Brownian motion, and $V_1(\cdot)$ is the unique nondecreasing process satisfying $\int_0^t \mathbb{1}_{[\hat{Q}_K(s) \geq 0]} dV_1(s) = 0$.

Given the asymptotic results for the JSQ policy in Subsections 4.1 and 4.2, the proofs of the asymptotic results for the JSQ($d(N)$) scheme in Theorems 4.1 and 4.2 involve establishing a universality result which shows that the limiting processes for the JSQ($d(N)$) scheme are '$g(N)$-alike' to those for the ordinary JSQ policy for suitably large $d(N)$. Loosely speaking, if two schemes are $g(N)$-alike, then in some sense, the associated system occupancy states are indistinguishable on $g(N)$-scale.

The next theorem states a sufficient criterion for the JSQ($d(N)$) scheme and the ordinary JSQ policy to be $g(N)$-alike, and thus, provides the key vehicle in establishing the universality result.

Theorem 4.3. *Let $g : \mathbb{N} \to \mathbb{R}_+$ be a function diverging to infinity. Then the JSQ policy and the JSQ($d(N)$) scheme are $g(N)$-alike, with $g(N) \leq N$, if*
(i) $d(N) \to \infty$ *for* $g(N) = O(N)$, (ii) $d(N)\left(\frac{N}{g(N)}\log\left(\frac{N}{g(N)}\right)\right)^{-1} \to \infty$ *for* $g(N) = o(N)$.

The proof of Theorem 4.3 relies on a novel coupling construction, called T-coupling ('T' stands for task-based), which will be used to (lower and upper) bound the difference of occupancy states of two arbitrary schemes. This T-coupling [29] is distinct from and inherently stronger than the S-coupling used in Subsection 3.5 in the single-server queueing scenario. Note that in the current infinite-server scenario, the departures of the ordered server pools cannot be coupled, mainly since the departure rate at the m^{th} ordered server pool, for some $m = 1, 2, \ldots, N$, depends on its number of active tasks. The T-coupling is also fundamentally different from the coupling constructions used in establishing the weak majorization results in [36, 39, 40, 44, 45] in the context of the ordinary JSQ policy in the single-server queueing scenario, and in [19, 24, 25, 35] in the scenario of state-dependent service rates.

5 Universality of load balancing in networks

In this section we return to the single-server queueing dynamics, and extend the universality properties to network scenarios, where the N servers are assumed to be inter-connected by some underlying graph topology G_N. Tasks arrive at the various servers as independent Poisson processes of rate λ, and each incoming task is assigned to whichever server has the smallest number of tasks among the one where it arrives and its neighbors in G_N. Thus, in case G_N is a clique, each incoming task is assigned to the server with the shortest queue across the entire system, and the behavior is equivalent to that under the JSQ policy. The stochastic optimality properties of the JSQ policy thus imply that the queue length process in a clique will be better balanced and smaller (in a majorization sense) than in an arbitrary graph G_N.

Besides the prohibitive communication overhead discussed earlier, a further scalability issue of the JSQ policy arises when executing a task involves the use of some data. Storing such data for all possible tasks on all servers will typically require an excessive amount of storage capacity. These two burdens can be effectively mitigated in sparser graph topologies where tasks that arrive at a specific server i are only allowed to be forwarded to a subset of the servers $\mathcal{N}_i$. For the tasks that arrive at server i, queue length information then only needs to be obtained from servers in $\mathcal{N}_i$, and it suffices to store replicas of the required data on the servers in $\mathcal{N}_i$. The subset $\mathcal{N}_i$ containing the peers of server i can be naturally viewed as its neighbors in some graph topology G_N. In this section we focus on the results in [28] for the case of undirected graphs, but most of the analysis can be extended to directed graphs.

The above model has been studied in [15, 41], focusing on certain fixed-degree graphs and in particular ring topologies. The results demonstrate that the flexibility to forward tasks to a few neighbors, or even just one, with possibly shorter queues significantly improves the performance in terms of the waiting time and tail distribution of the queue length. This resembles the "power-of-choice" gains observed for JSQ(d) policies in complete graphs. However, the results in [15, 41] also establish that the performance sensitively depends on the underlying graph topology, and that selecting from a fixed set of $d-1$ neighbors typically does not match the performance of re-sampling $d-1$ alternate servers for each incoming task from the entire population, as in the power-of-d scheme in a complete graph.

If tasks do not get served and never depart but simply accumulate, then the scenario described above amounts to a so-called balls-and-bins problem on a graph. Viewed from that angle, a close counterpart of our setup is studied in Kenthapadi & Panigrahy [21], where in our terminology each arriving task is routed to the shortest of $d \geq 2$ randomly selected neighboring queues.

The key challenge in the analysis of load balancing on arbitrary graph topologies is that one needs to keep track of the evolution of number of tasks at each vertex along with their corresponding neighborhood relationship. This creates a major problem in constructing a tractable Markovian state descriptor, and renders a direct analysis of such processes highly intractable. Consequently, even asymptotic results for load balancing processes on an arbitrary graph have remained scarce so far. The approach in [28] is radically different, and aims at comparing the load balancing process on an arbitrary graph with that on a clique. Specifically, rather than analyzing the behavior for a given class of graphs or degree value, the analysis explores for what types of topologies and degree properties the performance is asymptotically similar to that in a clique. The proof arguments in [28] build on the stochastic coupling constructions developed in Subsection 3.5 for JSQ(d) policies. Specifically, the load balancing process on an arbitrary graph is viewed as a 'sloppy' version of that on a clique, and several other intermediate sloppy versions are constructed.

Let $Q_i(G_N, t)$ denote the number of servers with queue length at least i at time t, $i = 1, 2, \ldots$, and let the fluid-scaled variables $q_i(G_N, t) := Q_i(G_N, t)/N$ be the corresponding fractions. Also, in the Halfin-Whitt heavy-traffic regime (2.1), define the centered and diffusion-scaled variables $\bar{Q}_1(G_N, t) := -(N - Q_1(G_N, t))/\sqrt{N}$ and $\bar{Q}_i(G_N, t) := Q_i(G_N, t)/\sqrt{N}$ for $i = 2, 3, \ldots$, analogous to (3.1).

The next definition introduces two notions of *asymptotic optimality*.

Definition 5.1 (Asymptotic optimality)**.** *A graph sequence $\mathbf{G} = \{G_N\}_{N\geq 1}$ is called 'asymptotically optimal on N-scale' or 'N-optimal', if for any $\lambda < 1$, the scaled occupancy process $(q_1(G_N, \cdot), q_2(G_N, \cdot), \ldots)$ converges weakly, on any finite time interval, to the process $(q_1(\cdot), q_2(\cdot), \ldots)$ given by* (3.5)*.*

Moreover, a graph sequence $\mathbf{G} = \{G_N\}_{N\geq 1}$ is called 'asymptotically optimal on $\sqrt{N}$-scale' or '$\sqrt{N}$-optimal', if in the Halfin-Whitt heavy-traffic regime (2.1)*, on any finite time interval, the process $(\bar{Q}_1(G_N, \cdot), \bar{Q}_2(G_N, \cdot), \ldots)$ converges weakly to the process $(\bar{Q}_1(\cdot), \bar{Q}_2(\cdot), \ldots)$ given by* (3.7)*.*

Intuitively speaking, if a graph sequence is N-optimal or $\sqrt{N}$-optimal, then in some sense, the associated occupancy processes are indistinguishable from those of the sequence of cliques on N-scale or $\sqrt{N}$-scale. In other words, on any finite time interval their occupancy processes can differ from those in cliques by at most $o(N)$ or $o(\sqrt{N})$, respectively.

5.1 Asymptotic optimality criteria for deterministic graph sequences

We now develop a criterion for asymptotic optimality of an arbitrary deterministic graph sequence on different scales. We first introduce some useful notation, and two measures of *well-connectedness*. Let $G = (V, E)$ be any graph. For a subset $U \subseteq V$, define $\textsc{com}(U) := |V \setminus N[U]|$ to be the set of all vertices that are disjoint from U, where $N[U] := U \cup \{v \in V : \exists\, u \in U \text{ with } (u, v) \in E\}$. For any fixed $\varepsilon > 0$ define

$$\textsc{dis}_1(G, \varepsilon) := \sup_{U\subseteq V, |U|\geq \varepsilon |V|} \textsc{com}(U), \qquad \textsc{dis}_2(G, \varepsilon) := \sup_{U\subseteq V, |U|\geq \varepsilon \sqrt{|V|}} \textsc{com}(U). \tag{5.1}$$

The next theorem provides sufficient conditions for asymptotic optimality on N-scale and $\sqrt{N}$-scale in terms of the above two well-connectedness measures.

Theorem 5.2. *For any graph sequence $\mathbf{G} = \{G_N\}_{N\geq 1}$,* (i) *$\mathbf{G}$ is N-optimal if for any $\varepsilon > 0$, $\textsc{dis}_1(G_N, \varepsilon)/N \to 0$ as $N \to \infty$.* (ii) *$\mathbf{G}$ is $\sqrt{N}$-optimal if for any $\varepsilon > 0$, $\textsc{dis}_2(G_N, \varepsilon)/\sqrt{N} \to 0$ as $N \to \infty$.*

The next corollary is an immediate consequence of Theorem 5.2.

Corollary 5.3. *Let $\mathbf{G} = \{G_N\}_{N\geq 1}$ be any graph sequence. Then* (i) *If $d_{\min}(G_N) = N - o(N)$, then $\mathbf{G}$ is N-optimal, and* (ii) *If $d_{\min}(G_N) = N - o(\sqrt{N})$, then $\mathbf{G}$ is $\sqrt{N}$-optimal.*

We now provide a sketch of the main proof arguments for Theorem 5.2 as used in [28], focusing on the proof of N-optimality. The proof of $\sqrt{N}$-optimality follows along similar lines. First of all, it can be established that if a system is able to assign each task to a server in the set $\mathcal{S}^N(n(N))$ of the $n(N)$ nodes with shortest queues, where $n(N)$ is $o(N)$, then it is N-optimal. Since the underlying graph is not a clique however (otherwise there is nothing to prove), for any $n(N)$ not every arriving task can be assigned to a server in $\mathcal{S}^N(n(N))$. Hence, a further stochastic comparison property is proved in [28] implying that if on any finite time interval of length t, the number of tasks $\Delta^N(t)$ that are not assigned to a server in $\mathcal{S}^N(n(N))$ is $o_P(N)$, then the system is N-optimal as well. The N-optimality can then be concluded when $\Delta^N(t)$ is $o_P(N)$, which is demonstrated in [28] under the condition that $\textsc{dis}_1(G_N, \varepsilon)/N \to 0$ as $N \to \infty$ as stated in Theorem 5.2.

5.2 Asymptotic optimality of random graph sequences

Next we investigate how the load balancing process behaves on random graph topologies. Specifically, we aim to understand what types of graphs are asymptotically optimal in the presence of randomness (i.e., in an average-case sense). Theorem 5.4 below establishes sufficient conditions for asymptotic optimality of a sequence of inhomogeneous random graphs. Recall that a graph $G' = (V', E')$ is called a supergraph of $G = (V, E)$ if $V = V'$ and $E \subseteq E'$.

Theorem 5.4. *Let $\mathbf{G} = \{G_N\}_{N\geq 1}$ be a graph sequence such that for each N, $G_N = (V_N, E_N)$ is a super-graph of the inhomogeneous random graph G'_N where any two vertices $u, v \in V_N$ share an edge with probability p^N_{uv}.*

(i) *If for each* $\varepsilon > 0$*, there exists subsets of vertices* $V_N^\varepsilon \subseteq V_N$ *with* $|V_N^\varepsilon| < \varepsilon N$*, such that* $\inf\{p_{uv}^N : u, v \in V_N^\varepsilon\}$ *is* $\omega(1/N)$*, then* $\mathbf{G}$ *is* N*-optimal.*

(ii) *If for each* $\varepsilon > 0$*, there exists subsets of vertices* $V_N^\varepsilon \subseteq V_N$ *with* $|V_N^\varepsilon| < \varepsilon\sqrt{N}$*, such that* $\inf\{p_{uv}^N : u, v \in V_N^\varepsilon\}$ *is* $\omega(\log(N)/\sqrt{N})$*, then* $\mathbf{G}$ *is* $\sqrt{N}$*-optimal.*

The proof of Theorem 5.4 relies on Theorem 5.2. Specifically, if G_N satisfies conditions (i) and (ii) in Theorem 5.4, then the corresponding conditions (i) and (ii) in Theorem 5.2 hold.

As an immediate corollary to Theorem 5.4 we obtain an optimality result for the sequence of Erdős-Rényi random graphs.

Corollary 5.5. *Let* $\mathbf{G} = \{G_N\}_{N\geq 1}$ *be a graph sequence such that for each* N*,* G_N *is a super-graph of* $\mathrm{ER}_N(p(N))$*, and* $d(N) = (N-1)p(N)$*. Then* (i) *If* $d(N) \to \infty$ *as* $N \to \infty$*, then* $\mathbf{G}$ *is* N*-optimal.* (ii) *If* $d(N)/(\sqrt{N}\log N) \to \infty$ *as* $N \to \infty$*, then* $\mathbf{G}$ *is* $\sqrt{N}$*-optimal.*

The growth rate condition for N-optimality in Corollary 5.5 (i) is not only sufficient, but necessary as well. Thus informally speaking, N-optimality is achieved under the minimum condition required as long as the underlying topology is suitably random.

6 Token-based load balancing

While a zero waiting time can be achieved in the limit by sampling only $d(N) = o(N)$ servers as Sections 3.5, 4 and 5 showed, even in network scenarios, the amount of communication overhead in terms of $d(N)$ must still grow with N. As mentioned earlier, this can be avoided by introducing memory at the dispatcher, in particular maintaining a record of only vacant servers, and assigning tasks to idle servers, if there are any, or to a uniformly at random selected server otherwise. This so-called Join-the-Idle-Queue (JIQ) scheme [5, 22] can be implemented through a simple token-based mechanism generating at most one message per task. Remarkably enough, even with such low communication overhead, the mean waiting time and the probability of a non-zero waiting time vanish under the JIQ scheme in both the fluid and diffusion regimes, as we will discuss in the next two subsections.

6.1 Asymptotic optimality of JIQ scheme

We first consider the fluid limit of the JIQ scheme. Let $q_i^N(\infty)$ be a random variable denoting the process $q_i^N(\cdot)$ in steady state. It was proved in [37] for the JIQ scheme (under very broad conditions),

$$q_1^N(\infty) \to \lambda, \qquad q_i^N(\infty) \to 0 \quad \text{for all } i \geq 2, \qquad \text{as} \quad N \to \infty. \tag{6.1}$$

The above equation in conjunction with the PASTA property yields that the steady-state probability of a non-zero wait vanishes as $N \to \infty$, thus exhibiting asymptotic optimality of the JIQ scheme on fluid scale.

We now turn to the diffusion limit of the JIQ scheme.

Theorem 6.1. (Diffusion limit for JIQ) *In the Halfin-Whitt heavy-traffic regime* (2.1)*, under suitable initial conditions, the weak limit of the sequence of centered and diffusion-scaled occupancy process in* (3.1) *coincides with that of the ordinary JSQ policy given by the system of SDEs in* (3.7)*.*

The above theorem implies that for suitable initial states, on any finite time interval, the occupancy process under the JIQ scheme is indistinguishable from that under the JSQ policy. The proof of Theorem 6.1 relies on a coupling construction as described in greater detail in [30]. The idea is to compare the occupancy processes of two systems following JIQ and JSQ policies, respectively. Comparing the JIQ and JSQ policies is facilitated when viewed as follows: (i) If there is an idle server in the system, both JIQ and JSQ perform similarly, (ii) Also, when there is no idle server and only $O(\sqrt{N})$ servers with

queue length two, JSQ assigns the arriving task to a server with queue length one. In that case, since JIQ assigns at random, the probability that the task will land on a server with queue length two and thus JIQ acts differently than JSQ is $O(1/\sqrt{N})$. Since on any finite time interval the number of times an arrival finds all servers busy is at most $O(\sqrt{N})$, all the arrivals except an $O(1)$ of them are assigned in exactly the same manner in both JIQ and JSQ, which then leads to the same scaling limit for both policies.

6.2 Multiple dispatchers

So far we have focused on a basic scenario with a single dispatcher. Since it is not uncommon for LBAs to operate across multiple dispatchers though, we consider in this subsection a scenario with N parallel identical servers as before and $R \geq 1$ dispatchers. (We will assume the number of dispatchers to remain fixed as the number of servers grows large, but a further natural scenario would be for the number of dispatchers $R(N)$ to scale with the number of servers as considered by Mitzenmacher [27], who analyzes the case $R(N) = rN$ for some constant r, so that the relative load of each dispatcher is λr.) Tasks arrive at dispatcher r as a Poisson process of rate $\alpha_r \lambda N$, with $\alpha_r > 0$, $r = 1, \ldots, R$, $\sum_{r=1}^{R} \alpha_r = 1$, and λ denoting the task arrival rate per server. For conciseness, we denote $\alpha = (\alpha_1, \ldots, \alpha_R)$, and without loss of generality we assume that the dispatchers are indexed such that $\alpha_1 \geq \alpha_2 \geq \cdots \geq \alpha_R$.

When a server becomes idle, it sends a token to one of the dispatchers selected uniformly at random, advertising its availability. When a task arrives at a dispatcher which has tokens available, one of the tokens is selected, and the task is immediately forwarded to the corresponding server.

We distinguish two scenarios when a task arrives at a dispatcher which has no tokens available, referred to as the *blocking* and *queueing* scenario respectively. In the blocking scenario, the incoming task is blocked and instantly discarded. In the queueing scenario, the arriving task is forwarded to one of the servers selected uniformly at random. If the selected server happens to be idle, then the outstanding token at one of the other dispatchers is revoked.

In the queueing scenario we assume $\lambda < 1$, which is not only necessary but also sufficient for stability. Denote by $B(R, N, \lambda, \alpha)$ the steady-state blocking probability of an arbitrary task in the blocking scenario. Also, denote by $W(R, N, \lambda, \alpha)$ a random variable with the steady-state waiting-time distribution of an arbitrary task in the queueing scenario.

Scenarios with multiple dispatchers have received limited attention in the literature, and the scant papers that exist [22, 27, 38] almost exclusively assume that the loads at the various dispatchers are strictly equal. In these cases the fluid limit, for suitable initial states, is the same as that for a single dispatcher, and in particular the fixed point is the same, hence, the JIQ scheme continues to achieve asymptotically optimal delay performance with minimal communication overhead. As one of the few exceptions, [7] allows the loads at the various dispatchers to be different.

Results for blocking scenario. For the blocking scenario, it is established in [7] that,

$$B(R, N, \lambda, \alpha) \to \max\{1 - R\alpha_R, 1 - 1/\lambda\} \quad \text{as } N \to \infty.$$

This result shows that in the many-server limit the system performance in terms of blocking is either determined by the relative load of the least-loaded dispatcher, or by the aggregate load. This indirectly reveals that, somewhat counter-intuitively, it is the least-loaded dispatcher that throttles tokens and leaves idle servers stranded, thus acting as bottleneck.

Results for queueing scenario. For the queueing scenario, it is shown in [7] that, for fixed $\lambda < 1$

$$\mathbb{E}[W(R, N, \lambda, \alpha)] \to \frac{\lambda_2(R, \lambda, \alpha)}{1 - \lambda_2(R, \lambda, \alpha)} \quad \text{as } N \to \infty,$$

where $\lambda_2(R,\lambda,\alpha) = 1 - \frac{1-\lambda\sum_{i=1}^{r^\star}\alpha_i}{1-\lambda r^\star/R}$, with $r^\star = \sup\left\{r\middle|\alpha_r > \frac{1}{R}\frac{1-\lambda\sum_{i=1}^{r}\alpha_i}{1-\lambda r/R}\right\}$, may be interpreted as the rate at which tasks are forwarded to randomly selected servers.

When the arrival rates at all dispatchers are strictly equal, i.e., $\alpha_1 = \cdots = \alpha_R = 1/R$, the above results indicate that the stationary blocking probability and the mean waiting time asymptotically vanish as $N \to \infty$, which is in agreement with the observations in [38] mentioned above. However, when the arrival rates at the various dispatchers are not perfectly equal, so that $\alpha_R < 1/R$, the blocking probability and mean waiting time are strictly positive in the limit, even for arbitrarily low overall load and an arbitrarily small degree of skewness in the arrival rates. Thus, the ordinary JIQ scheme fails to achieve asymptotically optimal performance for heterogeneous dispatcher loads.

In order to counter the above-described performance degradation for asymmetric dispatcher loads, [7] proposes two enhancements. Enhancement A uses a non-uniform token allotment: When a server becomes idle, it sends a token to dispatcher r with probability β_r. Enhancement B involves a token exchange mechanism: Any token is transferred to a uniformly randomly selected dispatcher at rate ν. Note that the token exchange mechanism only creates a constant communication overhead per task as long as the rate ν does not depend on the number of servers N, and thus preserves the scalability of the basic JIQ scheme.

The above enhancements can achieve asymptotically optimal performance for suitable values of the β_r parameters and the exchange rate ν. Specifically, the stationary blocking probability in the blocking scenario and the mean waiting time in the queueing scenario asymptotically vanish as $N \to \infty$, upon using Enhancement A with $\beta_r = \alpha_r$ or Enhancement B with $\nu \geq \frac{\lambda}{1-\lambda}(\alpha_1 R - 1)$.

7 Redundancy policies and alternative scaling regimes

In this section we discuss somewhat related redundancy policies and alternative scaling regimes and performance metrics.

Redundancy-d policies. So-called redundancy-d policies involve a somewhat similar operation as JSQ(d) policies, and also share the primary objective of ensuring low delays [1, 42]. In a redundancy-d policy, $d \geq 2$ candidate servers are selected uniformly at random (with or without replacement) for each arriving task, just like in a JSQ(d) policy. Rather than forwarding the task to the server with the shortest queue however, replicas are dispatched to all sampled servers.

Two common options can be distinguished for abortion of redundant clones. In the first variant, as soon as the first replica starts service, the other clones are abandoned. In this case, a task gets executed by the server which had the smallest workload at the time of arrival (and which may or may not have had the shortest queue length) among the sampled servers. This may be interpreted as a power-of-d version of the Join-the-Smallest Workload (JSW) policy discussed in Subsection 2.5. In the second option the other clones of the task are not aborted until the first replica has completed service (which may or may not have been the first replica to start service). While a task is only handled by one of the servers in the former case, it may be processed by several servers in the latter case.

Conventional heavy traffic. It is also worth mentioning some asymptotic results for the classical heavy-traffic regime as described in Subsection 2.2 where the number of servers N is fixed and the relative load tends to one in the limit. The papers [12, 34, 48] establish diffusion limits for the JSQ policy in a sequence of systems with Markovian characteristics as in our basic model set-up, but where in the K-th system the arrival rate is $K\lambda + \hat{\lambda}\sqrt{K}$, while the service rate of the i-th server is $K\mu_i + \hat{\mu}_i\sqrt{K}$, $i = 1, \ldots, N$, with $\lambda = \sum_{i=1}^{N}\mu_i$, inducing critical load as $K \to \infty$. It is proved that for suitable initial conditions the queue lengths are of the order O($\sqrt{K}$) over any finite time interval and exhibit a state-space collapse property.

Atar *et al.* [3] investigate a similar scenario, and establish diffusion limits for three policies: the JSQ(d) policy, the redundancy-d policy (where the redundant clones are abandoned as soon as the first replica starts service), and a combined policy called Replicate-to-Shortest-Queues (RSQ) where d replicas are dispatched to the d-shortest queues.

Non-degenerate slowdown. Asymptotic results for the so-called non-degenerate slow-down regime described in Subsection 2.2 where $N - \lambda(N) \to \gamma > 0$ as the number of servers N grows large, are scarce. Gupta & Walton [16] characterize the diffusion-scaled queue length process under the JSQ policy in this asymptotic regime. They further compare the diffusion limit for the JSQ policy with that for a centralized queue as described above as well as several LBAs such as the JIQ scheme and a refined version called Idle-One-First (I1F), where a task is assigned to a server with exactly one task if no idle server is available and to a randomly selected server otherwise.

It is proved that the diffusion limit for the JIQ scheme is no longer asymptotically equivalent to that for the JSQ policy in this asymptotic regime, and the JIQ scheme fails to achieve asymptotic optimality in that respect, as opposed to the behavior in the large-capacity and Halfin-Whitt regimes discussed in Subsection 2.7. In contrast, the I1F scheme does preserve the asymptotic equivalence with the JSQ policy in terms of the diffusion-scaled queue length process, and thus retains asymptotic optimality in that sense.

Sparse-feedback regime. As described in Section 2.7, the JIQ scheme involves a communication overhead of at most one message per task, and yet achieves optimal delay performance in the fluid and diffusion regimes. However, even just one message per task may still be prohibitive, especially when tasks do not involve big computational tasks, but small data packets which require little processing.

Motivated by the above issues, [6] proposes a novel class of LBAs which also leverage memory at the dispatcher, but allow the communication overhead to be seamlessly adapted and reduced below that of the JIQ scheme. Specifically, in the proposed schemes, the various servers provide occasional queue status notifications to the dispatcher, either in a synchronous or asynchronous fashion. The dispatcher uses these reports to maintain queue estimates, and forwards incoming tasks to the server with the lowest queue estimate. The results in [6] demonstrate that the proposed schemes markedly outperform JSQ(d) policies with the same number of $d \geq 1$ messages per task and they can achieve a vanishing waiting time in the limit when the update frequency exceeds $\lambda/(1-\lambda)$. In case servers only report zero queue lengths and suppress updates for non-zero queues, the update frequency required for a vanishing waiting time can in fact be lowered to just λ, matching the one message per task involved in the JIQ scheme.

Scaling of maximum queue length. So far we have focused on the asymptotic behavior of LBAs in terms of the number of servers with a certain queue length, either on fluid scale or diffusion scale, in various regimes as $N \to \infty$. A related but different performance metric is the maximum queue length $M(N)$ among all servers as $N \to \infty$. Luczak & McDiarmid [23] showed that for fixed $d \geq 2$ the steady-state maximum queue length $M(N)$ under the JSQ(d) policy is given by $\log(\log(N))/\log(d) + O(1)$ and is concentrated on at most two adjacent values, whereas for purely random assignment ($d = 1$), it scales as $\log(N)/\log(1/\lambda)$ and does not concentrate on a bounded range of values. This is yet a further manifestation of the "power-of-choice" effect.

The maximum queue length $M(N)$ is the central performance metric in balls-and-bins models where arriving items (balls) do not get served and never depart but simply accumulate in bins, and (stationary) queue lengths are not meaningful. In fact, the very notion of randomized load balancing and power-of-d strategies was introduced in a balls-and-bins setting in the seminal paper by Azar *et al.* [4].

References

[1] G. Ananthanarayanan, A. Ghodsi, S. Shenker, and I. Stoica. Effective straggler mitigation: Attack of the clones. In *NSDI '13*, pages 185–198, 2013.

[2] R. Atar. A diffusion regime with nondegenerate slowdown. *Oper. Res.*, 60(2):490–500, 2012.

[3] R. Atar, I. Keslassy, and G. Mendelson. Randomized load balancing in heavy traffic. *Preprint*.

[4] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal. Balanced allocations. In *Proc. STOC '94*, pages 593–602, 1994.

[5] R. Badonnel and M. Burgess. Dynamic pull-based load balancing for autonomic servers. In *Proc. IEEE/IFIP*, pages 751–754, 2008.

[6] M. van der Boor, S. C. Borst, and J. S. H. van Leeuwaarden. Hyper-scalable JSQ with sparse feedback. *Preprint*, 2017.

[7] M. van der Boor, S. C. Borst, and J. S. H. van Leeuwaarden. Load balancing in large-scale systems with multiple dispatchers. In *Proc. INFOCOM '17*, 2017.

[8] A. Budhiraja and E. Friedlander. Diffusion approximations for load balancing mechanisms in cloud storage systems. *arXiv:1706.09914*, 2017.

[9] A. Ephremides, P. Varaiya, and J. Walrand. A simple dynamic routing problem. *IEEE Trans. Autom. Control*, 25(4):690–693, 1980.

[10] P. Eschenfeldt and D. Gamarnik. Join the shortest queue with many servers. The heavy traffic asymptotics. *arXiv:1502.00999*, 2015.

[11] P. Eschenfeldt and D. Gamarnik. Supermarket queueing system in the heavy traffic regime. Short queue dynamics. *arXiv: 1610.03522*, 2016.

[12] G. Foschini and J. Salz. A basic dynamic routing problem and diffusion. *IEEE Trans. Commun.*, 26(3):320–327, 1978.

[13] S. G. Foss and N. I. Chernova. On optimality of the FCFS discipline in multiserver queueing systems and networks. *Siberian Math. J.*, 42(2):372–385, 2001.

[14] D. Gamarnik, J. Tsitsiklis, and M. Zubeldia. Delay, memory and messaging tradeoffs in distributed service systems. In *Proc. SIGMETRICS '16*, pages 1–12, 2016.

[15] N. Gast. The power of two choices on graphs: the pair-approximation is accurate. In *MAMA workshop '15*, 2015.

[16] V. Gupta and N. Walton. Load balancing in the non-degenerate slowdown regime. *arXiv:1707.01969*, 2017.

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

[18] P. Hunt and T. Kurtz. Large loss networks. *Stoch. Proc. Appl.*, 53(2):363–378, 1994.

[19] P. K. Johri. Optimality of the shortest line discipline with state-dependent service rates. *Eur. J. Oper. Res.*, 41(2):157–161, 1989.

[20] A. Karthik, A. Mukhopadhyay, and R. R. Mazumdar. Choosing among heterogeneous server clouds. *Queueing Syst.*, 85(1):1–29, 2017.

[21] K. Kenthapadi and R. Panigrahy. Balanced allocation on graphs. In *Proc. SODA '06*, pages 434–443, 2006.

[22] Y. Lu, Q. Xie, G. Kliot, A. Geller, J. R. Larus, and A. Greenberg. Join-idle-queue: a novel load balancing algorithm for dynamically scalable web services. In *Perf. Eval.*, volume 68, pages 1056–1071, 2011.

[23] M. J. Luczak and C. McDiarmid. On the maximum queue length in the supermarket model. *Ann. Prob.*, 34(2):493–527, 2006.

[24] R. Menich. Optimality of shortest queue routing for dependent service stations. In *Proc. CDC '87*, pages 1069–1072, 1987.

[25] R. Menich and R. F. Serfozo. Optimality of routing and servicing in dependent parallel processing systems. *Queueing Syst.*, 9(4):403–418, 1991.

[26] M. Mitzenmacher. The power of two choices in randomized load balancing. *IEEE Trans. Parallel Distrib. Syst.*, 12(10):1094–1104, 2001.

[27] M. Mitzenmacher. Analyzing distributed Join-Idle-Queue: A fluid limit approach. In *Proc. Allerton '16*, pages 312–318, 2016.

[28] D. Mukherjee, S. C. Borst, and J. S. H. van Leeuwaarden. Asymptotically optimal load balancing topologies. *arXiv:1707.05866*, 2017.

[29] D. Mukherjee, S. C. Borst, J. S. H. van Leeuwaarden, and P. A. Whiting. Asymptotic optimality of power-of-d load balancing in large-scale systems. *arXiv:1612.00722*, 2016.

[30] D. Mukherjee, S. C. Borst, J. S. H. van Leeuwaarden, and P. A. Whiting. Universality of load balancing schemes on the diffusion scale. *J. Appl. Probab.*, 53(4), 2016.

[31] D. Mukherjee, S. C. Borst, J. S. H. van Leeuwaarden, and P. A. Whiting. Universality of power-of-d load

balancing in many-server systems. *arXiv:1612.00723*, 2016.

[32] A. Mukhopadhyay, A. Karthik, R. R. Mazumdar, and F. Guillemin. Mean field and propagation of chaos in multi-class heterogeneous loss models. *Perf. Eval.*, 91:117–131, 2015.

[33] A. Mukhopadhyay, R. R. Mazumdar, and F. Guillemin. The power of randomized routing in heterogeneous loss systems. In *Proc. ITC '15*, pages 125–133, 2015.

[34] M. I. Reiman. Some diffusion approximations with state space collapse. In *Modelling and performance evaluation methodology*, pages 207–240. 1984.

[35] P. D. Sparaggis, D. Towsley, and C. G. Cassandras. Extremal properties of the shortest/longest non-full queue policies in finite-capacity systems with state-dependent service rates. *J. Appl. Probab.*, 30(1):223–236, 1993.

[36] P. D. Sparaggis, D. Towsley, and C. G. Cassandras. Sample path criteria for weak majorization. *Adv. Appl. Probab.*, 26(1):155–171, 1994.

[37] A. L. Stolyar. Pull-based load distribution in large-scale heterogeneous service systems. *Queueing Syst.*, 80(4):341–361, 2015.

[38] A. L. Stolyar. Pull-based load distribution among heterogeneous parallel servers: the case of multiple routers. *Queueing Syst.*, 85(1):31–65, 2017.

[39] D. Towsley. Application of majorization to control problems in queueing systems. In P. Chrétienne, E. G. Coffman, J. K. Lenstra, and Z. Liu, editors, *Scheduling Theory and its Applications*, chapter 14. John Wiley & Sons, Chichester, 1995.

[40] D. Towsley, P. Sparaggis, and C. Cassandras. Optimal routing and buffer allocation for a class of finite capacity queueing systems. *IEEE Trans. Autom. Control*, 37(9):1446–1451, 1992.

[41] S. R. Turner. The effect of increasing routing choice on resource pooling. *Probab. Eng. Inf. Sci.*, 12(01):109, 1998.

[42] A. Vulimiri, P. B. Godfrey, R. Mittal, J. Sherry, S. Ratnasamy, and S. Shenker. Low latency via redundancy. In *Proc. CoNEXT '13*, pages 283–294, 2013.

[43] N. D. Vvedenskaya, R. L. Dobrushin, and F. I. Karpelevich. Queueing system with selection of the shortest of two queues: An asymptotic approach. *Problemy Peredachi Informatsii*, 32(1):20–34, 1996.

[44] R. R. Weber. On the optimal assignment of customers to parallel servers. *J. Appl. Probab.*, 15(2):406–413, 1978.

[45] W. Winston. Optimality of the shortest line discipline. *J. Appl. Probab.*, 14(1):181–189, 1977.

[46] Q. Xie, X. Dong, Y. Lu, and R. Srikant. Power of d choices for large-scale bin packing. In *Proc. SIGMETRICS '15*, pages 321–334, 2015.

[47] L. Ying. Stein's method for mean field approximations in light and heavy traffic regimes. *Proc. ACM Meas. Anal. Comput. Syst.*, 1(1):12, 2017.

[48] H. Zhang, G.-H. Hsu, and R. Wang. Heavy traffic limit theorems for a sequence of shortest queueing systems. *Queueing Syst.*, 21(1):217–238, 1995.