Delay, Memory, and Messaging Tradeoffs in Distributed Service Systems

David Gamarnik
MIT, ORC
77 Massachusetts Ave.
Cambridge, MA 02139
gamarnik@mit.edu

John N. Tsitsiklis
MIT, LIDS
77 Massachusetts Ave.
Cambridge, MA 02139
jnt@mit.edu

Martin Zubeldia
MIT, LIDS
77 Massachusetts Ave.
Cambridge, MA 02139
zubeldia@mit.edu

ABSTRACT

We consider the following distributed service model: jobs with unit mean, exponentially distributed, and independent processing times arrive as a Poisson process of rate λN, with $0 < \lambda < 1$, and are immediately dispatched to one of several queues associated with N identical servers with unit processing rate. We assume that the dispatching decisions are made by a central dispatcher endowed with a finite memory, and with the ability to exchange messages with the servers.

We study the fundamental resource requirements (memory bits and message exchange rate), in order to drive the expected steady-state queueing delay of a typical job to zero, as N increases. We propose a certain policy and establish (using a fluid limit approach) that it drives the delay to zero when either (i) the message rate grows superlinearly with N, or (ii) the memory grows superlogarithmically with N. Moreover, we show that any policy that has a certain symmetry property, and for which neither condition (i) or (ii) holds, results in an expected queueing delay which is bounded away from zero.

Finally, using the fluid limit approach once more, we show that for any given $\alpha > 0$ (no matter how small), if the policy only uses a linear message rate αN, the resulting asymptotic (as $N \to \infty$) expected queueing delay is positive but upper bounded, uniformly over all $\lambda > 1$. This is a significant improvement over the popular "power-of-d-choices" policy, which has a limiting expected delay that grows as $\log(1/(1-\lambda))$ when $\lambda \uparrow 1$.

Categories and Subject Descriptors

G.3 [**Probability and Statistics**]: Queueing theory, Markov processes; C.2.1 [**Network Architecture and Design**]: Performance tradeoffs

Keywords

Distributed service, Dispatching policies, Performance tradeoffs.

SIGMETRICS '16, June 14 - 18, 2016, Antibes Juan-Les-Pins, France

ISBN 978-1-4503-4266-7/16/06…$15.00

DOI: http://dx.doi.org/10.1145/2896377.2901478

1. INTRODUCTION

This paper addresses the tradeoffs between various measures of performance (delay, message rate and memory) in the context of large scale queueing systems. More specifically, we consider the supermarket model [11], a service system that involves a stream of incoming jobs that are to be dispatched to the queue associated with one of multiple servers. This setting is the dynamic generalization of the classic balls and bins model [2], where N balls have to be sequentially placed into one of N bins.

There is a variety of ways that this system can be operated, which correspond to different system architectures and policies, with different delay performance. At one extreme, incoming jobs can be sent to a random queue. This policy has no informational requirements but incurs a substantial delay because it does not take advantage of resource pooling. At the other extreme, incoming jobs can be sent to a shortest queue, or to a server with the smallest workload. The latter policies have very good performance (small queueing delay), but rely on a substantial information exchange overhead.

Many intermediate policies have been explored in the literature, which achieve different performance levels while using varying degrees of resources such as messages and memory (e.g., the power-of-d-choices [11, 15] or the PULL algorithm [14]). Such policies have been individually analyzed, and the merits of each have been pointed out. However, a comparison is not always possible, because different policies often involve different types of decision making architectures. A more detailed discussion of the relevant literature and the various schemes therein is deferred to Section 4.

The performance-resources tradeoffs have been analyzed in the context of balls into bins models. In particular, the tradeoff between number of messages and maximum load was recently obtained by Lenzen and Wattenhofer [8]. Moreover, the tradeoff between the number of bits of memory available and the maximum load was studied in [1, 4]. To the best of our knowledge, the current study is the first to address these tradeoffs for a dynamic model.

1.1 Our contribution

Instead of focusing on yet another policy or decision making architecture, in this paper we step back and formulate a more fundamental problem. We consider a broad family of decision making architectures and policies, which includes most of those considered in the earlier literature, and work towards characterizing the attainable delay performance, for a given level of resources that are employed.

In more concrete terms, as far as delay is concerned, we

focus on the following qualitative criterion: does the average queueing delay go to zero, as the system size increases? Regarding resources, we allow the dispatcher to have some limited memory, where it can store information on the state of the queues. We also allow the exchange of messages, either from the dispatcher to the servers (queries), and from the servers to the dispatcher; these latter messages can be either responses to queries or spontaneous status updates.

The main question that we study is the following: What is the message rate (total over both directions) and/or memory size requirement that are necessary and sufficient in order to achieve vanishing delay, as the system size grows to infinity? We are able to provide a fairly complete answer to this question. Specifically,

a) If the message rate is proportional to the arrival rate **and** the memory size (in bits) is logarithmic in the number of servers, then *every* decision making architecture and policy, within a certain broad class of policies, results in a delay that does not vanish as the system size increases. The main constraints that we impose on the policies that we consider are: (i) there is no queueing at the dispatcher, and (ii) policies are symmetric, in a sense to be defined.

b) If either the message rate is superlinear in the arrival rate **or** the memory size is superlogarithmic in the number of servers, we provide a fairly simple and natural policy that results in a vanishing delay as the system size increases. Moreover, even if the message rate is proportional to the arrival rate and the memory is logarithmic in the number of servers, this same policy performs better than the alternatives considered earlier in the literature: among other properties, it has uniformly upper bounded queueing delay as $\lambda \uparrow 1$.

We analyze the proposed policy by considering a corresponding fluid limit. This results in expressions that provide a detailed characterization of the resulting queue lengths in the regime where delays do not vanish (case (b) above). Although we do not allow queueing at the dispatcher in our model, the negative result still holds even if we had a central queue with a finite buffer whose size is uniformly upper bounded independent of N. However, if the buffer size can grow with N, there exists a policy with a linear message rate and no memory that achieves vanishing queueing delay. Whether similar negative results can be established without our symmetry assumption is left as an open problem.

1.2 Outline of the paper

The rest of this paper is organized as follows. In Section 2 we introduce some notation. The model and the main results are presented in Section 3, where we also discuss the implications of the results and compare our policy to the so-called "power-of-d-choices" [11, 15]. In Section 4 we put our results in the context of some popular dispatching policies in the literature. In Sections 5, 6, and 7, we provide the proofs of the main results. Finally, in Section 8 we provide our conclusions and suggestions for future work.

2. NOTATION

We define the following sets:

$$\mathcal{S} \triangleq \left\{ s \in [0,1]^{\mathbb{Z}_+} : s_0 = 1;\ s_i \geq s_{i+1},\ \forall i \geq 0 \right\},$$

$$\mathcal{S}^1 \triangleq \left\{ s \in \mathcal{S} : \sum_{i=0}^{\infty} s_i < \infty \right\},$$

$$\mathcal{Q}_N \triangleq \left\{ x \in [0,1]^{\mathbb{Z}_+} : x_i = \frac{K_i}{N},\ \text{for some } K_i \in \mathbb{Z}_+,\ \forall i \right\}.$$

We define the weighted ℓ^2 norm $||\cdot||_w$ on $\mathbb{R}^{\mathbb{Z}_+}$ by

$$||x-y||_w^2 \triangleq \sum_{i=0}^{\infty} \frac{|x_i - y_i|^2}{2^i}.$$

We also define a partial order on $\mathbb{R}_+^{\mathbb{Z}_+}$ as follows:

$$x \geq y \Leftrightarrow x_i \geq y_i \quad \forall\, i \geq 0,$$
$$x > y \Leftrightarrow x \geq y,\ x \neq y,\ \text{and } x_i > y_i,\ \text{if } i \geq 1 \text{ and } y_i > 0.$$

For a given positive integer N, we define the set of server IDs $\mathcal{N} \triangleq \{1, \ldots, N\}$. We let Π_N be the set of probability distributions over $\mathcal{N}$, and let S_N denote the group of permutations over $\mathcal{N}$. We can also define the action of S_N over Π_N by letting, for every $\sigma \in S_N$ and $p \in \Pi_N$,

$$\sigma(p)(\sigma(i)) = p(i), \quad \forall i \in \mathcal{N}.$$

In words, the probability of $\sigma(i)$ under $\sigma(p)$ is the same as the probability of i under p. Finally, we denote by $\mathcal{P}(\mathcal{N})$ the power set of $\mathcal{N}$ and we further extend the action of S_N to $\Pi_{\mathcal{P}(\mathcal{N})}$ (the set of probability distributions over the power set $\mathcal{P}(\mathcal{N})$) so that, for any $\sigma \in S_N$ and $p \in \Pi_{\mathcal{P}(\mathcal{N})}$, and for all $r \in \mathcal{N}$,

$$\sigma(p)(\{\sigma(n_1), \ldots, \sigma(n_r)\}) = p(\{n_1, \ldots, n_r\}),$$

for all $\{n_1, \ldots, n_r\} \in \mathcal{P}(\mathcal{N})$. This means that the probability of selecting the subset of servers $\{\sigma(n_1), \ldots, \sigma(n_r)\}$ under $\sigma(p)$ is the same as the probability of selecting the subset $\{n_1, \ldots, n_r\}$ under p.

3. MODEL AND MAIN RESULTS

In this section we present our main results. In Section 3.1 we describe the model and our assumptions. In Section 3.2 we introduce and study three different variants of a certain pull-based dispatching policy. We study their fluid limits and state the validity of fluid approximations for the transient and the steady-state regimes. Then, in Section 3.3 we present a unified framework for a broad set of dispatching policies that includes most of the policies studied in previous literature, and present our negative result on the queueing delay for resource constrained policies.

3.1 Modeling assumptions

We consider a system consisting of N parallel servers, where each server has a processing rate equal to 1. Furthermore, each server is associated with an infinite capacity, FIFO queue. We use the convention that a job that is being served remains in the queue until its processing is completed. We assume that each server is work conserving: a server is idle if and only if the corresponding queue is empty. Jobs arrive to the system as a single Poisson process of rate λN (for some fixed $\lambda < 1$). Job sizes are i.i.d., independent from the arrival process, and exponentially distributed with mean 1. A central controller (dispatcher) is responsible for routing every incoming job to a suitable queue, immediately upon arrival. The dispatcher may have limited information on the state of the queues; it can only rely on a limited amount

of local memory and on messages that provide partial information about the state of the system. These messages (which are assumed to be instantaneous) can be sent from the servers to the dispatcher at any time, or from the dispatcher to the server (in the form of queries) at the time of an arrival. Messages from the servers can only contain information about the state of their queue (number of remaining jobs and total remaining workload). Within this context, a system designer has the freedom to choose a messaging policy, the rules with which the memory is updated, and the dispatching policy. We are interested in the case where N is very large and where we have some constraints on the number of messages exchanged and on the memory size. The performance metric of interest is the expected delay in steady state of a typical job, i.e., the expected time between its arrival and the time at which it starts receiving service.

3.2 Our dispatching policy and its performance

In this section we introduce our policy, and study the properties of three of its variants.

3.2.1 Policy description

Let us fix N. We consider a policy whereby the dispatcher's memory is used to run a virtual queue that stores up to $C(N)$ server IDs (or tokens), so that the memory size is of order $C(N)\log_2(N)$ bits. Since there are only N distinct servers, we will assume throughout the rest of the paper that $C(N) \leq N$. While a server is idle, it sends messages to the dispatcher as a Poisson process of rate $\mu(N)$ to inform or remind the dispatcher of its idleness. Whenever the dispatcher receives a message, it adds the ID of the server that sent the message to the virtual queue of tokens that is kept in the memory, unless this ID is already stored or the virtual queue is full, in which case it discards the new message. When a new job arrives, if there is at least one server ID in the virtual queue, the job goes to the queue of a server chosen uniformly at random from the virtual queue and the corresponding token is deleted. If there are no tokens present, the job is sent to a queue chosen uniformly at random. This policy, called from now on the Resource Constrained Pull-Based (RCPB) policy or Pull-Based policy for short, is depicted in Figure 1.

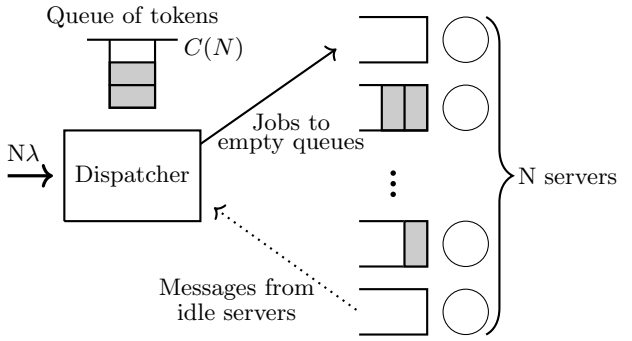

Figure 1: Resource Constrained Pull-Based policy. Jobs are sent to queues associated with idle servers, based on tokens in the virtual queue. If no tokens are present, a queue is chosen at random.

Note that under our policy, no messages are sent from the dispatcher to the servers, which is why it is called a "Pull-Based" policy [14]. We present a summary of our results for the three variants of our policy in Table 1, where we also introduce some mnemonic terms that we will use to refer to these variants. For our purposes, the most relevant subcase of the High Memory variant is when $\mu \geq \frac{\lambda}{1-\lambda}$, which achieves zero asymptotic delay with superlogarithmic memory and linear message rate.

Variant	Memory	Idle message rate	Delay
High Mem.	$C(N) \in \omega(1)$	$\mu(N) = \mu \geq \frac{\lambda}{1-\lambda}$	0
		$\mu(N) = \mu < \frac{\lambda}{1-\lambda}$	> 0
High Mess.	$C(N) = C \geq 1$	$\mu(N) \in \omega(1)$	0
Constr.	$C(N) = C \geq 1$	$\mu(N) = \mu > 0$	> 0

Table 1: The three variants (regimes) of our policy, and the resulting delays, in the limit as $N \to \infty$.

3.2.2 System state representation

For a fixed N, we can model the system as a continuous time Markov chain using as state the queue length vector, denoted by $\left(Q_i^N(t)\right)_{i=1}^N \in \mathbb{Z}_+^N$, together with the number of tokens, denoted by $M^N(t) \in \{0, 1, \ldots, C(N)\}$. Furthermore, as the system is symmetric with respect to the queues, we can replace the queue length vector by the more convenient representation $S^N(t) = \left(S_i^N(t)\right)_{i=0}^\infty$, where

$$S_i^N(t) \triangleq \frac{1}{N}\sum_{j=1}^N \mathbb{1}_{[i,+\infty)}\left(Q_j^N(t)\right), \quad i \in \mathbb{Z}_+,$$

is the fraction of queues with at least i jobs at time t.

3.2.3 Fluid model

We now introduce the fluid model of $S^N(t)$, associated with our policy.

Definition 1. (Fluid model) Given an initial condition $s^0 \in \mathcal{S}$, a function $s(t) : [0,\infty) \to \mathcal{S}$ is said to be a solution to the fluid model (or fluid solution) if

1. $s(0) = s^0$.
2. For all $t \geq 0$, $s_0(t) = 1$.
3. For almost all $t \in [0,\infty)$, and for every $i \geq 1$, $s_i(t)$ is differentiable and satisfies

$$\frac{ds_1(t)}{dt} = \lambda[1 - s_1(t)P_0(t)] - [s_1(t) - s_2(t)], \tag{1}$$

$$\frac{ds_i(t)}{dt} = \lambda[s_{i-1}(t) - s_i(t)]P_0(t) - [s_i(t) - s_{i+1}(t)], \tag{2}$$

for all $i \geq 2$, where $P_0(t)$ is given, for the three variants considered, by:

(i) High Memory: $P_0(t) = \left[1 - \frac{\mu[1-s_1(t)]}{\lambda}\right]^+$;

(ii) High Message: $P_0(t) = \left[1 - \frac{1-s_2(t)}{\lambda}\right]^+ \mathbb{1}_{\{s_1(t)=1\}}$;

(iii) Constrained: $P_0(t) = \left[\sum_{k=0}^C \left(\frac{\mu[1-s_1(t)]}{\lambda}\right)^k\right]^{-1}$.

Note that the fluid model does not include a variable associated with the number of tokens. This is because, as we will see, the virtual queue process $M^N(t)$ evolves on a much faster time scale than the processes of the queue lengths and does not have a deterministic limit. We thus have a process with two different time scales: on the one hand, the virtual queue evolves on a fast time scale (at least N times faster) and from its perspective the queue process $S^N(t)$ appears static; on the other hand, the queue process $S^N(t)$ evolves on a slower time scale; from its perspective, the virtual queue appears to be at equilibrium. This latter property is manifested in the drift of the fluid model: $P_0(t)$ is the probability that the virtual queue is empty when the rest of the system is fixed at the state $s(t)$ (and is given by a different expression for each variant).

We now provide some intuition for each of the drift terms in Equations (1) and (2).

(i) $\lambda[1 - P_0(t)]$: This term corresponds to arrivals to an empty queue while there are tokens in the virtual queue, taking into account that the virtual queue is nonempty with probability $1 - P_0(t)$, in the limit.

(ii) $\lambda[s_{i-1}(t) - s_i(t)]P_0(t)$: This term corresponds to arrivals to a queue with exactly $i-1$ jobs while there are no tokens in the virtual queue. This happens when the virtual queue is empty and when a queue with $i-1$ jobs is drawn uniformly at random, which happens with probability $P_0(t)[s_{i-1}(t) - s_i(t)]$.

(iii) $-[s_i(t)-s_{i+1}(t)]$: This term corresponds to departures from queues with exactly i jobs, which occur at a rate equal to the proportion $s_i(t) - s_{i+1}(t)$ of servers with exactly i jobs.

(iv) Finally, the expressions for $P_0(t)$ are obtained through an explicit calculation of the steady-state distribution of $M^N(t)$ when $S^N(t)$ is fixed at the value $s(t)$, while also letting $N \to \infty$.

3.2.4 Properties of the fluid solutions

The existence of fluid solutions is established by showing that the limit of every convergent subsequence of sample paths $(S^N(t))$ is a fluid solution. Moreover, the following theorem states their uniqueness for all initial conditions $s^0 \in \mathcal{S}$, and it also characterizes the (unique) equilibrium of the fluid model and its asymptotic stability for all initial conditions $s^0 \in \mathcal{S}^1 \subset \mathcal{S}$. The variants described below correspond to assumptions on memory and message rates described in the 2nd and 3rd columns of Table 1, respectively.

Theorem 1. **(Uniqueness and stability of fluid solutions)** *A fluid solution, as described in Definition 1, exists and is unique for any initial condition $s^0 \in \mathcal{S}$. Furthermore, within the set $\mathcal{S}^1$, the fluid model has a unique equilibrium s^*, given by $s_i^* = \lambda[\lambda P_0^*]^{i-1}$, for all $i \geq 1$, where P_0^* is given for the three variants considered by:*

(i) High Memory: $P_0^* = \left[1 - \frac{\mu(1-\lambda)}{\lambda}\right]^+$;

(ii) High Message: $P_0^* = 0$;

(iii) Constrained: $P_0^* = \left[\sum_{k=0}^{C} \left(\frac{\mu(1-\lambda)}{\lambda}\right)^k\right]^{-1}$.

This equilibrium is asymptotically stable, i.e.,

$$\lim_{t\to\infty} \|s(t) - s^*\|_w = 0,$$

for all initial conditions $s^0 \in \mathcal{S}^1$.

The proof for uniqueness and stability is given in Section 5. The proof of the existence is omitted due to space constraints.

3.2.5 Approximation theorems

The three results in this section justify the use of the fluid model as an approximation to the finite stochastic system. The first theorem states that the evolution of the process $S^N(t)$ is almost surely uniformly close, over any finite time horizon $[0, T]$, to the unique fluid solution $s(t)$.

Theorem 2. **(Fluid limit)** *Fix $T > 0$. If*

$$\lim_{N\to\infty} \left\|S^N(0) - s^0\right\|_w = 0, \qquad a.s.,$$

for some $s^0 \in \mathcal{S}$, then

$$\lim_{N\to\infty} \sup_{0\leq t\leq T} \left\|S^N(t) - s(t)\right\|_w = 0, \qquad a.s.,$$

where $s(t)$ is the fluid solution with initial condition s^0.

The proof of this theorem is omitted due to space constraints.

If we combine Theorems 2 and 1 we have that the finite system $S^N(t)$ can be approximated by the equilibrium of the fluid model s^* after some time, because

$$S^N(t) \xrightarrow{N\to\infty} s(t) \xrightarrow{t\to\infty} s^*,$$

almost surely. If we interchange the order of the limits over N and t, we obtain the limiting behavior of the invariant distributions π_s^N of the finite systems as N increases. In the next proposition and theorem, it is shown that the result is the same, i.e., that

$$S^N(t) \xrightarrow{t\to\infty} \pi_s^N \xrightarrow{N\to\infty} s^*,$$

in distribution, so that the interchange of limits is justified.

We start by showing that the stochastic process for any fixed N is positive recurrent.

Proposition 3. **(Positive recurrence for finite N)** *For any fixed N, the stochastic process $\left(S^N(t), M^N(t)\right)$ is positive recurrent and therefore has a unique invariant distribution π^N.*

Proof. In this case it is advantageous to work with the number of jobs in the queues $\left(Q_i^N(t)\right)_{i=1}^N$ instead of with $S^N(t)$. Then the positive recurrence follows by using the simple quadratic Lyapunov function

$$\Phi(Q, m) \triangleq \frac{1}{N}\sum_{i=1}^{N} Q_i^2. \tag{3}$$

□

We then show that the sequence of invariant measures converges in distribution to the Dirac measure concentrated on the equilibrium of the fluid model.

Theorem 4. **(Interchange of limits)** *For any N, let π^N be the unique invariant distribution of the stochastic process $\left(S^N(t), M^N(t)\right)$ and let $\pi_s^N(\cdot) = \sum_{m=0}^{C(N)} \pi^N(\cdot, m)$ be the marginal for S^N. Then,*

$$\lim_{N\to\infty} \pi_s^N = \delta_{s^*}, \quad \textit{in distribution.}$$

The proof is given in Section 6.

Putting everything together, we conclude that the fluid model is a very accurate approximation for the stochastic system when N is very large for the both transient regime (Theorems 2 and 1) and for the steady state regime (Theorem 4).

3.2.6 Delay

Having shown that we can approximate the stochastic system by its fluid model for large N, we can analyze the performance of the equilibrium of the fluid model and thus approximate the expected delay implied by our policy.

For the N-th system, we define the average delay (more precisely, the waiting time) of a job, in steady state, generically denoted by $\mathbb{E}\left[W^N\right]$, as the mean time that the job spends in queue until its service starts. Then by Little's law, we have that

$$\mathbb{E}\left[W^N\right] = \frac{1}{\lambda N}\mathbb{E}\left[\sum_{i=1}^{\infty} N S_i^N\right] - 1,$$

and taking limits as $N \to \infty$ we obtain the limiting queueing delay as

$$\mathbb{E}[W] \triangleq \lim_{N\to\infty} \mathbb{E}\left[W^N\right] = \frac{1}{\lambda}\left(\sum_{i=1}^{\infty} s_i^*\right) - 1,$$

where the interchange of the limit in N, and the expectation and the infinite sum can be justified by applying the Dominated Convergence Theorem using the geometric upper bound from Lemma 10.

High Memory and High Message variants.

For the High Memory variant with $\mu \geq \frac{\lambda}{1-\lambda}$, as well as for the High Message variant, the equilibrium is $s_1^* = \lambda$, and $s_i^* = 0$ for all $i \geq 2$, which yields the claimed zero queueing delay (cf. Table 1).

Constrained variant.

For the Constrained variant, even though we obtain a positive delay, our Pull-Based policy has some remarkable properties. First of all, as discussed above, the delay is

$$\mathbb{E}[W] = \frac{1}{\lambda}\sum_{i=1}^{\infty} s_i^* - 1 = \sum_{i=1}^{\infty} (\lambda P_0^*)^{i-1} - 1 = \frac{\lambda P_0^*}{1-\lambda P_0^*}. \quad (4)$$

Suppose that the message rate of each idle server is $\mu = \mu'\left(\frac{\lambda}{1-\lambda}\right)$ for some constant $\mu' > 0$. Since a server is idle (on average) a fraction $1-\lambda$ of the time, the resulting average message rate at each server is $\lambda\mu'$. We can rewrite the equilibrium probability as

$$P_0^* = \left[\sum_{k=0}^{C} \left(\mu'\right)^k\right]^{-1}.$$

Substituting this in Equation (4) we can obtain an upper bound (uniform in $\lambda < 1$) for the delay in steady state

$$\mathbb{E}[W] \leq \left[\sum_{k=1}^{C} \left(\mu'\right)^k\right]^{-1} \qquad \forall\, \lambda < 1. \quad (5)$$

This implies that if $\mu' > 1$, the queueing delay decreases exponentially fast with the maximum number of tokens C. More importantly, for a fixed $\mu' > 0$ and C, the delay is in fact bounded as $\lambda \uparrow 1$. Finally, the delay is monotonically decreasing with μ' and C.

Phase transition of heavy traffic delay.

We have a phase transition between $\mu' = 0$ (which corresponds to a random uniform routing) and $\mu' > 0$. In the first case, we have the usual unbounded heavy traffic delay of the M/M/1 queue. However, when $\mu' > 0$ the delay is upper bounded uniformly in λ, as shown in Equation (5). This highlights the importance of sending messages while idle and the fact that an intelligent choice of the message budget is to set $\mu = \mu'\frac{\lambda}{1-\lambda}$. Note that this choice still results in a constant ($\lambda\mu'$) average message rate at each server, uniformly upper bounded by μ' for all N and all $\lambda < 1$. This is a key qualitative improvement over all other policies in the literature that have a non-vanishing delay; see the discussion below and in Section 4.

Comparison with the power-of-d-choices.

The expected steady-state delay for the power-of-d-choices policy was shown in [11, 15] to be

$$\mathbb{E}[W_{\text{Pod}}] = \sum_{i=1}^{\infty} \lambda^{\frac{d^i-d}{d-1}} - 1 \geq \lambda^d,$$

which means that the delay decreases at best exponentially with d, much like the delay decreases exponentially with C in our scheme. However, increasing d increases the number of messages sent, unlike our policy.

Furthermore, the delay in heavy traffic for the power-of-d-choices is shown in [11] to grow as $\log\left(\frac{1}{1-\lambda}\right)$ as $\lambda \uparrow 1$ for any fixed d. In contrast, the delay of our scheme has a constant upper bound, independent of λ.

In conclusion, our Pull-Based policy provides better expected delay (especially in heavy traffic) using fewer messages: the power-of-d-choices policy requires $2\lambda dN$ messages per unit of time, while our policy requires $\mu'\lambda N$ messages per unit of time and $C\log_2(N)$ bits of memory.

3.3 Delay lower bound for resource constrained policies

In this subsection we present a negative result that shows that resource constraints on the message rate and on the size of the memory result in asymptotically non-vanishing delays, for a broad class of symmetric policies.

3.3.1 Unified framework for policies

In order to provide a negative result for a class of policies, we need to be very precise on the family of policies considered, including details on the way that the memory is operated, and the precise meaning of "symmetry". We assume that in order to decide the destination of each arriving job, the dispatcher relies entirely on queries to a subset of servers regarding the state of their queues, and on a memory

with C_N bits. The memory is updated only when either (i) the dispatcher receives a message from a server, or (ii) the dispatcher has just dispatched a job.

We define the set of memory states as $\mathcal{M} \triangleq \{1, \ldots, 2^{C_N}\}$. Furthermore, we define the set of possible states at a given queue as the set of nonnegative sequences $\mathcal{Q} \triangleq \mathbb{R}_+^{\mathbb{Z}_+}$, where a sequence specifies the remaining workload of each job in the queue, including the one that is being served. As long as a queue has a finite number of jobs, then the queue state is a sequence with a finite number of non-zero entries. We define next a broad class of memory-based dispatching policies.

Definition 2. (Memory-based dispatching policy) For a fixed N, a memory-based dispatching policy η^N has four components:

1. (Process of messages from servers) We have a bounded rate function $\mu^N : \mathcal{Q} \to \mathbb{R}_+$ which defines a set of modulated Poisson[1] counting processes
$$\left\{Y_n^N(t)\right\}_{n=1}^N.$$
The rate of $Y_n^N(\cdot)$ at each time t depends on the current queue state $Q_n^N(t)$ and equals $\mu^N(Q_n^N(t))$. Besides, the functions are uniformly upper bounded, i.e., $\mu^N(q) \leq \mu^N(0)$ for all $q \in \mathcal{Q}$.

2. (Memory update due to a new message from a server) Given a memory state $m \in \mathcal{M}$, whenever there is an arrival of a message from server $n \in \mathcal{N}$, containing the current queue state $q \in \mathcal{Q}$, the memory is updated according to a function
$$g_{pull}^N : \mathcal{M} \times \mathcal{N} \times \mathcal{Q} \to \mathcal{M}.$$

3. (Server sampling and dispatching rule) Given a memory state $m \in \mathcal{M}$, whenever a new job arrives to the system, the dispatcher sends queries to a subset of the servers, chosen according to the distribution specified by a function
$$\nu^N : \mathcal{M} \to \Pi_{\mathcal{P}(\mathcal{N})}.$$
Then, given the set of pairs $\{(n_1, q_{n_1}), \ldots, (n_r, q_{n_r})\}$ of sampled subset of servers and the state of the corresponding queues, the new arrival is dispatched according to a probability distribution specified by a function
$$f^N : \mathcal{M} \times \mathcal{B} \to \Pi_N,$$
where $\mathcal{B}$ is the subset of $\mathcal{P}(\mathcal{N} \times \mathcal{Q})$ where each element of $\mathcal{N}$ appears only once.

4. (Memory update due to a dispatched job) Immediately after a new job is dispatched to some queue, the memory is updated according to a function
$$g_d^N : \mathcal{M} \times \mathcal{B} \times \mathcal{N} \to \mathcal{M},$$
where the last argument of the function g_d^N is the identity of the server to which the job was dispatched.

[1]Although we will use Poisson processes for the analysis, our negative result can also be established even if the messages are more general renewal processes (e.g., with deterministic inter-arrival times).

Remark 1. Note that we only consider the memory used to store information between one arrival or message to the next. When counting the memory resources used by a policy, we do not take into account information that is used in zero time (e.g., the responses from the queries at the time of an arrival), or the memory required to evaluate the various functions that describe the policy. If that additional memory was accounted for, then the memory constraints would be more severe, and therefore our negative result would still hold.

The functions g_{pull}^N, ν^N, f^N, and g_d^N define a Markov process over the state space $\mathcal{Q}^N \times \mathcal{M}$, as follows. First of all, the states of the queues progress naturally through time as jobs get processed. Then, there are two sources of events: the exogenous arrivals and the messages that servers send to the dispatcher according to the processes $\left\{Y_n^N(\cdot)\right\}_{n=1}^N$. When a message from server n with queue state q arrives to the dispatcher, the memory is updated according to $g_{pull}^N(\cdot, n, q)$. When we have an arrival of a new job, there are three steps that occur sequentially but instantaneously: first, an element of the power set $\mathcal{P}(\mathcal{N})$ (i.e., a subset of $\mathcal{N}$) is selected according to ν^N and the current memory state; second, the job is sent to a queue according to the distribution obtained by f^N evaluated on the actual memory state, sampled subset of servers, and actual states of the queues associated with them; and third, the memory is updated once more according to g_d^N, taking into account the destination of the last job, as well as the IDs and the queue states of the sampled servers.

Remark 2. Here we are considering policies that send messages to the servers only at the time of an arrival and in one round of communication. This eliminates the possibility of having policies that sequentially poll the servers uniformly at random until they find an idle one. The reason behind this restriction is that, in practice, these queries involve some processing and travel time ϵ. Were we to consider a model with $\epsilon > 0$, the sequential polling policies would suffer greatly and would incur positive delay. In contrast our pull-based policy would only have to work with slightly delayed information, but the asymptotic vanishing result would still hold, as long as ϵ is small enough.

Remark 3. We could expand the set of policies by allowing the servers to also send messages whenever there is a service completion. This would result in some additional notation. However, the same negative result can be proved for this somewhat enlarged class of policies as well.

We now introduce a symmetry assumption on the policies.

Definition 3. (Symmetric policy) The memory-based dispatching policy η^N is symmetric if for every permutation of the servers $\sigma \in S_N$ (and associated permutations, also denoted by σ, over the sets $\mathcal{P}(\mathcal{N})$, Π_N, and $\Pi_{\mathcal{P}(\mathcal{N})}$), there exists some permutation σ_M of the memory states such that the following properties hold.

1. (Memory update due to new message from a server)
$$g_{pull}^N(\sigma_M(m), \sigma(n), q) = \sigma_M\left(g_{pull}^N(m, n, q)\right),$$
for all $(m, n, q) \in \mathcal{M} \times \mathcal{N} \times \mathcal{Q}$.

2. (Server sampling and dispatching rule)

$$\nu^N(\sigma_M(m)) = \sigma\left(\nu^N(m)\right),$$

for all $m \in \mathcal{M}$, and

$$f^N(\sigma_M(m),\{(\sigma(n_1),q_1),\ldots,(\sigma(n_r),q_r)\}) = \sigma\left(f^N(m,\{(n_1,q_1),\ldots,(n_r,q_r)\})\right),$$

for all $(m,\{(n_1,q_1),\ldots,(n_r,q_r)\}) \in \mathcal{M}\times\mathcal{B}$.

3. (Memory update due to a dispatched job)

$$g_d^N(\sigma_M(m),\{(\sigma(n_1),q_1),\ldots,(\sigma(n_r),q_r)\},\sigma(n)) = \sigma_M\left(g_d^N(m,\{(n_1,q_1),\ldots,(n_r,q_r)\},n)\right),$$

for all $(m,\{(n_1,q_1),\ldots,(n_r,q_r)\},n) \in \mathcal{M}\times\mathcal{B}\times\mathcal{N}$.

3.3.2 *Lower bound for queueing delay*

We are now ready to state the main result of this section.

Theorem 5. *Fix $C,\alpha > 0$. For any symmetric policy η^N with $C_N = C\log_2(N)$ bits of memory, with a time-average message rate bounded by αN almost surely, and with a unique steady state distribution, we have that the expected queueing delay is uniformly lower bounded, i.e.,*

$$\mathbb{E}\left[W^N\right] \geq b(C,\alpha) > 0,$$

where $\mathbb{E}\left[W^N\right]$ is the expected queueing delay in steady state, and $b(C,\alpha)$ is a constant that only depends on C and α.

A detailed proof is given in Section 7.

4. COMPARISON WITH OTHER POLICIES IN THE LITERATURE

In this section we put our results in perspective by reviewing various dispatching policies in the literature, focusing on their queueing delay and on the amount of resources they use.

4.1 Open-loop policies (without messages)

An optimal open-loop policy dispatches arriving jobs to the queues in a round robin fashion [13]. This policy does not require messages but needs $\log_2(N)$ bits of memory to store the ID of the next queue to receive a job. In the limit, each queue behaves similar to an D/M/1 queue, which still has a $\Omega\left(\frac{1}{1-\lambda}\right)$ delay (see [13]). This policy is not symmetric.

4.2 Policies based on queue lengths

4.2.1 *Join the shortest queue (SQ)*

If we only have access to the queue lengths, an optimal policy is to have each incoming job join a shortest queue. This policy is symmetric, achieves vanishing delay, and requires no memory, but uses $2\lambda N^2$ messages per unit of time (N queries and N responses for each arrival, with rate λN). Alternatively, joining a queue with the least remaining workload performs just as well.

4.2.2 *SQ(d) with memory (SQ(d,b))*

An improvement over the power-of-d-choices, proposed by Mitzenmacher et al. in [12], is obtained by using extra memory to store the b least loaded queues sampled at the time of the previous arrival. When a new job arrives, d queues are sampled uniformly at random and the job is sent to a least loaded queue among the d sampled and the b stored queues. This policy is symmetric, needs $2\lambda dN$ messages per unit of time and $\Omega(b\log_2(N))$ bits of memory, and has positive delay as expected. It compares unfavorably to our policy for the same reasons as for the power-of-d-choices.

4.2.3 *SQ(d) for divisible jobs*

More recently, Ying et al. [16] considered the case of having an arrival rate of $\frac{N\lambda}{m(N)}$ (with $m(N) \in \omega(1)$ and $m(N) \in o(N)$), where each job can be divided into $m(N)$ tasks with mean size 1. The dispatcher samples $dm(N)$ queues and does a water-filling of those queues with the $m(N)$ tasks. In this case, the number of messages sent per unit of time is $2\lambda dN$ and it needs no memory.

Even though this was not mentioned in [16], this policy can drive the queueing delay to 0 if $d \geq \frac{1}{1-\lambda}$. This policy does not fall into our framework because it works with divisible jobs. We could get a very similar policy by queueing $m(N)$ jobs in the dispatcher and dispatching them all at the same time, but this needs an unbounded buffer at the dispatcher.

4.3 Pull-based policies

In order to reduce the message rate, Badonnel and Burgess [3], Lu et al. [10], and then Stolyar [14] proposed that messages be sent from a server to the dispatcher whenever the server becomes idle, so that the dispatcher can keep track of the set of idle servers in real time. Then, an arriving job is to be sent to an empty queue (if there is one) or to a queue chosen uniformly at random if all queues are non-empty. This symmetric policy requires at most λN messages per unit of time and exactly N bits of memory (one bit for each server, indicating whether it is empty or not). Stolyar [14] has shown that when N goes to infinity, the expected delay vanishes.

These pull-based policies are very efficient and they were the inspiration for our own policy. In fact, the High Memory variant of our own policy can be viewed as a more memory-efficient version of Stolyar's PULL policy [14] where instead of requiring N bits of memory, we achieve the desired vanishing delay with any superlogarithmic memory size.

4.4 Memory, messages, and queueing delay

We now summarize the resource requirements (memory and message rate) and the asymptotic delay of the policies reviewed in this section that fall within our framework, together with our three variants.

Policy	Memory (bits)	Message rate	Delay
RRobin [13]	$\log_2(N)$	0	> 0
JSQ	0	$2\lambda N^2$	0
JSQ(d) [11]	0	$2d\lambda N$	> 0
JSQ(d,b) [12]	$\Omega(b\log_2(N))$	$2d\lambda N$	> 0
Pull-based [14]	N	λN	0
High Memory	$\omega(\log_2(N))$	λN	0
High Message	$C\log_2(N)$	$\omega(N)$	0
Constrained	$C\log_2(N)$	$\mu'\lambda N$	> 0

Note that there are several policies that achieve 0 queueing delay in the limit, but (as expected) all of them fall into one (or both) of two categories:

a) Those that need $\omega(N)$ messages. For example: JSQ and JLW require $\Theta(N^2)$ messages per unit of time.

b) Those that need $\omega(\log_2(N))$ bits of memory. For example, the PULL policy [14] requires N bits of memory.

This is consistent with our results: the only way to achieve vanishing delay is by having a superlogarithmic memory or a superlinear message rate. Alternatively, we could obtain a vanishing delay with a linear message rate and an unbounded central buffer [16].

5. PROPERTIES OF THE FLUID MODEL

We divide the proof of Theorem 1 into Lemmas 6 and 7, and Proposition 9 in which we establish uniqueness of solutions, uniqueness of an equilibrium, and asymptotic stability, respectively. The proof of existence is omitted due to space constraints.

5.1 Uniqueness of solutions

Lemma 6. *A fluid solution, as described in Definition 1, with initial condition $s^0 \in \mathcal{S}$, is unique.*

PROOF. The drift in the High Memory and Constrained variants is always Lipschitz-continuous, so the uniqueness of solutions is guaranteed by the Picard-Lindelöff theorem for infinite dimensional spaces [9]. On the other hand, the drift in the High Message variant is only continuous "almost everywhere" so we cannot guarantee the uniqueness of classical (differentiable) solutions. However, we can show the uniqueness of the possibly non-differential solutions of Definition 1. Uniqueness issues only appear when the trajectories hit the closure of the set where the drift is not Lipschitz-continuous, which in this case is the closure of

$$D = \{s \in \mathcal{S} : s_1 = 1 \text{ and } s_2 > 1 - \lambda\}.$$

The proof of the uniqueness for the High Message variant consists in concatenating up to three unique solutions, and has three steps:

1. If $s^0 \in \mathcal{S} \backslash \overline{D}$, then we have uniqueness up to the point in which the solution hits $\overline{D}$.
2. If $s^0 \in D$, we show that the solution stays in D until we have $s_2 = 1 - \lambda$, implying that it can only escape D through $\overline{D} \backslash D$. This is because $\dot{s}_1 > 0$ if $s_2 > 1 - \lambda$. Then, the uniqueness of the solution up to this point is guaranteed by the fact that the drift restricted to $\overline{D}$ is also Lipschitz-continuous.
3. Last, we show that once that it escapes D, it cannot go back in again, thus guaranteeing the uniqueness of the whole solution.

We omit the details due to space constraints. □

5.2 Existence and uniqueness of an equilibrium

Lemma 7. *The fluid model has the unique equilibrium $s^* \in \mathcal{S}^1$ defined in Theorem 1.*

PROOF. Suppose that $s^* \in \mathcal{S}^1$ is an equilibrium. Since $s^* \in \mathcal{S}^1$, we have $\sum_{i=1}^{\infty} s_i^* < \infty$ and we can sum the equilibrium conditions over all coordinates $j \geq i$ for all $i \geq 0$, and get the result. □

5.3 Asymptotic stability of the equilibrium

We will establish asymptotic stability by sandwiching a solution between two solutions that converge to s^*, similar to the argument in [15]. For this purpose, we first establish a monotonicity result.

Lemma 8. *Suppose that s^1 and s^2 are two fluid solutions with $s^1(0) \geq s^2(0)$. Then $s^1(t) \geq s^2(t)$, for all $t \geq 0$.*

The proof of this Lemma consists of checking the drift terms of the fluid model. It is straightforward and is omitted. Now we are ready to prove the convergence result.

Proposition 9. *The equilibrium s^* of the fluid model is asymptotically stable for all initial conditions $s^0 \in \mathcal{S}^1$, i.e.,*

$$\lim_{t \to \infty} \|s(t) - s^*\|_w = 0.$$

PROOF. Fix some $s^0 \in \mathcal{S}^1$. We define initial conditions $s^u(0)$ and $s^l(0)$ by letting

$$s_i^u(0) = \max\{s_i(0), s_i^*\},$$
$$s_i^l(0) = \min\{s_i(0), s_i^*\}.$$

We then have $s^u(0) \geq s^0 \geq s^l(0)$, $s^u(0) \geq s^* \geq s^l(0)$, and $s^u(0), s^l(0) \in \mathcal{S}^1$. Due to Lemma 8 we obtain that $s^u(t) \geq s(t) \geq s^l(t)$ and $s^u(t) \geq s^* \geq s^l(t)$, for all $t \geq 0$. Thus it suffices to prove that $\|s^u(t) - s^*\|_w$ and $\|s^l(t) - s^*\|_w$ converge to 0 as $t \to \infty$.

At this point it is convenient work with the representation $v^u(t)$, $v^l(t)$, and v^*, where $v_i \triangleq \sum_{j=i}^{\infty} s_j$. Then, the dynamics of $v^u(t)$ are of the form

$$\begin{aligned}
\frac{dv_1^u(t)}{dt} &= \lambda - s_i^u(t) = s_1^* - s_i^u(t) \\
\frac{dv_i^u(t)}{dt} &= \lambda s_{i-1}^u(t) P_0^u(t) - s_i^u(t) \qquad \forall\, i \geq 2 \\
&= \lambda [s_{i-1}^u(t) P_0^u(t) - s_{i-1}^* P_0^*] - [s_i^u(t) - s_i^*],
\end{aligned}$$

where in the last step we subtracted the equilibrium condition $0 = \lambda s_{i-1}^* P_0^* - s_i^*$ from the right-hand side.

Now we will prove the coordinate-wise convergence by induction on i. Note that, from the definition of v_i, we have $v_1^u(t) \geq v_i^u(t)$. Furthermore, we have $s_1^u(t) \geq s_1^*$, so that $\frac{dv_1(t)}{dt} \leq 0$. It follows that $v_1^u(0) \geq v_i^u(t) - v_i^u(0) \geq -v_i^u(0)$ uniformly in t. Then, for the base case, we have

$$0 \leq \int_0^{\infty} (s_1^u(\tau) - s_1^*)\, d\tau \leq v_1^u(0) < \infty,$$

which, together with the Lipschitz-continuity of s_1, implies $(s_1^u(\tau) - s_1^*) \to 0$ as $\tau \to \infty$. Now assume that we have $\int_0^{\infty} (s_k^u(\tau) - s_k^*)\, d\tau < \infty$ for all non-negative integers $k \leq i-1$. Then,

$$-v_i^u(0) \leq \int_0^t \left(\lambda \left[s_{i-1}^u(\tau) P_0^u(\tau) - s_{i-1}^* P_0^*\right] - [s_i^u(\tau) - s_i^*]\right) d\tau.$$

Adding and subtracting $\lambda s^*_{i-1} P^u_0(\tau)$ inside the integral, we can check that the former is less than or equal to

$$\int_0^t \left([s^u_{i-1}(\tau) - s^*_{i-1}] + [P^u_0(\tau) - P^*_0] - [s^u_i(\tau) - s^*_i]\right) d\tau, \quad (6)$$

where the first term is uniformly upper-bounded in t by the inductive hypothesis, and it can be checked that the second one is also uniformly upper bounded for all variants using the base case. This completes the induction. It is easily checked that this coordinate-wise convergence implies convergence in $\|\cdot\|_w$, which concludes the proof. □

Note that we proved asymptotic stability only for initial conditions $s^0 \in \mathcal{S}^1 \subset \mathcal{S}$, which correspond to the initial conditions with finite expected queue lengths.

6. INTERCHANGE OF LIMITS

We will prove Theorem 4 in this section, which shows that the sequence of the marginals of the invariant distributions $\left\{\pi^N_s\right\}_{N=1}^{\infty}$ converges in distribution to a measure concentrated on the unique equilibrium (in $\mathcal{S}^1$) of the fluid model. This, together with Proposition 3, guarantees that the properties derived from the equilibrium of the fluid model s^*, specifically for the queueing delay, are an accurate approximation for the steady state of the stochastic system for N large enough. First, we will find a uniform upper bound for $\mathbb{E}_{\pi^N}\left[V_1^N\right]$ in the following lemma.

Lemma 10. *Let π^N be the unique invariant distribution of our process, and let $V_1^N = \frac{1}{N}\sum_{i=1}^N Q_i^N$. We then have the uniform upper bounds*

$$\pi^N\left(Q_1^N \geq k\right) \leq \left(\frac{1}{2-\lambda}\right)^{\frac{k}{2}}, \quad \forall N, \quad \forall k,$$

and

$$\mathbb{E}_{\pi^N}\left[V_1^N\right] \leq 1 + \frac{2}{1-\lambda}, \qquad \forall N.$$

Proof. Consider the linear Lyapunov function

$$\Psi(Q, m) = Q_1.$$

The bounded upward jumps allows us to use Theorem 2.3 from [7] to obtain that $\mathbb{E}_{\pi^N}[Q_1] < \infty$. Thus, all conditions in Theorem 1 in [5] are satisfied, yielding the upper bounds

$$\pi^N\left(Q_1^N \geq 1 + 2m\right) \leq \left(\frac{1}{2-\lambda}\right)^{m+1},$$

and

$$\mathbb{E}_{\pi^N}\left[Q_1^N\right] \leq 1 + \frac{2}{1-\lambda}.$$

The first part of the result is obtained by letting $m = (k-1)/2$ if k is odd or $m = k/2 - 1$ if k is even. Finally, using the definition of V_1^N and then symmetry, we have

$$\mathbb{E}\left[V_1^N\right] = \frac{1}{N}\sum_{i=1}^N \mathbb{E}\left[Q_i\right] = \mathbb{E}\left[Q_1\right],$$

which concludes the proof. □

Now we prove the interchange of limits.

Proof of Theorem 4. Consider the set $\mathbb{Z}_+ \cup \{\infty\}$ endowed with the topology of Alexandroff's compactification, which is known to be metrizable. Moreover, the topology defined by the norm $\|\cdot\|_w$ on $[0,1]^{\mathbb{Z}_+}$ is equivalent to the product topology, which makes it compact. Then, $\mathcal{S}$ (cf. Section 2 for its definition) is also compact because it is a closed subset of $[0,1]^{\mathbb{Z}_+}$. As a result, we have that the product $\mathcal{S} \times (\mathbb{Z}_+ \cup \{\infty\})$ is a compact Polish space under the product topology. Then, $\left\{\pi^N\right\}_{N=1}^{\infty}$ is trivially tight and by Prohorov's theorem [6], it is also relatively compact in the weak topology, and any sequence has a weakly convergent subsequence.

Given a weakly convergent subsequence $\left\{\pi^{N_k}\right\}_{k=1}^{\infty}$ and its limit π, we can use Skorokhod's representation theorem to construct a probability space $(\Omega_0, \mathcal{A}_0, \mathbb{P}_0)$ and a sequence of random variables $\left(S^{N_k}(0), M^{N_k}(0)\right)$ distributed according to π^{N_k}, such that

$$\lim_{k\to\infty} \left\|S^{N_k}(0) - s(0)\right\|_w = 0 \qquad \mathbb{P}_0 - a.s. \quad (7)$$

for some random variable $s(0) \sim \pi_s$, where π_s is the marginal of π. Furthermore, we use $\left(S^{N_k}(0), M^{N_k}(0)\right)$ as the initial conditions for a sequence of processes $\left\{\left(S^{N_k}(t), M^{N_k}(t)\right)\right\}_{k=1}^{\infty}$, which makes these processes stationary. Note that the initial conditions, distributed as π^{N_k}, do not necessarily converge to a deterministic initial condition (this is actually part of what we are trying to prove), so we cannot use Theorem 2 directly to find the limit of the sequence of processes $\left\{S^{N_k}(t)\right\}_{k=1}^{\infty}$. However, given any $\omega \in \Omega_0$ outside a zero P_0-measure set, we can restrict this sequence of stochastic processes to the probability space $(\Omega_\omega, \mathcal{A}_\omega, \mathbb{P}_\omega) =$

$$(\Omega_D \times \Omega_S \times \{\omega\}, \mathcal{A}_D \times \mathcal{A}_S \times \{\emptyset, \{\omega\}\}, \mathbb{P}_D \times \mathbb{P}_S \times \delta_\omega)$$

and apply Theorem 2 to this new space, to obtain

$$\lim_{k\to\infty} \sup_{0\leq t\leq T} \left\|S^{N_k}(t,\omega) - s(t,\omega)\right\|_w = 0, \quad \mathbb{P}_\omega - a.s.,$$

where $s(t,\omega)$ is the fluid solution with initial condition $s(0,\omega)$. Since this is true for all $\omega \in \Omega_0$ except for a set of zero P_0-measure, it follows that

$$\lim_{k\to\infty} \sup_{0\leq t\leq T} \left\|S^{N_k}(t) - s(t)\right\|_w = 0, \qquad \mathbb{P} - a.s.,$$

where $\mathbb{P} = \mathbb{P}_D \times \mathbb{P}_S \times \mathbb{P}_0$ and where $s(t)$ is another stochastic process whose randomness is only in the initial condition (its trajectory is the deterministic fluid solution for that specific initial condition).

The uniform bound on $\mathbb{E}_{\pi^N}[V_1^N]$ (Lemma 10) and the convergence of π^N to π imply that $\mathbb{E}_\pi[v_1] \leq 1 + 2/(1-\lambda)$ and therefore $v_1 < \infty$ a.s., or equivalently, $\pi_s\left(\mathcal{S}^1\right) = 1$. This means that every possible initial condition is in $\mathcal{S}^1$, almost surely (and not in $\mathcal{S}\backslash\mathcal{S}^1$). Now suppose that $\left\{S^{N_k}(0)\right\}_{k=1}^{\infty}$ does not converge $\mathbb{P}_0$-a.s. to s^* (i.e., that $\pi_s \neq \delta_{s^*}$). Then, the set of processes with initial conditions in $\mathcal{S}^1\backslash\{s^*\}$ has probability $\pi_s\left(\mathcal{S}^1\backslash\{s^*\}\right) > 0$ and for every such process $s(t,\omega)$ with initial condition $s(0,\omega)$ we have

$$\|s(0,\omega) - s^*\|_w > \|s(t,\omega) - s^*\|_w$$

for t large enough (this is due to the fact that these sample paths converge asymptotically to s^*, by Theorem 1). As a result,

$$\mathbb{E}_{\pi_s}\left[\|s(0) - s^*\|_w\right] > \mathbb{E}_{\pi_s}\left[\|s(t) - s^*\|_w\right]$$

for t large enough, which is a contradiction, because S^{N_k} is stationary and thus s also has to be stationary. Then, the limit of every convergent subsequence of the marginals $\{\pi_s^N\}_{N=1}^{\infty}$ has to be δ_{s^*}. $\square$

7. PROOF OF THE NEGATIVE RESULT

We now prove our negative result for resource constrained policies. The proof proceeds through a sequence of lemmas. Due to space constraints, we will only prove it for the case in which the distribution for the sampling of servers, $\nu^N(m)$, is independent of m. The general case can be proven with a similar line of reasoning.

We first show that for any fixed time t, we can have any given finite number of arrivals before the next message or departure, with probability bounded away from zero, uniformly for all N.

Lemma 11. *Let η^N be a memory-based dispatching policy with message rate of at most αN for some $\alpha > 0$. If T_t^N is the (random) time of the first departure or message from a server after some fixed time $t > 0$, then for every $k \in \mathbb{Z}_+$ we have that $\mathbb{P}\left(k \text{ arrivals in } \left(t, T_t^N\right)\right) \geq P(\alpha, \lambda, k) > 0$, where $P(\alpha, \lambda, k)$ is a constant independent from N.*

PROOF. Because of the upper bound for the message rate of any server (cf. part 1 of Definition 2), we have that $\mu^N\left(Q_n^N(t)\right) \leq \mu^N(0)$ for all $t \geq 0$. In any interval $[0, t]$ we must have at most as many departures as arrivals to the system. Therefore, the time-average number of departures per unit of time is at most λN and thus we will have at least $(1-\lambda)N$ idle servers on the average. As a result, $\mu^N(0)$ has to be uniformly upper bounded by $\frac{\alpha}{1-\lambda}$ in order to have at most αN messages per unit of time, in average.

Starting at any time $t > 0$, we have k arrivals after an Erlang(k,λN) time. Moreover, we will have a departure no sooner than after an exponential time of rate N (which corresponds to the worst case in which all servers are busy), and we will have a message no sooner than after an exponential time of rate $N\frac{\alpha}{1-\lambda}$ (which corresponds to the worst case in which all servers are idle). Then, we get the uniform bound

$$\begin{aligned}
&\mathbb{P}\left(k \text{ arrivals in } \left(t, T_t^N\right)\right) \\
&\qquad \geq \mathbb{P}\left(Erl(k, N\lambda) < exp\left(\frac{N\alpha}{1-\lambda} + N\right)\right) \\
&\qquad \geq \mathbb{P}\left(Erl(k, \lambda) < exp\left(\frac{\alpha}{1-\lambda} + 1\right)\right),
\end{aligned}$$

which is a positive number, independent of N. $\square$

We now state a simple lemma regarding the number of probability distributions over $\mathcal{N}$. It implies that unless many servers have the same probability, a distribution will have many distinct permutations.

Lemma 12. *If $p \in \Pi_N$ has at most u servers with the same probability of being selected, then*

$$\#\{\sigma(p) : \sigma \in S_N\} \geq \max\left\{\binom{N}{u}, \lceil N/u \rceil!\right\}.$$

The proof consists of a simple counting argument and is thus omitted.

The next step is to characterize the probability distributions on the set of servers induced by the dispatching rule function f^N when we take C_N to be logarithmic in N.

Lemma 13. *Consider a symmetric dispatching policy, with $C_N = C\log_2(N)$, and dispatching rule function f^N, and let r be a positive integer. Then for every $m \in \mathcal{M}$, for every subset $\{n_1, \ldots, n_r\} \in \mathcal{P}(\mathcal{N})$, and for every $(q_{n_1}, \ldots, q_{n_r}) \in \mathcal{Q}^r$, there exist at least $N - C - r$ servers with the same probability of being selected under $f^N(m, \{(n_1, q_1), \ldots, (n_r, q_r)\})$.*

PROOF. Fix $(q_1, \ldots, q_r) \in \mathcal{Q}^r$. First, note that there are $\binom{N}{r}$ different subsets $\{n_1, \ldots, n_r\}$, and N^C different memory states m. Then, with $(q_1, \ldots, q_r) \in \mathcal{Q}^r$ fixed, the argument of f^N can only take $\binom{N}{r}N^C$ different values, that lead to at most $\binom{N}{r}N^C$ different distributions. Let $p \in \Pi_N$ be one of those distributions, i.e., $p = f^N(m, \{(n_1, q_1), \ldots, (n_r, q_r)\})$ for some $m \in \mathcal{M}$ and $\{n_1, \ldots, n_r\} \in \mathcal{P}(\mathcal{N})$. Then, $\sigma(p) = f^N(\sigma_M(m), \{(\sigma(n_1), q_1), \ldots, (\sigma(n_r), q_r)\})$ is also a possible distribution due to the symmetry hypothesis. As a result,

$$\#\{\sigma(p) : \sigma \in S_N\} \leq \binom{N}{r}N^C, \tag{8}$$

which is the upper bound for the total number of different distributions implied by the limited number of different arguments for f^N.

On the other hand, if p has at most u servers with the same probability of being selected, Lemma 12 provides the lower bound

$$\#\{\sigma(p) : \sigma \in S_N\} \geq \max\left\{\binom{N}{u}, \lceil N/u \rceil!\right\},$$

which combined with Equation (8) yields

$$\max\left\{\binom{N}{u}, \lceil N/u \rceil!\right\} \leq \binom{N}{r}N^C. \tag{9}$$

Now, if $C + r < u < N - C - r$, then $\binom{N}{u} > \binom{N}{C+r+1} > \binom{N}{r}N^C$ for N large enough, which contradicts Equation (9). If $u \leq C + r$, then $\lceil N/u \rceil! \geq \lceil N/(C+r) \rceil! > \binom{N}{r}N^C$ for N large enough, which also contradicts Equation (9). Thus $u \geq N - C - r$. $\square$

We are now ready to prove our main result.

PROOF OF THEOREM 5. Consider the system in steady state, fix some time $T > 0$ and let $\mathcal{C}^N(T)$ be the event that there are at least $C + 1$ arrivals before the next message or departure. This event has probability bounded away from zero due to Lemma 11, and it is an event that depends only on the Poisson processes of arrivals, departures, and messages. We will show that, for N large enough, a policy will send at least one of those $C + 1$ jobs to a non empty queue, with probability bounded away from zero.

Without loss of generality, we may assume that the expected delay is finite. This implies that there is no accumulation of jobs in the system, and that the expected number of busy servers is λN (because our system is in steady state). Then, by the Markov inequality, we have that the probability of $Q^N(T)$ having at least $N(\lambda - \epsilon)$ busy servers is greater than $\epsilon/(1 - \lambda + \epsilon)$, for all $\epsilon \in (0, \lambda)$. Furthermore, since the message rate is at most αN, and since the number of sampled queues is independent from the arrivals (only depends

on ν^N), it follows that the expected number of queues sampled at the time of an arrival is less than or equal to α/λ. Therefore, using the Markov inequality once more, we get that the probability of sampling less than β servers is lower bounded by $1-\alpha/(\lambda\beta)$, which is positive for $\beta > \alpha/\lambda$. Then, by the independence of the sampling, the probability that the number of sampled servers for each of the first $C+1$ arrivals after time T are fewer than β, is lower bounded by $(1-\alpha/(\lambda\beta))^{C+1}$. Moreover, due to the symmetry hypothesis, the sampling distribution ν^N is uniform over all subsets of the same size. This implies that we have a further subset of sample paths $\mathcal{C}'(T) \subset \mathcal{C}(T)$, with probability bounded away from zero uniformly in N, such that for the first $C+1$ arrivals after time T, all sampled servers are busy. From now on, we will focus on those sample paths.

Recall that a policy defines a continuous time Markov process, $\left(Q^N(t), M^N(t)\right)$, over the state space $\mathcal{Q}^N \times \mathcal{M}$. The evolution of this process between time T and the time of the $(C+1)$-st arrival, for all sample paths in $\mathcal{C}'(T)$, can be described in terms of some basic random variables, as follows.

Let t_k be the time of the k-th arrival after time T, let $\{U(k)\}_{k=1}^{C+1}$ be a collection of independent random subsets distributed as ν^N (the sampled servers at the time of each arrival), let $\{V(k)\}_{k=1}^{C+1}$ be a collection of i.i.d. random variables, uniform in $[0,1]$ (the seeds for the randomized selection of the destinations), and let $\{W(k)\}_{k=1}^{C+1}$ be a collection of i.i.d. exponential random variables with parameter 1 (the workloads of each incoming job). All the randomness in our system is modeled by these random variables, and the process $\left(Q^N(t), M^N(t)\right)$ between times T and t_{C+1} will be a deterministic function of them.

To begin with, the remaining workload of the jobs that are being processed, decreases at a rate of one unit of work per unit of time. Moreover, for all sample paths in $\mathcal{C}'(T)$, the memory process $M^N(t)$ is only updated at the time of an arrival. Consider the mapping $F_{f^N} : [0,1] \times \mathcal{M} \times \mathcal{B} \to \mathcal{N}$, such that $F_{f^N}(V(k), m, \{(n_1,q_1),\dots,(n_r,q_r)\})$ is distributed according to $f^N(m, \{(n_1,q_1),\dots,(n_r,q_r)\})$, and such that

$$\sigma\left(F_{f^N}(V(k), m, \{(n_1,q_1),\dots,(n_r,q_r)\})\right) = F_{f^N}(V(k), \sigma_M(m), \{(\sigma(n_1),q_1),\dots,(\sigma(n_r),q_r)\}), \quad (10)$$

for all $\sigma \in S_N$. Then, let

$$X(k) = F_{f^N}\left(V(k), M^N\left(t_k^-\right), \left\{\left(U(k), Q^N\left(t_k^-\right)\right)\right\}\right) \quad (11)$$

be the random destination of the new job. At the time of the k-th arrival, the following steps occur sequentially but in zero time.

1. A random subset of servers is chosen ($U(k)$).

2. Based on $U(k)$ and the state of their queues, a destination for the new job is chosen ($X(k)$) using the seed $V(k)$.

3. The memory is updated according to

$$M^N(t_k) = g_d^N\left(M^N(t_k^-), \left\{\left(U(k), Q^N(t_k^-)\right)\right\}, X(k)\right), \quad (12)$$

and a new job with workload $W(k)$ is added to queue $X(k)$ in Q^N.

Recall that for all sample paths in $\mathcal{C}'(T)$, the memory process is constant between arrivals, and thus its evolution after time T is completely determined by the previous update equation, for which we have $M^N(t_1^-) = M^N(T)$ and $M^N(t_k^-) = M^N(t_{k-1})$ for all $k \geq 2$.

For every sample path ω, let $E_k(\omega)$ be a least cardinality set such that $f^N\left(M^N(t_{k-1},\omega), \left\{\left(U(k,\omega), Q^N(t_k^-,\omega)\right)\right\}\right)$ is uniform (or zero) in $E_k(\omega)^c$, and let $E^N(C+1)(\omega)$ be the union of those $E_k(\omega)$ for all $1 \leq k \leq C+1$. The sets $E_k(\omega)$ will have cardinality of at most $C+\beta$ by Lemma 13, for all $\omega \in \mathcal{C}'(T)$.

Using the construction of $\left(Q^N(t), M^N(t)\right)$ described above, we can define a mapping G, that takes as arguments

- the initial condition $\left(M^N(T,\omega), Q^N(T,\omega)\right)$,
- the arrival times $\{t_k(\omega)\}_{k=1}^{C+1}$, and the corresponding workloads $\{W(k,\omega)\}_{k=1}^{C+1}$,
- the subsets of sampled servers $\{U(k,\omega)\}_{k=1}^{C+1}$ and seeds $\{V(k,\omega)\}_{k=1}^{C+1}$,

and yields the set $E^N(C+1)(\omega)$.

Using the symmetry hypothesis in Equations (11) and (12), the symmetry of the map F_{f^N} from Equation (10), and proceeding recursively, it can be checked that the previous mapping G is "symmetric". In particular, for every permutation $\sigma \in S_N$ such that $\sigma(U(k,\omega)) = U(k,\omega)$, if $1 \leq k \leq C+1$, we have that $\sigma\left(E^N(C+1)(\omega)\right)$ is obtained when the argument of G is

- the initial condition $\left(\sigma_M\left(M^N(T,\omega)\right), Q^N(T,\omega)\right)$,
- the arrival times $\{t_k(\omega)\}_{k=1}^{C+1}$, and the corresponding workloads $\{W(k,\omega)\}_{k=1}^{C+1}$,
- the subsets of sampled servers $\{U(k,\omega)\}_{k=1}^{C+1}$ and seeds $\{V(k,\omega)\}_{k=1}^{C+1}$.

We omit the details of this claim due to space constraints. Note that, in order to obtain $\sigma\left(E^N(C+1)\right)$, we only need to change the initial memory state to $\sigma_M\left(M^N(T)\right)$. Then, due to the fact that we have only N^C memory states, we can only obtain at most N^C different sets $\sigma(E^N(C+1))$ by varying σ among those that have $\sigma(U(k,\omega)) = U(k,\omega)$, if $1 \leq k \leq C+1$, i.e., we have that

$$\#\left\{\sigma\left(E^N(C+1)\right) : \sigma \in A\right\} \leq N^C, \quad (13)$$

where

$$A = \{\sigma \in S_N : \sigma(U(k)) = U(k), \text{ if } 1 \leq k \leq C+1\}. \quad (14)$$

Recall that for all $\omega \in \mathcal{C}'(T)$, the number of sampled servers ($\#U(k,\omega)$) is at most β, so these permutations leave at most β servers fixed for each one of the $C+1$ arrivals, which translates into at most $(C+1)\beta$ servers fixed in total.

Now consider the collection of sample paths $\mathcal{C}'_1(T) \subset \mathcal{C}'(T)$ where at least one of the sets $E_k(\omega)$ is empty. For those sample paths, at least one of the jobs is dispatched according to a uniform distribution, and because we had at least $N(\lambda-\epsilon)$ non empty queues with probability bounded away from zero, at least one job is sent to a non empty queue with probability bounded away from zero (this last probability is conditioned on $\mathcal{C}'_1(T)$). If this event has overall probability bounded away from zero (without conditioning

on $\mathcal{C}_1'(T)$), we are done. On the other hand, for all the sample paths in $\mathcal{C}'(T)\backslash\mathcal{C}_1'(T)$, none of the $E_k(\omega)$ are empty. In order to derive a contradiction, assume that the first $C+1$ jobs are sent to an empty queue for a collection of sample paths with "high probability" (in this context, "high probability" refers to probabilities converging to 1 as $N \to \infty$). This implies that the jobs are sent to servers in $E^N(C+1)$ with high probability. Otherwise, their destination would be distributed uniformly over E_k^c (which has cardinality of at least $N-\beta-C$ by Lemma 13), and they would go to a non empty queue with probability bounded way from zero, for the same reason as before. Then, each E_k must contain at least 1 idle server with high probability (and at most $C+\beta$ by Lemma 13), and thus $E^N(C+1)$ must contain between $C+1$ and $(C+\beta)(C+1)$ idle servers. This means that, if $a = \#E^N(C+1)$, then

$$\#\left\{\sigma\left(E^N(C+1)\right) : \sigma \in S_N\right\} = \binom{N}{a} \geq \binom{N}{C+1}.$$

If instead of using all permutations $\sigma \in S_N$, we take the subset A defined in Equation (14), then, if $b = \#\bigcup_{k=1}^{C+1} U(k)$ and $a' = \#\left(E^N(C+1)\backslash\bigcup_{k=1}^{C+1} U(k)\right)$, we have that

$$\#\{\sigma(E^N(C+1)) : \sigma \in A\} = \binom{N-b}{a'} \geq \binom{N-(C+1)\beta}{C+1},$$

because all sampled servers $U(k)$ (which are at most $(C+1)\beta$) are busy, and thus disjoint with the idle servers in $E^N(C+1)$ (which are at least $C+1$). However, we also had an upper bound for this quantity in Equation (13), which leads to a contradiction because $N^C < \binom{N-(C+1)\beta}{C+1}$ for N large enough. This concludes the proof. □

8. CONCLUSIONS AND FUTURE WORK

The main objective of this paper is to find necessary and sufficient conditions on the amount of resources (messages and memory) available to a central dispatcher, in order to achieve a vanishing queueing delay as the system size increases. This is done by defining a unified framework for a broad class of dispatching policies and by proving two separate results: First, we show that when we have a limited amount of memory and a modest budget of messages per unit of time, all dispatching policies result in queueing delay uniformly bounded away from zero. Second, we present two variants of a simple symmetric pull-based dispatching policy for which it is sufficient to have a little more memory or messages in order to have vanishing queueing delay. As an added bonus, we analyze a third variant of the same dispatching policy with positive queueing delay. We show that by wisely exploiting an arbitrarily small message rate (but still proportional to the arrival rate) we obtain a queueing delay which is finite and uniformly upper bounded even in heavy traffic, a significant qualitative improvement over the M/M/1 queueing delay obtained in the absence of messages.

There are several interesting directions for future research. In light of the symmetry assumption in Theorem 5, an immediate open question is whether the result still holds without this assumption. Our proof heavily relies on the symmetry assumption and is difficult to generalize. Second, we could relax the assumption of homogeneous servers, consider pools of servers with different service rates, and study the tradeoffs between resources and the stability region of a policy. In this setting we expect a result similar to our lower bound for queueing delay, stating that a symmetric resource constrained policy cannot be stable for every stabilizable system. Last but not least, it would be interesting to extend some or all of these results to the case of general job size distributions (e.g., heavy tailed) and/or different service disciplines such as processor sharing or LIFO, as these are prevalent in many applications.

9. REFERENCES

[1] N. Alon, E. Lubetzky, and O. Gurel-Gurevich. Choice-memory tradeoff in allocations. In *Proceedings of FOCS*, 2009.

[2] Y. Azar, A. Broder, A. Karlin, and E. Upfal. Balanced allocations. *SIAM Journal on Computing*, 29(1):180–200, 1999.

[3] R. Badonnel and M. Burgess. Dynamic pull-based load balancing for autonomic servers. In *Network Operations and Management Symposium*, 2008.

[4] I. Benjamini and Y. Makarychev. Balanced allocations: memory performance tradeoffs. *The Annals of Applied Probability*, 22(4):1642–1649, 2012.

[5] D. Bertsimas, D. Gamarnik, and J. Tsitsiklis. Performance of multiclass markovian queueing networks via piecewise linear lyapunov functions. *The Annals of Applied Probability*, 11(4):1384–1428, 2002.

[6] P. Billingsley. *Convergence of Probability Measures.* Wiley, second edition, 1999.

[7] B. Hajek. Hitting-time and occupation-time bounds implied by drift analysis with applications. *Advances in Applied Probability*, 14(3):502–525, 1982.

[8] C. Lenzen and R. Wattenhofer. Tight bounds for parallel randomized load balalcing. *Distributed Computing*, pages 1–16, 2014.

[9] S. Lobanov and O. Smolyanov. Ordinary differential equations in locally convex spaces. *Uspekhi Mat. Nauk*, 49:93–168, 1994.

[10] Y. Lu, Q. Xie, G. Kliot, A. Geller, J. Larus, and A. Greenberg. Join-Idle-Queue: A novel load balancing algorithm for dynamically scalable web services. *Performance Evaluation*, 68(11):1056–1071, Nov. 2011.

[11] M. Mitzenmacher. *The power of two choices in randomized load balancing.* PhD thesis, U.C. Berkeley, 1996.

[12] M. Mitzenmacher., B. Prabhakar., and D. Shah. Load balancing with memory. In *Proceedings of FOCS*, 2002.

[13] G. Stamoulis and J. Tsitsiklis. Optimal distributed policies for choosing among multiple servers. In *Proceedings of the CDC*, pages 815–820, 1991.

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

[15] N. Vvedenskaya, R. Dobrushin, and F. Karpelevich. Queueing system with selection of the shortest of two queues: an asymptotic approach. *Problems of Information Transmission*, 32(1):15–27, 1996.

[16] L. Ying, R. Srikant, and X. Kang. The power of slightly more than one sample in randomized load balancing. In *Proceedings of INFOCOM*, 2015.