Faculteit Wetenschappen
Departement Wiskunde - Informatica

# Load Balancing in Large Distributed Networks

Proefschrift voorgelegd tot het behalen van de graad van
Doctor in de Wetenschappen: Informatica
aan de Universiteit Antwerpen
te verdedigen door

**Wouter Minnebo**

**Promotor:**
Prof. Dr. B. Van Houdt

**ANTWERPEN, 2016**

**Promotor:**
Prof. Dr. B. Van Houdt

**Commissieleden:**
Prof. Dr. J. Broeckhove, Prof. Dr. C. Blondia

*Load Balancing in Large Distributed Networks*
Nederlandse titel: Load Balancing in Grote Gedistribueerde Netwerken

*Nothing ever becomes real till it is experienced.*

John Keats

# Acknowledgements

Completing this PhD has been a personal challenge and would not have happened without the support of several people important to me. I am fortunate to have them in my life, and I would like to take a moment to express my appreciation.

First of all, I would like to thank my advisor, Benny, for giving me the opportunity to do a PhD. His inquisitive mindset encouraged me to investigate several hunches with varying success. I enjoyed our meetings and learned more than just the technical advice. Problems were solved with my favorite approach: humor and flexibility.

I thank my parents for understanding me and offering support when I needed it most. I also thank my brother for being himself and deciding to share a home together, it is a pleasure.

I thank my non-academic teachers as well, for their patience and guidance over the years. It is a joy to practice and refine my skills with you. Thank you Jimmy, Marc, Carsten and Karin.

I also thank my friends, for reminding me there exist also fun non-work-related activities. Apparently there is a whole world out there worth exploring.

My office mates also directly and indirectly affected this work, balancing lively discussions and thoughtful silence appropriately. Thank you Robbe, Robin and Ignace! Other colleagues further contributed to a pleasant atmosphere at work. Thank you, Andrew, Bart, Bart, Daniel, Erwin, Jeremy, Jeroen, Johan, Julia, Kathleen, Kim, Kurt, Kurt, Michael, Nico, Nik, Nikolaas, Pieter, Przemyslaw, Ruben, Sam, Sean, Stijn, Tim.

Also a word of thanks to the members of my doctoral jury: Chris Blondia, Luca Bortolussi, Jan Broeckhove, Toon Calders, Giuliano Casale. Their comments significantly improved this thesis.

**Wouter Minnebo,**
**Antwerpen, 2016**

# Contents

# Publications

- **W. Minnebo** and B. Van Houdt, Pull versus push mechanism in large distributed networks: Closed form results, in Proc. of the 24-th International Teletraffic Congress, Krakau (Poland), 2012. (received the Best paper award ITC 2012 and the Internet Technical Committee Best Paper 2012)
- **W. Minnebo** and B. Van Houdt, A fair comparison of pull and push strategies in large distributed networks, IEEE/ACM Transactions on Networking, 2013.
- **W. Minnebo** and B. Van Houdt, Improved rate-based pull and push strategies in large distributed networks, in Proc. of the IEEE 21-th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, San Francisco, 2013.
- **W. Minnebo** and B. Van Houdt, Analysis of rate-based pull and push strategies with limited migration rates in large distributed networks, in Proc. of the 9th EAI International Conference on Performance Evaluation Methodologies and Tools, Berlin, 2015.
- **W. Minnebo** and B. Van Houdt, On a Class of Push and Pull Strategies with Single Migrations and Limited Probe Rate, Performance Evaluation, 2016 [In Submission]

Chapter 1

# Introduction

There was a time when teams of programmers shared a single computer. This was when computers were an exotic piece of equipment the size of a room and reserved for institutions. Having a personal computer at home was a distant dream. Times have changed and now computers are everywhere, cheaper and faster than ever. This has significantly advanced science and transformed our daily lives. An important step in this evolution has been the ability for multiple computers to collaborate in order to achieve a common goal. In other words, a program is no longer confined to a single computer, but coordinates the execution of tasks on multiple computers interconnected through a network. Such a collection of cooperating computers is called a *distributed system*. We consider a commonly used model, representing the distributed system as a collection of nodes or queues containing tasks or jobs, each equipped with a single server that processes those jobs in sequence.

Developing, deploying and managing distributed systems involves solving several interesting questions. The question this thesis explores can be roughly stated as: Which server should be responsible for processing what task? However, we are not necessarily interested in initial job assignments, but rather the system's ability to (re)distribute the workload later. This is a key feature of a contemporary distributed system consisting of a large number of processing nodes, and instrumental in reducing the queuing delay of tasks. Put differently, the question reads: How should nodes distribute their tasks over the available resources in order to increase responsiveness? This is known as the problem of *load balancing / load sharing*.

Maximizing responsiveness of tasks in distributed networks is increasingly relevant due to the explosive growth of *cloud computing*. One main idea of cloud computing is that the infrastructure can be scaled at runtime, adding or removing servers depending on the real-time resource requirements of the running program(s). This is a tangible benefit for a variety of applications, for example web-servers that can scale up if they experience many concurrent users. Our setup assumes that all nodes have a way to communicate with all other nodes. In practice this might be impractical as the system becomes large and the size is dynamic on a short timescale, as it requires updating this info on all nodes. A plausible way to sidestep

this issue is to organize the nodes hierarchically, such that each node is only aware of a certain neighborhood. Our results should be applicable if the neighborhoods are of reasonable size, approximately a hundred nodes or more.

Another area where cloud computing plays a significant role is the offloading of computationally intensive tasks from smart devices to more powerful servers. This is also known as *mobile cloud computing* [1,2]. Apart from the question when to add or remove servers, the load balancing problem is apparent: how should we balance the incoming requests over the active servers? Proposed solutions must scale well, as cloud applications typically use a large number of servers. A good load balancing implementation reduces the delay tasks experience. This is important because even a small increase in delay can result in the loss of users and revenue [3].

A key distinguishing feature of how a distributed system is organized is the initial task assignment or origin of the jobs. This can be organized by one or multiple central dispatchers (e.g., [4–6]), or jobs can enter the system via the processing nodes themselves (e.g., [7–10]).

## 1.1 Centralized Load Balancing

Traditionally, distributed applications use a single central component called the load balancer to distribute incoming tasks among available servers. In this case join-the-shortest-queue is a straightforward strategy [11]. However, this requires that the load balancer is aware of all the queue lengths in the system. As the system grows in size this becomes impractical, especially if multiple load balancers use the same server pool. Furthermore, this queue length information must be updated regularly, as decisions based on old information can negatively affect the system's performance [12].

A practical solution when using multiple load balancers is join-the-idle queue [6], where idle servers inform a well chosen load balancer of their idle state. When a new task arrives, the load balancer forwards it to an idle server if one is known at that time. This is closely related with the asymptotically optimal PULL proposed in [13], which uses a single load balancer and requires one bit of state information per server.

In [14] the fundamental memory and communication requirements of a strategy are determined, in order to drive the expected steady state queueing delay of a job to zero as the system size increases. It is shown that this objective can be met if the message rate grows superlinearly with the system size, or the memory of the load balancer grows superlogarithmically with the system size.

Another approach using a centralized load balancer which we will revisit later, is the strategy which we will call *d-choices*. It lets the load balancer sample $d$ queue lengths and forwards the task to the least loaded node. This policy does not require knowledge of all queue lengths at all times, and improves the queuing delays dramatically compared to randomized load balancing. This strategy is also known as the power-of-d-choices and is widely studied [4, 5, 15, 16]. A small adaptation is useful if tasks arrive in batch, then it is advantageous to sample multiple servers and distribute the batch over the discovered servers instead of treating each task separately [17].

## 1.2 Distributed Load Balancing

In case no centralized dispatchers are used, the workload is redistributed by the exchange of jobs between the processing nodes. Two important families of strategies have been identified for redistributing jobs: the *pull* and *push* strategies. Under the pull strategy lightly-loaded nodes try to attract work from highly-loaded nodes, a strategy that is also known as *work stealing*. Under the push strategy the highly-loaded nodes take the initiative to transfer jobs to lightly-loaded nodes, a strategy that is also known as *work sharing*.

We further distinguish between *traditional* strategies which send a batch of probes at task arrival (push) or completion times (pull), and *rate-based* strategies which send probes periodically. Even though they operate differently we show that under comparable conditions, both approaches perform equally well. We note that for some systems it is not feasible to migrate tasks after the initial server assignment. Therefore, the strategies we consider are more suited for computational workloads where the cost of migration is small, as opposed to web services where TCP connections have to be migrated along with the task [6].

To facilitate the exchange of jobs, some central information may be stored. However, as the network size grows continuously updating this information becomes more challenging. Therefore, fully distributed networks do not rely on centralized information. Instead a node that wishes to pull/push a job will transmit a probe to one (or multiple) other nodes that are typically selected at random. If a node that receives such a probe is willing to take part in the exchange, it will send a positive reply and the job exchange can proceed. Clearly, the overall rate at which probe messages are sent based on a strategy plays a pivotal role in its effectiveness.

## 1.3 Motivation and Main Results

Several authors studied the performance of push and pull strategies. A comparison for a homogeneous distributed system with Poisson arrivals and exponential job lengths was presented in [7, 18] and extended to heterogeneous systems in [19, 20]. Load stealing is also commonly used in context of shared-memory multiprocessor scheduling [21]. These studies showed that the pull strategy is superior under high load conditions, while the push strategy achieves a lower mean delay under low to moderate loads. More recent analytic studies of the performance of pull and push strategies include [9, 22].

A common feature in these papers is that the number of nodes $N$ is not a model parameter. Instead in [7, 18, 21] some global balance equations are solved, and it is confirmed numerically that the solution provides a good approximation for finite systems of moderate size. In contrast, we start from a finite system modeled as a Markov chain with size $N$ and let $N$ tend to infinity, yielding a so-called infinite system model which we show to be the correct limiting process, similar to [9]. However, we model our nodes to have infinite buffers instead of a buffer with finite capacity, resulting in a state space of infinite dimensionality where compactness is not guaranteed. This requires us to prove several conditions that would automatically hold in the finite dimensional case, such as the existence of a suitable environment around trajectories and tightness of the sequence of steady

state measures.

The infinite system model consists of a set of ordinary differential equations (ODEs) describing the time evolution of the system. Such infinite system models are also constructed directly in [22]. We then show that the ODEs we consider have a unique fixed point that is a global attractor. Our proof of global attraction relies on La Salle's invariance principle for Banach spaces, a novel approach we did not encounter in our review of the literature related to mean field modeling. Other authors sometimes prove global attraction by finding a Lyapunov function for the system, e.g. the weighted $L_1$ norm in [23]. Alternatively, global attraction can be proven if the system has a monotonicity property such as in [24].

Having a unique fixed point that is a global attractor allows us to express the steady state queue length distribution. The equations to find this fixed point correspond with the global balance equations used by other authors. Furthermore, we confirm by simulation that when using the infinite system model to approximate a finite system with size $N$, the error decreases as $N$ increases.

All of these studies provided valuable insights with respect to the performance of pull and push strategies. However, they also paid hardly any attention to the probe rate, that is, the number of probe messages that the strategies under consideration generate per time unit.

Therefore, the strength of the push and pull mechanism was only compared to some extent, mainly because the overall probe rate $R$ of these strategies may be very different. This is especially true for the hybrid pull/push strategy introduced in [8], where it has been shown to outperform both the push and pull strategy for all loads $\lambda$. However, as indicated in [8], such a hybrid strategy results in a (far) higher probe rate $R$.

As different strategies tend to have different probe rates that depend to a large extent on the arrival rate $\lambda$, it is typically not possible to adapt the parameters of the strategies under consideration such that they generate the same overall probe rate (for arbitrary $\lambda$), making any comparison biased.

In Chapter 2 we develop a rate-based push and pull strategy where probes are sent periodically with rate $r$. We can choose this $r$ to match a predefined overall probe rate $R$, thereby allowing a fair comparison between push and pull strategies. We show that the infinite system model we introduce is the correct limiting process as the system size $N$ tends to infinity, and that the ODEs describing the behavior of the system have a unique fixed point that is a global attractor. Moreover, we provide closed form results for the steady state queue length distribution and mean delay. Simulation results show that the infinite system model is an accurate approximation for a finite system of moderate size, and that the approximation error decreases as the system size increases. In a performance comparison we show that the push strategy outperforms the pull strategy in terms of the mean response time as well as in the decay rate of the queue length distribution, for any probe rate $R > 0$ when $\lambda < \phi - 1$, where $\phi = (1 + \sqrt{5})/2 \approx 1.6180$ is the golden ratio. More generally, we show that the push strategy prevails for any

$$\lambda < \frac{\sqrt{(R+1)^2 + 4(R+1)} - (R+1)}{2}.$$

We also show that a pure pull or push strategy is always superior to a hybrid approach. In addition, we show that these new rate-based strategies are equivalent

to the traditional strategies where probes are sent at arrival or completion instants, if the same overall probe rate is used.

In Chapter 3 we extend the range where push outperforms pull, by realizing that a push strategy can increase its performance by only transferring tasks from relatively long queues. If the expected mean queue length is large, it intuitively makes sense for a push strategy to only send probes if the queue length exceeds some threshold. Therefore, we consider a push strategy that only allows queues with length longer than $T$ to send probes. As a natural extension we introduce the max-push strategy, where only queues with length $T$ send probes and all new arrivals at these queues are instantly migrated to an empty server. We explore whether it would be beneficial for a pull strategy to only accept jobs from a queue with length longer than $T$, and show that for the pull strategy setting $T = 1$ is optimal. Furthermore, we generalize the proofs from Chapter 2 to also be applicable for the strategies introduced in this chapter, and numerically validate the infinite system models.

In Chapter 4 we analyze the effect of a limited migration rate $M$ in addition to a limited probe rate $R$. We provide clear policies on how to choose the parameter $r$ in order to satisfy both the $R$ and $M$ constraints. In this situation, we show that setting $T = 1$ is no longer optimal for the pull strategy. In order to increase the performance of the pull strategy in this problem setting, we introduce an extension called the conditional-pull strategy. We note that for this new strategy the theoretical results from Chapter 3 also hold, and validate the infinite system model numerically. Additionally, we show an elegant way to express the mean delay of all considered strategies in terms of the migration rate $M$ as a side result.

In Chapter 5 we further extend the model to a general class of strategies. One specific push strategy within this class allows that not only empty servers can receive incoming migrations, and lets queues with length at most $B$ do so as well. Similarly, we consider a pull strategy that allows all queues with length at most $B$ to send probes. We show that rate-based strategies in this setting are equivalent to their traditional counterparts, if the same overall probe rate is used. We conjecture that the max-push strategy we introduce in Chapter 3 and extend in this chapter is an optimal push strategy within the general class we consider, when the overall probe rate $R$ is limited. For pull strategies we conjecture that letting only empty servers send probes and accept incoming tasks from all busy servers is optimal within the general class we consider, when only the overall probe rate $R$ is limited. In case the migration rate $M$ is also limited, only accepting tasks from longer queues can increase performance, as shown in Chapter 4. Furthermore, we develop a distributed version of d-choices which needs an overall probe rate of

$$R = \frac{\lambda^2(1-\lambda^{d-1})}{1-\lambda}$$

to perform equally well as a naive implementation with $d$ probes per task. We also show that rate-based push variants can be constructed that are equivalent in performance when given the same probe budget, if we allow a queue's probe rate to be dependent on its length. Moreover, we compared the performance of the best performing rate-based push and pull strategy with d-choices, given that the same overall probe rate is used. The pull strategy is the best choice for high loads, but its simplicity and reasonable performance for low to moderate loads makes it a viable

solution in case the system must use a single strategy. For low loads the max-push and d-choices performance is nearly equivalent, with the max-push achieving a slightly lower mean delay for medium to high loads. Still, it is remarkable that the simple d-choices strategy performs so close to the more complicated max-push which we conjecture to be an optimal push strategy.

In Chapter 6 we summarize the main contributions of the thesis and propose several possibilities for future work.

Chapter 2

# A Fair Comparison of Pull and Push Strategies in Large Distributed Networks

*In this chapter* [1] *we compare the performance of the pull and push strategy in a large homogeneous distributed system. When a pull strategy is in use, lightly loaded nodes attempt to steal jobs from more highly loaded nodes, while under the push strategy more highly loaded nodes look for lightly loaded nodes to process some of their jobs.*

*Given the maximum allowed overall probe rate $R$ and arrival rate $\lambda$, we provide closed form solutions for the mean response time of a job for the push and pull strategy under the infinite system model. More specifically, we show that the push strategy outperforms the pull strategy for any probe rate $R > 0$ when $\lambda < \phi - 1$, where $\phi = (1 + \sqrt{5})/2 \approx 1.6180$ is the golden ratio. More generally, we show that the push strategy prevails if and only if $2\lambda < \sqrt{(R+1)^2 + 4(R+1)} - (R+1)$. We also show that under the infinite system model, a hybrid pull and push strategy is always inferior to the pure pull or push strategy.*

*The relation between the finite and infinite system model is discussed and simulation results that validate the infinite system model are provided.*

[1] This chapter is based on work published in IEEE Transactions on Networking as "A Fair Comparison of Pull and Push Strategies in Large Distributed Networks" by Wouter Minnebo and Benny Van Houdt [25]. That work is an extended version of the paper "Pull versus Push Mechanism in Large Distributed Networks: Closed Form Results", published in ITC24 by the same authors [26].

## 2.1 Introduction

Our aim in this chapter is to compare the pull and push mechanism given that both generate the same overall probe rate $R$. Further, we also provide closed form expressions for the main performance measures under the infinite system model.

To this end we introduce a rate-based pull and push strategy, under which nodes do not transmit probes at job completion or job arrival times. Instead idle nodes will generate probes at some rate $r$ under the pull strategy, while under the push strategy nodes send probes at some rate $r$ whenever they have jobs waiting. A desirable property of both these strategies is that they can match any overall probe rate $R$ under any load $\lambda$ by setting $r$ in the appropriate manner. More specifically, let $R$ be the average number of probes send by a node per time unit, irrespective of its queue length. Clearly, the overall probe rate $R$ will be less than the rate $r$. By establishing a simple relationship between $r$ and $R$, we will determine $r$ to match any predefined overall probe rate $R$. In fact, we show that if rate based pull/push strategy matches the overall probe rate of the traditional pull/push strategy, it also matches the queue length distribution and therefore the mean response time (under the infinite system model).

Given some $R > 0$, we will show that under the infinite system model the push strategy outperforms the pull strategy for any

$$\lambda < \frac{\sqrt{(R+1)^2 + 4(R+1)} - (R+1)}{2},$$

in terms of the mean response time (as well as in the decay rate of the queue length distribution). As $R$ approaches zero, the right-hand side decreases to $\phi - 1$, where $\phi = (1+\sqrt{5})/2$ is the golden ratio, which indicates that the push strategy prevails for any $R$ when $\lambda < (-1+\sqrt{5})/2 \approx 0.6108$.

As a side result we show that the queue length distribution of the pull and push strategy is in fact identical when they use the same rate $r$ (instead of the same $R$). We also consider a hybrid strategy where idle nodes probe at rate $r_1$ and nodes with pending jobs probe at rate $r_2$, where $r = r_1 + r_2$ is again determined by matching $R$, leaving one degree of freedom. We will show that for any for $\lambda$ and $R$ the optimal policy exists in setting either $r_1$ or $r_2$ to zero. This implies that the hybrid strategy is in fact never better when the overall probe rate $R$ is not allowed to increase.

The infinite system model corresponds to a distributed system with an infinite number of nodes $N$. Using simulation results, we will show that the infinite system is quite accurate for both strategies and moderate to large size systems, e.g., for $N \geq 100$ the relative error is typically below 1 percent. For smaller systems, e.g., $N = 25$, the infinite system model results in higher relative errors, especially for the pull strategy under high loads. The loads for which the push strategy outperforms the pull strategy are however still quite accurately predicted by the infinite system model, even for small systems.

The chapter is structured as follows. In Section 2.2 we introduce the push, pull and hybrid strategy considered in this chapter and discuss its relation with many existing strategies studied before. The infinite system model is presented in Section 2.3 and closed form results are derived for the queue length distribution and mean delay. Using these results we identify the loads at which the push strategy outperforms the pull strategy and prove that a hybrid strategy is always inferior.

Simulation results that validate the infinite system model are presented in Section 2.4. In Section 2.5 we discuss and prove the technical issues related to showing that the infinite system model is indeed the proper limit process of the sequence of finite system models, while in Section 2.6 we show that rate-based strategies matching the probe rate of the traditional strategies also match the queue length distribution. Conclusions are drawn in Section 2.7.

## 2.2 Pull and Push Strategies

As in [7, 9, 19, 22] the model considered in this chapter relies on the following assumptions:

1. The system consists of $N$ nodes, where each node consist of a single server and an infinite buffer to store jobs.

2. Each node is subject to its own local Poisson arrival process with rate $\lambda$. Jobs require an exponential processing time with mean 1 and are served in a first-come-first-served (FCFS) order.

3. The time required to transfer probe messages and jobs between different nodes can be neglected in comparison with the processing time (i.e., the transfer times are assumed to be zero).

The above assumptions also imply that all jobs can be successfully executed by any node in the system and that there are no communication failures on the network that interconnects the nodes.

We consider the following three basic strategies:

1. *Push:* Whenever a node has $i \geq 2$ jobs in its queue, meaning $i - 1$ jobs are waiting to be served, the node will generate probe messages at rate $r$. Thus, as long as the number of jobs in the queue remains above 1, probes are sent according to a Poisson process with rate $r$. Whenever the queue length $i$ drops to 1, this process is interrupted and will remain interrupted as long as the queue length remains below 2. The node that is probed is selected at random and is only allowed to accept a job if it is idle.

2. *Pull:* Whenever a node has $i = 0$ jobs in its queue, meaning the server is idle, the node will generate probe messages at rate $r$. Thus, as long as the server remains idle, probes are sent according to a Poisson process with rate $r$. This process is interrupted whenever the server becomes busy. The probed node is also selected at random and the probe is successful if there are jobs waiting to be served.

3. *Hybrid:* This strategy combines the above two strategies. When queue length $i$ equals 0 a node generates probes at rate $r_1$, while for $i \geq 2$ the probe rate is set equal to $r_2$.

We will show that under the infinite system model (i.e., $N = \infty$) the push and pull strategy result in exactly the same queue length distribution when the same rate $r$ is used. This is even true for the hybrid strategy if we define $r = r_1 + r_2$

(i.e., the queue length only depends on the sum of $r_1$ and $r_2$). However, when the same rate $r$ is used by these different strategies, the overall probe rate $R$ will typically differ. Hence, we aim at comparing these strategies when the rates $r$ are set such that the overall probe rate matches some predefined $R$.

The pull strategy considered in this chapter is in fact identical to the pull strategy with repeated attempts considered in [22, Section 2.4], except that our nodes do not immediately generate a probe message when the server becomes idle. Generating probes in that way would automatically result in a high probe rate $R$ when the load $\lambda$ is small and would no longer allow us to match any $R > 0$ by setting $r$ in the appropriate manner.

The traditional pull and push strategies considered in [7,8,19] and discussed in Section 2.6 (for $T = 1$) are somewhat different. The pull strategy tries to attract a job whenever a job completes and the resulting queue length is below $T$, while the push strategy tries to push arriving jobs that find $T$ or more jobs in the queue upon arrival. Further, instead of sending a single probe, both strategies repeatedly send probes until either one gets a positive reply or a predefined maximum of $L_p$ probes is reached. The overall probe rate $R$ clearly depends on $T, L_p$ and the load $\lambda$, which makes it hard to compare the pull and push strategies in a completely fair manner.

When the time required to transfer probes and jobs between nodes is neglected (as in [7]), setting $T = 1$ ensures that exchanged jobs can immediately start (as in our setup). Assuming zero transfer time for probe messages is quite realistic as transferring jobs typically requires considerably more time than sending a probe. The models in [8,19] do take an exponentially distributed job transfer time into account (while still assuming zero transfer time for the probes). The results show that when increasing the transfer time, the delays also increase, while the performance differences between the pull and push strategies become less significant (but remain similar). Further, the setting $T = 1$ minimizes the mean response time when the job transfer times are sufficiently small (but also results in a higher overall probe rate $R$).

The strategies considered in [9,10] are more aggressive pull and push strategies. In [9] a successful probe message results in exchanging half of the jobs that are waiting, while in [10] the number of probes send under the push strategy depends on the current queue length. Although the push strategy of [10] significantly reduces the mean response time and outperforms the pull strategy, its overall probe rate is also much higher.

## 2.3 Infinite System Model

In this section we present various analytical results in closed form for the system with $N = \infty$ nodes, termed the infinite system model. The evolution of the infinite system model will be defined by a set of ordinary differential equations (ODEs) and is thus deterministic. The evolution of the finite system models (for which $N < \infty$) on the other hand will be captured by an $N$-dimensional continuous time Markov chain (CTMC) and is therefore stochastic. To define the infinite system model we first consider a system with a finite number of nodes $N$. Due to the assumptions on the arrival process, processing times and transfer times, it suffices

to keep track of the $N$ queue lengths in order to obtain a CTMC. Further, as the system is homogeneous, it also suffices to keep track of the number of nodes that have $i$ jobs in their queue for all $i \geq 1$. More precisely, we define a CTMC $\{X^{(N)}(t) = (X_1^N(t), X_2^N(t), \ldots)\}_{t\geq 0}$, where $X_i^{(N)}(t) \in \{0, \ldots, N\}$ is the number of nodes with *at least* $i$ jobs in the queue at time $t$ (the superscript $N$ is used to indicate that we consider a finite system consisting of $N$ nodes). For any state $x = (x_1, x_2, \ldots)$ we clearly have that $x_i \geq x_{i+1}$ for all $i \geq 1$.

Let us first indicate that the transition rates of this CTMC are identical for the push, pull and hybrid strategy provided that they use the same rate $r$. Transitions take place when either one of the following three events takes place: an arrival, a job completion, or a job exchange between an idle node and a node with at least two jobs. Let $q^{(N)}(x, y)$ be the transition rate between state $x = (x_1, x_2, \ldots)$ and state $y = (y_1, y_2, \ldots)$. If an arrival occurs in a queue with $i - 1$ jobs, then $x_i$ will increase by one. Thus, due to the arrivals we have

$$q^{(N)}(x, y) = \lambda(x_{i-1} - x_i),$$

for $y = x + e_i$ and $i \geq 1$, where $x_0 = 1$ (as all the queues contain zero or more jobs) and $e_i$ is a vector with a 1 in position $i$ and 0s elsewhere. Similarly, a job completion in a queue with $i$ jobs reduces $x_i$ by one:

$$q^{(N)}(x, y) = (x_i - x_{i+1}),$$

for $y = x - e_i$ and $i \geq 1$. A job exchange between an idle node and a node with $i$ jobs increases $x_1$ by one and decreases $x_i$ by one; hence, $y = x + e_1 - e_i$. Under the push strategy the rate of such exchanges equals the number of nodes with exactly $i$ jobs $x_i - x_{i+1}$ times $r$ times the probability that a probe message is successful[2], which equals $(N - x_1)/N$. Hence, for the push strategy we have

$$q^{(N)}(x, y) = r(1 - x_1/N)(x_i - x_{i+1}),$$

for $y = x + e_1 - e_i$ and $i \geq 2$. Under the pull strategy this event takes place with a rate equal to the number of idle nodes $(N - x_1)$ times $r$ times the probability $(x_i - x_{i+1})/N$ that we select a node with $i$ jobs. The transition rate is therefore the same in both systems. For the hybrid strategy these events occur at rate $r_1(1 - x_1/N)(x_i - x_{i+1})$ (due to pull) plus $r_2(1 - x_1/N)(x_i - x_{i+1})$ (due to the push), which results in the same overall rate. Using a coupling argument one can prove that this CTMC is positive recurrent for all $\lambda < 1$ (see proof of Theorem 2.10). Let $\pi^{(N)} = (\pi_1^{(N)}, \pi_2^{(N)}, \ldots)$ be the unique stationary measure of the Markov chain $\{X^{(N)}(t)\}_{t\geq 0}$, then the overall probe rate $R^{(N)}$ for the pull, push and hybrid strategy equal $r(1 - \pi_1^{(N)})$, $r\pi_2^{(N)}$ and $r_1(1 - \pi_1^{(N)}) + r_2\pi_2^{(N)}$, respectively.

We will now define the infinite system model, the evolution of which is described by a set of ODEs, using the rates $q^{(N)}(x, y)$. As these rates are the same for the three strategies, they also result in the same set of ODEs. Let $L = \{e_i, i \geq 1\} \cup \{-e_i, i \geq 1\} \cup \{e_1 - e_i, i \geq 2\}$ be the set of possible transitions (arrivals, job completions and

[2]We assume that the probed node is selected at random, in fact we even allow a node to select itself with probability $1/N$. Disallowing nodes to select themselves results in the same limiting process.

job transfers). Next, we define

$$F(x) = \sum_{\ell \in L} \ell \beta_\ell(x),$$

where $\beta_\ell(x)$ is defined such that $\beta_\ell(x/N) = q^{(N)}(x, x+\ell)/N$. Given the above expressions for $q^{(N)}(x, x+\ell)/N$, this implies

$$\begin{aligned} \beta_{e_i}(x) &= \lambda(x_{i-1} - x_i), \\ \beta_{-e_i}(x) &= (x_i - x_{i+1}), \end{aligned}$$

for all $i \geq 1$ (with $x_0 = 1$ for $i = 1$) and

$$\beta_{e_1 - e_i}(x) = r(1 - x_1)(x_i - x_{i+1}),$$

for $i \geq 2$. In other words,

$$F(x) = \sum_{i \geq 1} (e_i \beta_{e_i}(x) - e_i \beta_{-e_i}(x)) + \sum_{i \geq 2} (e_1 - e_i)\beta_{e_1 - e_i}(x),$$

where $x = (x_1, x_2, \ldots)$, with $x_i \in [0,1]$ and $x_i \geq x_{i+1}$ for $i \geq 1$. The set of ODEs describing the evolution of the infinite system model is now given by $\frac{d}{dt}x(t) = F(x(t))$, where $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$ in the infinite system. This set of ODEs can be written as

$$\frac{d}{dt}x_1(t) = (\lambda + r x_2(t))(1 - x_1(t)) - (x_1(t) - x_2(t)), \tag{2.1}$$

and

$$\frac{d}{dt}x_i(t) = \lambda(x_{i-1}(t) - x_i(t)) - (1 + r(1 - x_1(t)))(x_i(t) - x_{i+1}(t)), \tag{2.2}$$

for $i \geq 2$. In Section 2.5 we will discuss the relation between this dynamical system and the finite system models for large $N$.

Let $E = \{(x_1, x_2, \ldots) | x_i \in [0,1], x_i \geq x_{i+1}, i \geq 1, \sum_{j \geq 1} x_j < \infty\}$. The next two theorems show that this set of ODEs is Lipschitz on $E$ and it has a unique fixed point in $E$.

**Theorem 2.1.** *The function $F$ is Lipschitz on $E$.*

*Proof.* $F$ is Lipschitz provided that for all $x, y \in E$ there exists an $Z > 0$ such that $|F(x) - F(y)| \leq Z\,|x - y|$. By definition of $F(x)$ one finds

$$|F(x) - F(y)| \leq 2(\lambda + 1 + 2r)\,|x - y| + 2r \sum_{i \geq 2} |x_1(x_i - x_{i+1}) - y_1(y_i - y_{i+1})|\,.$$

The above sum can be bounded by

$$\sum_{i \geq 2} |(x_1 - y_1)(x_i - x_{i+1}) + y_1(x_i - x_{i+1} - y_i + y_{i+1})|\,,$$

which is bounded by $2\,|x - y|$ on $E$. Hence, $F$ is Lipschitz by letting $Z = 2\lambda + 2 + 8r$. □

As $E$ is a Banach space the Lipschitz condition of $F$ suffices to guarantee that the set of ODEs $\frac{d}{dt}x(t) = F(x(t))$, with $x(0) \in E$, has a unique solution[3] $\phi_t(x(0))$ [27, Section 1.1].

**Theorem 2.2.** *The set of ODEs given by* (2.1) *and* (2.2) *has a unique fixed point* $\pi = (\pi_1, \pi_2, \ldots)$ *with* $\sum_{i\geq 1} \pi_i < \infty$. *Further,*

$$\pi_i = \lambda \left( \frac{\lambda}{1 + (1-\lambda)r} \right)^{i-1}.$$

*Proof.* Assume $\pi$ is a fixed point with $\sum_{i\geq 1} \pi_i < \infty$, meaning $F_i(\pi) = 0$ for $i \geq 1$, where $F(x) = (F_1(x), F_2(x), \ldots)$. When $\sum_{i\geq 1} \pi_i < \infty$, we can write $\sum_{i\geq 1} F_i(\pi)$ using (2.2) as

$$\begin{aligned}\sum_{i\geq 1} F_i(\pi) &= F_1(\pi) + \sum_{i\geq 2} \lambda(\pi_{i-1} - \pi_i) - \sum_{i\geq 2} (1 + r(1-\pi_1))(\pi_i - \pi_{i+1}) \\ &= F_1(\pi) + \lambda\pi_1 - (1 - r(1-\pi_1))\pi_2 \\ &= \lambda - \pi_1,\end{aligned}$$

where the last equality follows from (2.1). In other words, $\sum_{i\geq 1} F_i(\pi) = 0$ implies that $\pi_1 = \lambda$. Further, by defining $\eta_i = \pi_i - \pi_{i+1}$, the condition $F_i(\pi) = 0$, for $i \geq 2$, readily implies that $\eta_{i+1} = \lambda\eta_i/(1 + (1-\pi_1)r)$ and therefore by induction we find the expression for $\pi_i$, for $i \geq 2$. □

If we take the set of ODEs in (2.1) and (2.2) and replace the first $x_2(t)$ by $\pi_2$ in (2.1) and $x_1(t)$ by $\lambda$ in (2.2), then we end up with the Kolmogorov equations for a state dependent M/M/1 queue with $\lambda_0 = \lambda + r\pi_2$, $\lambda_i = \lambda$, for $i \geq 1$, $\mu_1 = 1$ and $\mu_i = 1 + (1-\lambda)r$, for $i \geq 2$. The arrival process of such an M/M/1 queue is Poisson with rate $\lambda_i$ and the service is exponential with rate $\mu_i$ whenever the queue length equals $i$. Hence, the fixed point $\pi$ also corresponds to the steady state of a state dependent M/M/1 queue (where $\pi_i$ is the probability that the queue contains at least $i$ jobs).

The set of ODEs in (2.1) and (2.2) describes the transient behavior of the infinite system, while we are in fact interested in its behavior as $t$ goes to infinity. Thus, we are interested in the limit of all the trajectories of this set of ODEs.

We start by proving the following Lemma:

**Lemma 2.1.** *Let* $x(t)$ *be the unique solution of the ODEs given by* (2.1) *and* (2.2) *with* $x(0) \in E$. *The* $L_1$*-distance to the unique fixed point* $\sum_{i\geq 1} |x_i(t) - \pi_i|$ *does not increase as a function of* $t$.

*Proof.* Define $\epsilon_i(t) = x_i(t) - \pi_i$, for $i \geq 1$, such that $\Phi(t) = \sum_{i\geq 1} |\epsilon_i(t)|$ represents the $L_1$-distance. As $\frac{d}{dt}x_i(t) = \frac{d}{dt}\epsilon_i(t)$ and $\pi$ is a fixed point of (2.1) and (2.2), we find

$$\frac{d}{dt}\epsilon_1(t) = -\lambda\epsilon_1(t) + (1 + r(1-\pi_1))\epsilon_2(t) - r\epsilon_1(t)(\epsilon_2(t) + \pi_2) - \epsilon_1(t), \tag{2.3}$$

[3]The solution $\phi_t(x)$ belongs to the class of continuously differentiable functions as in the finite dimensional case.

and

$$\begin{aligned}\frac{d}{dt}\epsilon_i(t) =&\lambda(\epsilon_{i-1}(t)-\epsilon_i(t))-(1+r(1-\pi_1))(\epsilon_i(t)-\epsilon_{i+1}(t))\\ &+r\epsilon_1(t)(\epsilon_i(t)-\epsilon_{i+1}(t)+\pi_i-\pi_{i+1}),\end{aligned} \tag{2.4}$$

for $i \geq 2$. Assume for now that $\epsilon_i(t) \neq 0$ for all $i$ such that $\frac{d}{dt}\Phi(t)$ is properly defined as

$$\frac{d}{dt}\Phi(t) = \sum_{i:\epsilon_i(t)>0}\frac{d}{dt}\epsilon_i(t) - \sum_{i:\epsilon_i(t)<0}\frac{d}{dt}\epsilon_i(t).$$

If $\epsilon_i(t)$ has the same sign for all $i$, one finds that $\frac{d}{dt}\Phi(t) = -|\epsilon_1(t)|$ by summing (2.3) and (2.4). We will show that

$$\frac{d}{dt}\Phi(t) \leq -|\epsilon_1(t)|,$$

holds in general. Let $I = \{i_1, i_2, \ldots\}$, with $i_1 < i_2 < \ldots$, be the set of indices where $\epsilon_i(t)$ changes sign, that is, $\epsilon_{i-1}(t)$ and $\epsilon_i(t)$ have a different sign if and only if $i \in I$. Assume $\epsilon_1(t) < 0$ and let $I_+ = \{i_1, i_3, \ldots\}$ and $I_- = \{i_2, i_4, \ldots\}$ such that $i \in I_+$ implies that $\epsilon_{i-1}(t) < 0$ and $\epsilon_i(t) > 0$, while $i \in I_-$ implies that $\epsilon_{i-1}(t) > 0$ and $\epsilon_i(t) < 0$.

When $\epsilon_{i-1}(t)$ and $\epsilon_i(t)$ are both positive (for $i \geq 2$), the terms $\lambda\epsilon_{i-1}(t)$, $-(1 + r(1-\pi_1))\epsilon_i(t)$ and $r\epsilon_1(t)(\epsilon_i(t)+\pi_i)$ in $\frac{d}{dt}\epsilon_i(t)$ are canceled by $\frac{d}{dt}\epsilon_{i-1}(t)$ when computing $\frac{d}{dt}\Phi(t)$. However, if $\epsilon_{i-1}(t) < 0$ and $\epsilon_i(t) > 0$, $\frac{d}{dt}\epsilon_{i-1}(t)$ is replaced by $-\frac{d}{dt}\epsilon_{i-1}(t)$ in $\frac{d}{dt}\Phi(t)$ and therefore contains these three terms twice. Hence, in general

$$\begin{aligned}\frac{d}{dt}\Phi(t) = \epsilon_1(t) &+ 2\sum_{i\in I_+}\underbrace{(\lambda\epsilon_{i-1}(t)-(1+r(1-\pi_1))\epsilon_i(t))}_{<0}\\ &-2\sum_{i\in I_-}\underbrace{(\lambda\epsilon_{i-1}(t)-(1+r(1-\pi_1))\epsilon_i(t))}_{>0}\\ &+2\sum_{i\in I_+} r\epsilon_1(t)(\epsilon_i(t)+\pi_i) - 2\sum_{i\in I_-} r\epsilon_1(t)(\epsilon_i(t)+\pi_i).\end{aligned} \tag{2.5}$$

This implies that $\frac{d}{dt}\Phi(t) \leq \epsilon_1(t)$ provided that

$$\sum_{i\in I_+}(\epsilon_i(t)+\pi_i) - \sum_{i\in I_-}(\epsilon_i(t)+\pi_i) \geq 0,$$

which clearly holds as this expression is equal to $(x_{i_1}(t)-x_{i_2}(t))+(x_{i_3}(t)-x_{i_4}(t))+\ldots$ and $x_i(t) \geq x_j(t)$ for $i < j$. Hence, $\frac{d}{dt}\Phi(t) \leq -|\epsilon_1(t)|$ if $\epsilon_1(t) < 0$. A similar argument can be used for $\epsilon_1(t) > 0$.

Finally, we consider the technical issue of defining $\frac{d}{dt}\Phi(t)$ in case $\epsilon_i(t) = 0$ for some $i$ and $t = t_0$. In this case the above proof remains unchanged provided that we

rely on the upper right-hand derivative (as in [4, Theorem 3]), that is, if we define $\frac{d}{dt}|\epsilon_i(t_0)|$ as

$$\frac{d}{dt}|\epsilon_i(t_0)| = \lim_{t \to t_0^+} \frac{|\epsilon_i(t)|}{t - t_0}.$$

□

The above lemma shows that the $L_1$-distance to the fixed point does not increase along any trajectory $x(t)$ in $E$, and only remains the same whenever $x_1(t) = \pi_1$ (as $\epsilon_1(t) = 0$ in such a case).

**Lemma 2.2.** *The only trajectory $x(t)$ of the ODE given by (2.1) and (2.2) with $x(0) \in E$ for which $x_1(t) = \pi_1$ for all $t$ is given by $x(t) = \pi$ for all $t$.*

*Proof.* If $x_1(t) = \pi_1 = \lambda$ for all $t$, then (2.1) implies that $x_2(t) = \pi_2$. Similarly, for $i \geq 2$, if $x_j(t) = \pi_j$ for all $j \leq i$ and $t$, then (2.2) implies that $x_{i+1}(t) = \pi_{i+1}$. □

We now recall La Salle's invariance principle for Banach spaces, where a (positively) invariant subset of $K \subset E$ of an ODE defined on $E$ is such that $x(t) \in K$ for all $t$ provided that $x(t)$ is the unique solution of the ODE with $x(0) \in K$.

**Theorem 2.3** ( [28])**.** *Let $V(x)$ be a continuous real valued function from $E$ to $\mathbb{R}$ with $\frac{d}{dt}V(x) = \limsup_{t \to 0^+} \frac{1}{t}(V(x(t)) - V(x)) \leq 0$, where $x(t)$ is the unique solution of an ODE with $x(0) = x$. Let $K = \{x \in E | \frac{d}{dt}V(x) = 0\}$ and let $M$ be the largest (positively) invariant subset of $K$. If $x(t)$ is precompact (i.e., remains in a compact set) for $x(0) \in E$, then*

$$\lim_{t \to \infty} dist(x(t), M) = 0,$$

*where $dist(x, M)$ represents the Banach distance between the point $x \in E$ and the set $M \subset E$.*

We are now in a position to prove the following Theorem:

**Theorem 2.4.** *All the trajectories of the set of ODEs given by (2.1) and (2.2), starting from $x \in E$ converge towards the unique fixed point $\pi$ in the weighted $L_1$-norm $\sum_{i \geq 1} \frac{|x_i|}{2^i}$.*

*Proof.* We rely on La Salle's invariance principle for Banach spaces by setting $V(x)$ equal to the $L_1$-distance to the fixed point, i.e., $V(x) = \sum_{i \geq 1} |x_i - \pi_i|$. Lemma 2.1 implies that $\frac{d}{dt}V(x) \leq 0$, while Lemma 2.2 shows that $M = \{\pi\}$ is a singleton. Hence, $\pi$ is a global attractor provided that we can show that the trajectory $x(t)$ remains in a compact set if $x(0) \in E$. Let $m = \sum_{i \geq 1} |x_i(0) - \pi_i|$, then by Lemma 2.1 we know that $x(t)$ remains in the set $E_m = \{x \in E | \sum_{i \geq 1} |x_i - \pi_i| \leq m\}$. This set is not compact in the Banach space $E$ equipped with the $L_1$-norm, but La Salle's invariance principle holds in any Banach space. If $E$ is equipped with the weighted $L_1$-norm $\sum_{i \geq 1} \frac{|x_i|}{2^i}$, the sets $E_m$ are compact and global attraction follows from Theorem 2.3. □

Due to the above theorem, we can now express the main performance measures of the pull, push and hybrid strategy via Theorem 2.2, where the overall probe rate $R$ for the pull, push and hybrid strategy are defined as $r(1 - \pi_1)$, $r\pi_2$ and $r_1(1 - \pi_1) + r_2\pi_2$, respectively.

**Corollary 2.1.** *The mean response time $D$ of a job under the push, pull and hybrid strategy equals*

$$D = 1 + \frac{\lambda}{(1-\lambda)(1+r)}.$$

*Under the hybrid strategy the overall probe rate $R$ can be expressed as*

$$R = (1-\lambda)r_1 + \frac{\lambda^2 r_2}{1+(1-\lambda)r},$$

*with $r = r_1 + r_2$. Setting $(r_1, r_2) = (r, 0)$ and $(0, r)$ results in the probe rate $R$ of the pull and push strategy, respectively.*

*Proof.* The mean response time $D$ can be expressed as $\sum_{i\geq 1} \pi_i/\lambda = 1+\lambda/(1+(1-\lambda)r-\lambda)$ by Little's law. The overall probe rate under the pull and push strategy equals $r(1-\pi_1)$ and $r\pi_2$, respectively. Under the hybrid strategy the overall probe rate equals $r_1(1-\pi_1) + r_2\pi_2$. □

Our interest lies in comparing the mean response time $D$ of the three policies given $\lambda$ and the overall allowed probe rate $R$. Using the above result, we can easily set $r$ such that the overall probe rate equals some predefined $R$. For the hybrid policy this still leaves one degree of freedom as only the sum of $r_1 + r_2$ has been determined. The above result also indicates that $R$ converges to $\lambda^2/(1-\lambda)$ as $r$ goes to infinity under the push strategy (which is in contrast to the pull strategy where $R$ also goes to infinity). This indicates that an overall probe rate $R$ close to $\lambda^2/(1-\lambda)$ suffices to get a mean response time close to 1 under the push strategy. We should however also note that this rate $R$ becomes large as $\lambda$ approaches one.

**Theorem 2.5.** *The mean response time $D$ of a job under the push strategy equals*

$$D_{push} = \frac{\lambda}{(1-\lambda)(\lambda+R)},$$

*for $R < \lambda^2/(1-\lambda)$ and $D_{push} = 1$ for $R \geq \lambda^2/(1-\lambda)$. Under the pull strategy we get*

$$D_{pull} = \frac{1+R}{1-\lambda+R}.$$

*Hence, given $\lambda$ the push strategy outperforms the pull strategy if and only if $(1+R) > \lambda^2/(1-\lambda)$ and given $R$ the push is the best strategy if and only if*

$$\lambda < \frac{\sqrt{(1+R)^2 + 4(1+R)} - (1+R)}{2}.$$

*Further, the push strategy outperforms the pull strategy for all $\lambda < \phi - 1$, where $\phi = (1+\sqrt{5})/2$ is the golden ratio.*

*Proof.* The expressions for $D_{push}$ and $D_{pull}$ are readily obtained from Corollary 2.1 by plugging in the appropriate value for $r$ in the expression for $D$. Requiring that $D_{push} = D_{pull}$ results in a quadratic equation for $R$ with roots in 0 and $\lambda^2/(1-\lambda)-1$, which results in the condition for $(1+R)$ and $\lambda$. The last result is obtained by noting that $\sqrt{(1+R)^2+4(1+R)}/2-(1+R)/2$ is an increasing function in $R$ and its limit for $R$ going to zero equals $\sqrt{5}/2 - 1/2$. □

Looking at the expression for the mean delay in Corollary 2.1, we note that a strategy with a lower mean response time actually has a larger $r$ value when matching $R$. By Theorem 2.2 we also know that the queue length distribution decays geometrically with parameter $\lambda/(1+(1-\lambda)r)$. Hence, a smaller mean delay therefore also implies a faster decay of the queue length distribution. In fact, in this case a smaller mean delay even implies that the queue length distribution becomes smaller in the usual stochastic ordering sense [29].

We observe another fundamental difference between the push and pull strategy when the load approaches 1. In this case the mean delay of the push strategy still goes to infinity as in the M/M/1 queue (the mean response time of which is $1/(1-\lambda)$). For the pull strategy the mean delay approaches $1+1/R$, hence remains finite. We should note that $r$ does go to infinity when $\lambda$ approaches 1 under the pull strategy (for any $R > 0$).

**Theorem 2.6.** *The mean delay under the hybrid strategy $(r_1, r_2)$ with overall probe rate $R$ is minimized by setting $r_1$ or $r_2$ equal to zero. Hence, a pure pull or push strategy is always optimal.*

*Proof.* Let $R_1$ and $R_2 = R - R_1$ be the overall probe rate generated by the pull and push operations, respectively. By Corollary 2.1, we have $R_1 = (1-\lambda)r_1$ and $R_2 = \lambda^2 r_2/(1+(1-\lambda)r)$, while we also note that $D$ is minimized by maximizing $r = r_1 + r_2$. Hence, by letting $R_2 = y$ and $R_1 = R - y$, we wish to maximize

$$g(y) = \frac{R-y}{1-\lambda} + \frac{y(1+R-y)}{\lambda^2-(1-\lambda)y},$$

for $y \in [0, R]$ and $R < \lambda^2/(1-\lambda)$. For $R \geq \lambda^2/(1-\lambda)$ the response time is minimized by setting $r_1 = 0$ as $D_{push} = 1$. Some basic algebraic manipulations show that

$$\frac{d}{dy}g(y) = \left((1+R) - \frac{\lambda^2}{1-\lambda}\right)\left(\frac{\lambda}{\lambda^2-(1-\lambda)y}\right)^2,$$

on $y \in [0, R]$ with $R < \lambda^2/(1-\lambda)$. Depending on the sign of $(1+R) - \lambda^2/(1-\lambda)$ the derivative of $g(y)$ is therefore positive or negative on the entire interval and the minimum is found in $y = 0$ (i.e., $r_2 = 0$) or $y = R$ (i.e., $r_1 = 0$). □

## 2.4 Model Validation

In this section we validate the infinite system model by comparing the closed form results of Theorem 2.5 with time consuming simulation results for systems with a finite number of nodes $N$. The infinite and finite system model only differ in the system size. Hence, the rate $r$ in the simulation experiments is independent of $N$ and was determined by $\lambda$ and $R$ using the expression for $R$ in Corollary 2.1. Each simulated point in the figures represents the average value of 25 simulation runs. Each run has a length of $10^6$ time units (where the service time is exponentially distributed with a mean of 1 time unit) and a warm-up period of length $10^6/3$ time units.

Figure 2.1 compares the mean delay in a finite system with $N$ nodes with the mean delay in the infinite system model under the push strategy with $R = 1$ for

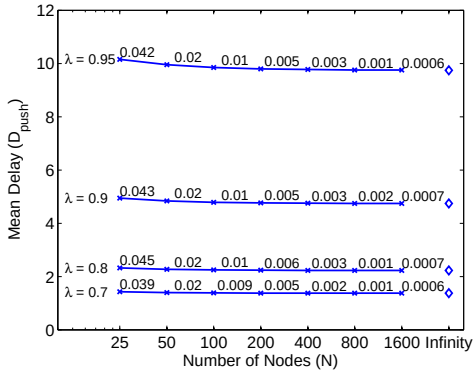


Figure 2.1: Mean delay and relative error of the push strategy in a finite vs. infinite system for $R = 1$

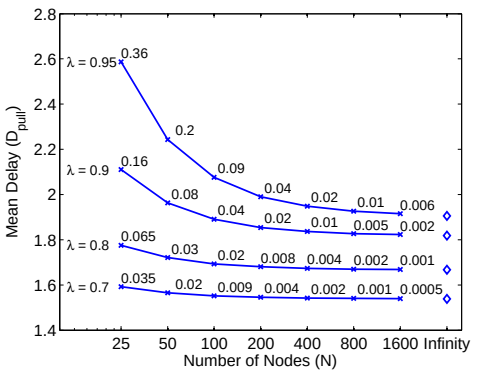


Figure 2.2: Mean delay and relative error of the pull strategy in a finite vs. infinite system for $R = 1$.

$N = 25, 50, \ldots, 1600$ and $\lambda = 0.7, 0.8, 0.9$ and $0.95$. For each combination of $N$ and $\lambda$ we also show the relative error. The error clearly decreases to zero as $N$ goes to infinity. Further, even for a system with $N = 100$ nodes we observe a relative error of 1% only. It may seem unexpected that the relative error is nearly insensitive to the load, as one might expect higher errors as $\lambda$ increases. In fact, if $r$ is kept fixed we would observe an increased error. However, we are looking at the curves for $R = 1$, meaning $r = 1/(\lambda^2 - (1-\lambda))$ decreases with $\lambda$ (see Corollary 2.1). As setting $r = 0$ gives exact results for any finite $N$, we can expect an improved accuracy for smaller $r$ values (if $\lambda$ remains fixed). Thus, in Figure 2.1 we see more or less the same relative errors because higher loads, which worsen the accuracy, correspond to lower $r$ values, which improve the accuracy.

Figure 2.2 depicts the same results as Figure 2.1, but for the pull strategy. Although we still see the convergence as $N$ goes to infinity, the relative errors grow quickly with $\lambda$ and an error of 9% is observed even for a system with $N = 100$ nodes. Under the pull strategy $r = 1/(1-\lambda)$ for $R = 1$, which implies that larger $\lambda$ values also correspond to larger $r$ values. Therefore, the less accurate results for higher loads are not unexpected.

The overall request rate observed in the simulation experiments was typically within 0.1% of the targeted $R$ value, meaning the relation $R = (1-\lambda)r$ seems highly accurate even for finite systems. This is not unexpected as the fraction of idle nodes should also match $(1-\lambda)$ in the finite system. Figure 2.3 shows the observed overall request rate for the push strategy, which exceeds the targeted value of $R$ and decreases as a function of $N$ and $\lambda$. Hence, the relation $R = \lambda^2 r/(1 + (1-\lambda)r)$ of Corollary 2.1 is not highly accurate for small system sizes. This can be explained by noting that the infinite system model is optimistic with respect to the queue length distribution for $N$ finite and therefore also predicts a lower overall probe rate.

In Figure 2.4 we compare the mean delay of the push and pull strategy in the infinite system model (full lines) with a finite system consisting of $N = 100$ nodes (crosses) for $\lambda \geq 0$ and $R = 0.5$ and $R = 1$. The results indicate that the infinite system model provides accurate results under any load $\lambda$, while the pull strategy becomes less accurate as the load increases (which is in agreement with the results in

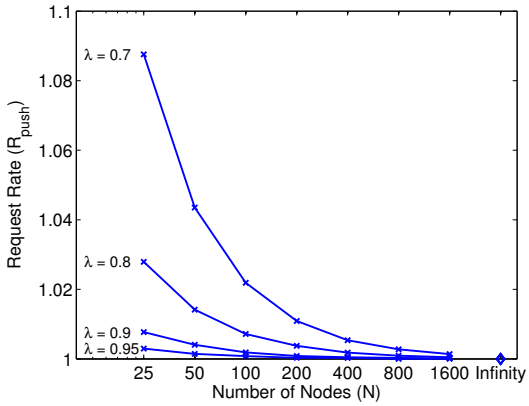


Figure 2.3: Overall request rate of the push strategy in a finite vs. infinite system for $R = 1$.

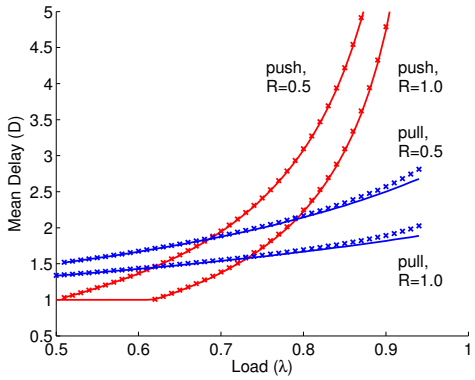


Figure 2.4: Mean delay of the push and pull strategy in a finite system with $N = 100$ nodes (crosses) vs. infinite system model (full lines) for $R = 0.5$ and $R = 1$.

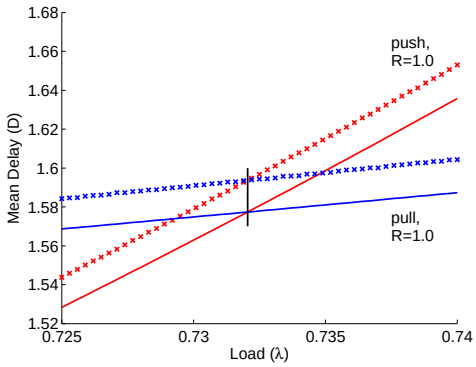


Figure 2.5: Mean delay of the push and pull strategy in a finite system with $N = 100$ nodes (crosses) vs. infinite system model (full lines) for $R = 1$.

Figures 2.1 and 2.2). Note, under the push strategy setting $\lambda < (\sqrt{5}-1)/2 \approx 0.6180$ implies that $r$ can be chosen arbitrarily large such that the overall probe rate $R$ remains below 1 (see Theorem 2.5). For $r = \infty$ the mean delay becomes 1 and there is little use in simulating the system for finite $N$.

In Figure 2.5 we have zoomed in on the intersection of the pull and push curves for $R = 1$ to indicate that the region where the push strategy outperforms the pull strategy is in perfect agreement with the infinite system model. This can be understood by noting that the $r$ value used during the simulation is determined by the relation between $R$ and $r$ in Corollary 2.1. When $\lambda = \sqrt{(1+R)^2 + 4(1+R)}/2 - (1+R)/2$, we therefore make use of the same $r$ value for the push and pull strategy. Hence, the evolution of the finite system model with $N$ nodes is captured by the same Markov chain $(X^{(N)}(t))_{t\geq 0}$, meaning both strategies have the same queue length distribution and mean delay for all $N$. We should however keep in mind

that the observed overall probe rate tends to exceed $R$ under the push strategy, especially for small systems. It is therefore fair to say that when $N$ is small, the region where the push strategy outperforms the pull strategy is in fact overestimated by the infinite system model.

## 2.5 Finite Versus Infinite System Model

In this section we discuss the relation between the set of ODEs in (2.1) and (2.2) and the sequence of Markov chains $\{X^{(N)}(t)\}_{t\geq 0}$ as $N$ tends to infinity. More specifically, we will identify and prove the technical issues related to formally showing that the steady state measures $\pi^{(N)}$ of $\{X^{(N)}(t)\}_{t\geq 0}$ converge to the unique fixed point $\pi$. These issues arise from having an infinite dimensional state space $E$. Replacing the infinite size buffer in each node by a finite large buffer (such that the loss rate can be neglected) would result in a finite dimensional (compact) space $E$ and would resolve most of the issues. This also explains why large finite buffers are often considered as opposed to infinite buffers (see [9, 10]).

We start by recalling the definition of a density dependent family of Markov chains [30]. A set of Markov chains $\{X^{(N)}(t)\}_{t\geq 0}$, with $N \geq 1$, where $E_N = E \cap \{k/N, k \in \mathbb{Z}^m\}$ is the state space of $\{X^{(N)}(t)\}_{t\geq 0}$, is a family of density dependent Markov chains provided that the transition rates $q^{(N)}(x, y)$ between state $x \in E_N$ and $y \in E_N$ can be written as

$$q^{(N)}(x, y) = N\beta_{(y-x)N}(x),$$

where $\beta_\ell$ is a function from $E \subset \mathbb{R}^m$ to $\mathbb{R}^+$. Let $F(x) = \sum_{\ell\in L} \ell\beta_\ell(x)$, where $L$ is the set of all possible transitions. Note, the set of CTMCs considered in Section 2.3 matches this definition, with $L = \{e_i, i \geq 1\} \cup \{-e_i, i \geq 1\} \cup \{e_1 - e_i, i \geq 2\}$, except that $E$ is not a part of $\mathbb{R}^m$ for some finite $m$. However, this definition was extended to $\mathbb{R}^\infty$ in [23], where the following generalization of Kurtz's theorem was proven [23, Theorem 3.13]:

**Theorem 2.7** (Kurtz)**.** *Consider a family of density dependent CTMCs, with $F$ Lipschitz. Let $\lim_{N\to\infty} X^{(N)}(0) = \tilde{x}$ a.s. and let $\phi_t(\tilde{x})$ be the unique solution to the initial value problem $\frac{d}{dt}x(t) = F(x(t))$ with $x(0) = \tilde{x}$. Consider the path $\{\phi_t(\tilde{x}), t \leq T\}$ for some fixed $T \geq 0$ and assume that there exists a neighborhood $K$ around this path satisfying*

$$\sum_{\ell\in L} |\ell| \sup_{x\in K} \beta_\ell(x) < \infty, \tag{2.6}$$

*then*

$$\lim_{N\to\infty} \sup_{t\leq T} \left| X^{(N)}(t) - \phi_t(\tilde{x}) \right| = 0 \quad a.s.$$

In the finite dimensional case, the set $L$ is finite and therefore (2.6) is automatically met. For our system, condition (2.6) corresponds to showing that there exists an environment $K$ such that $\sum_{i\geq 2} \sup_{x\in K}(x_i - x_{i+1}) < \infty$.

We start by proving the following lemma:

**Lemma 2.3.** *For any $T > 0$ and $x(0) \in E$, the unique solution $x(t) = (x_1(t), x_2(t), \ldots)$ to the initial value problem defined by (2.1) and (2.2) satisfies*

$$\sum_{i\geq 2} \sup_{0\leq t\leq T} x_i(t) \leq \exp(\lambda T) \left(1 + \sum_{i\geq 2} x_i(0)\right). \tag{2.7}$$

*Proof.* By (2.2) we have $\frac{d}{dt}x_i(t) \leq \lambda x_{i-1}(t)$ for $i \geq 2$. Hence, for $i \geq 2$

$$x_i(t) = x_i(0) + \int_{u=0}^{t} dx_i(u) \leq x_i(0) + \lambda \int_{u=0}^{t} x_{i-1}(u)du.$$

Combining the above inequality with the fact that $x_1(t) \leq 1$, readily allows us, by induction on $i$, to establish the following inequality:

$$x_i(t) \leq \sum_{j=2}^{i} x_j(0) \frac{(\lambda t)^{i-j}}{(i-j)!} + \frac{(\lambda t)^{i-1}}{(i-1)!}.$$

Interchanging the order of summation therefore yields

$$\sum_{i\geq 2} \sup_{0\leq t\leq T} x_i(t) \leq \sum_{j\geq 2} x_j(0) \sum_{i\geq j} \frac{(\lambda T)^{i-j}}{(i-j)!} + \sum_{i\geq 2} \frac{(\lambda T)^{i-1}}{(i-1)!},$$

which implies (2.7). □

We are now in a position to prove the following Theorem:

**Theorem 2.8.** *Given $\tilde{x} \in E$ and $T \geq 0$, there exists an environment $K$ of $\{\phi_t(\tilde{x}), t \leq T\}$ such that $\sum_{i\geq 2} \sup_{x\in K}(x_i - x_{i+1}) < \infty$.*

*Proof.* This readily follows from (2.7) by defining $K$ as

$$K = \{x \in E | \exists t, \forall i \geq 0 : |x_i - \tilde{x}_i(t))| < 2^{-i}\},$$

such that

$$\begin{aligned}\sum_{i\geq 2} \sup_{x\in K} x_i &< \sum_{i\geq 2} \left(\sup_{0\leq t\leq T} \tilde{x}_i(t) + 2^{-i}\right) \\ &\leq \exp(\lambda T)(1 + \sum_{i\geq 2} \tilde{x}_i(0)) + 1 < \infty.\end{aligned}$$

□

Hence, the set of ODEs given by (2.1) and (2.2) describes the proper limit process of the finite systems over any finite time horizon $[0, T]$.

A natural question is whether this convergence extends to the stationary regime. Sufficient conditions for the finite dimensional case can be found in [31]. We will instead rely on a more general result in [32], which considers a family of stochastic processes on some Polish space $E$, which includes the set of infinite dimensional,

separable and complete spaces. As $E = \{(x_1, x_2, \ldots) | x_i \in [0,1], x_i \geq x_{i+1}, i \geq 1, \sum_{j\geq 1} x_j < \infty\}$ is a subspace of the space $l1 = \{(x_1, x_2, \ldots) | \sum_{j\geq 1} |x_j| < \infty\}$, it is separable. $E$ is clearly also complete and therefore Polish. Let $\pi^{(N)} = (\pi_1^{(N)}, \pi_2^{(N)}, \ldots)$ be the unique stationary measure of the Markov chain $\{X^{(N)}(t)\}_{t\geq 0}$. Given that we have a unique solution $\phi_t(x)$ (which is continuous in $t$ for all $x$) and that convergence over finite time intervals occurs, Corollary 1 of [32] can be rephrased as:

**Theorem 2.9** (Benaïm, Le Boudec)**.** *Given that $\phi_t(x)$ is continuous in $x$ for all $t$ and that the sequence $(\pi^{(N)})_{N\geq 1}$ is tight, we have*

$$\lim_{N\to\infty} \lim_{t\to\infty} \left| X^{(N)}(t) - \pi \right| = 0,$$

*in probability.*

The sequence $(\pi^{(N)})_{N\geq 1}$ is tight if for every $\epsilon > 0$ there exists some compact set $K_\epsilon$ such that $\mathbb{P}\{\pi^N \in K_\epsilon\} > 1 - \epsilon$ for all $N$. Note, if $E$ is compact (as is often the case in finite dimension), tightness is immediate. In our case $E$ is not compact and we rely on the following theorem:

**Theorem 2.10.** *The sequence of measures $(\pi^{(N)})_{N\geq 1}$ is tight.*

*Proof.* Define the set $F_m \subset E$ as $F_m = \{x \in E | \sum_{i\geq 1} x_i \leq m\}$. If we consider the metric space $(E, \rho)$, where $\rho$ is the weighted $L_1$-norm $\sum_{i\geq 1} \frac{|x_i|}{2^i}$, then $F_m$ is compact for any $m > 0$. Note, it suffices to prove tightness in this metric space as Prokhorov's theorem holds in any separable metric space [33]. To prove that the measures $(\pi^{(N)})_{N\geq 1}$ are tight, we will show that for any $\epsilon > 0$, setting $m_\epsilon = \frac{1}{(1-\lambda)\epsilon}$ implies that $\mathbb{P}\{\pi^N \in F_{m_\epsilon}\} > 1 - \epsilon$ for all $N$.

We start by considering a modified system consisting of $N$ nodes in which we give preemptive priority to *local* jobs, that is, transferred jobs are interrupted whenever a local job arrives (and can be transferred to yet another node). Let $X_{i,mod}^{(N)}(t) \in \{0, \ldots, N\}$ be the number of nodes with *at least* $i$ jobs in the queue at time $t$ in the modified system. Due to the exponential job size durations we have $X_{i,mod}^{(N)}(t) = X_i^{(N)}(t)$, which implies that the modified system consisting of $N$ nodes has the same stationary measure $\pi^{(N)}$, meaning it suffices to prove tightness for the modified system.

Using the modified system consisting of $N$ nodes, we can now rely on a simple sample path argument to show that the length of queue $i$, for $i = 1, \ldots, N$, in the modified system is upper bounded by one plus the queue length of the $i$-th queue in a system consisting of $N$ independent M/M/1 queues. After all, in the modified system service to the local jobs is never prevented by a transferred job, each queue contains at most one transferred job and some local jobs may even be transferred. As the stationary queue length distribution in an M/M/1-queue is geometric with a mean equal to $\lambda/(1-\lambda)$, we may conclude that the mean queue length in the $i$-th node of the modified system is upper bounded by $1/(1-\lambda)$, for $i = 1, \ldots, N$.

Let $Y_{i,mod}^{(N)}(t)$ be the random variable representing the queue length of the $i$-th queue in the modified system at time $t$ and set $Y_{i,mod}^{(N)} = \lim_{t\to\infty} Y_{i,mod}^{(N)}(t)$, then

$$\frac{1}{N}\sum_{i=1}^{N} Y_{i,mod}^{(N)}(t) = \frac{1}{N}\sum_{i=1}^{N} X_{i,mod}^{(N)}(t).$$

Therefore, by the Markov inequality,

$$\begin{aligned}\mathbb{P}\{\pi^N \in F_{m_\epsilon}\} &= 1 - \mathbb{P}\left\{\frac{1}{N}\sum_{i=1}^{N} Y_{i,mod}^{(N)} > m_\epsilon\right\} \\ &\geq 1 - \mathbb{E}\left\{\frac{1}{N}\sum_{i=1}^{N} Y_{i,mod}^{(N)}\right\}/m_\epsilon \\ &\geq 1 - \frac{1}{(1-\lambda)m_\epsilon} = 1-\epsilon,\end{aligned}$$

with $m_\epsilon = \frac{1}{(1-\lambda)\epsilon}$ for all $N$. □

The continuity of $\phi_t(x)$ in $x$ for all $t$ is guaranteed by the uniqueness of the solution in finite dimensions, but this result does not in general extend to Banach spaces of infinite dimension [34]. However, for $F$ Lipschitz, as in our case, the classical finite dimensional results still hold and we may conclude that convergence of the steady state measures to the fixed point $\pi$ occurs.

## 2.6 Rate-based Versus Traditional Strategies

The aim of this section is to show that the performance of the rate-based pull/push strategies coincides with the traditional pull/push strategies when the former match the overall probe rate of the latter. To this end, we introduce an infinite system model for the following traditional pull and push strategy:

1. *Traditional Push:* A server starts sending probes whenever a job arrives in a queue with $i \geq 1$ jobs, meaning $i-1$ jobs are waiting to be served. The nodes that are probed are selected at random and a node is only allowed to accept a job if it is idle. The server starts by probing a single node. If the probe fails (because the selected node is not idle), the server sends another probe. This procedure is repeated until a probe is either successful or $L_p$ unsuccessful probes were sent.

2. *Traditional Pull:* A server starts sending probes whenever the server becomes idle. The nodes that are probed are selected at random and a node is only allowed to transfer a job if its queue length exceeds one. Probes are sent one at a time until one is successful or $L_p$ unsuccessful probes were sent.

Analytical models to assess the performance of a class of pull and push strategies that include the above two strategies were presented in [7, 18]. These models relied on a decoupling assumption and the mean response time was expressed as the solution to a nonlinear equation that was solved numerically.

In this section we present ODE models for the traditional strategies similar to the ODE model in Section 2.3 for the rate-based strategies and show that its unique fixed point can be expressed in closed form. It is not hard to verify that the nonlinear equation for the unique fixed point of the ODE corresponds to the nonlinear equation in [7] for the pull strategy with $T = 1$ and the one in [18] for the push strategy with $T = 1$ and $C = 0$.

### 2.6.1 Traditional Push

Let $s_i(t)$ denote the fraction of queues containing at least $i$ jobs at time $t$ and set $s(t) = (s_1(t), s_2(t), \ldots)$. The dynamics of the infinite system model for the traditional push strategy is captured by the following set of ODEs:

$$\frac{ds_1(t)}{dt} = \lambda(1 - s_1(t)) + \lambda s_1(t)(1 - s_1(t)^{L_p}) - (s_1(t) - s_2(t)) \tag{2.8}$$

$$\frac{ds_i(t)}{dt} = \lambda(s_{i-1}(t) - s_i(t))s_1(t)^{L_p} - (s_i(t) - s_{i+1}(t)) \tag{2.9}$$

for $i \geq 2$. The terms $s_i(t) - s_{i+1}(t)$, for $i \geq 1$, correspond to the service completion events (as the job durations are exponential with mean 1). The rate at which arrivals occur in a node with exactly $i - 1$ jobs is $\lambda(s_{i-1}(t) - s_i(t))$ and $s_1(t)^{L_p}$ is the probability that $L_p$ probes are unsuccessful; hence queues of length $i$ are created at rate $\lambda(s_{i-1}(t) - s_i(t))s_1(t)^{L_p}$, for $i \geq 2$. Finally, queues of length 1 are created by new arrivals (at rate $\lambda(1 - s_1(t))$) or job transfers. The latter occur at rate $\lambda s_1(t)(1 - s_1(t)^{L_p})$, as $\lambda s_1(t)$ is the rate at which probes are sent to the idle nodes and $(1 - s_1(t)^{L_p})$ is the probability that one of the probes is successful.

The set of ODEs given by (2.8) and (2.9) has a unique fixed point $\hat{\pi} = (\hat{\pi}_1, \hat{\pi}_2, \ldots)$ with $\sum_{i \geq 1} \hat{\pi}_i < \infty$ given by

$$\begin{aligned}
&\hat{\pi}_1 = \lambda, \\
&\hat{\pi}_2 = \lambda^{L_p+2}, \\
&\hat{\pi}_{i+1} = \hat{\pi}_i - \lambda^{L_p+1}(\hat{\pi}_{i-1} - \hat{\pi}_i),
\end{aligned}$$

for $i > 2$, where the first equality follows from taking the sum of (2.8) and (2.9) for $i \geq 1$. We can simplify this further to $\hat{\pi}_i = \lambda^{(i-1)(L_p+1)+1}$.

For the traditional push strategy, every busy node will send on average

$$1 + \sum_{i=1}^{L_p-1} \hat{\pi}_1^i = \frac{1 - \lambda^{L_p}}{1 - \lambda}$$

probes at the task arrival rate $\lambda$, meaning that the overall probe rate $\hat{R}$ equals

$$\hat{R} = \hat{\pi}_1 \lambda \frac{1 - \lambda^{L_p}}{1 - \lambda}.$$

From the relationship $R = r_{push}\pi_2$, we observe that a rate-based push strategy with

$$r_{push} = \frac{\lambda \hat{\pi}_1}{\pi_2} \frac{1 - \lambda^{L_p}}{1 - \lambda}$$

matches $\hat{R}$. By substituting the rate $r_{push}$ in Theorem 2.2 we find that under the infinite system model, the traditional and rate-based push strategy have the same fixed point. This indicates that the rate-based strategy matches the queue length distribution of the traditional variant, provided that we match the overall probe rate $R$.

### 2.6.2 Traditional Pull

A similar set of ODEs describes the evolution of the traditional pull strategy (for $L_p = 1$ this corresponds to the model in [22]):

$$\frac{ds_1(t)}{dt} = \lambda(1 - s_1(t)) - (s_1(t) - s_2(t))(1 - s_2(t))^{L_p} \tag{2.10}$$

$$\frac{ds_i(t)}{dt} = \lambda(s_{i-1}(t) - s_i(t)) - (s_i(t) - s_{i+1}(t)) - \frac{(s_i(t) - s_{i+1}(t))}{s_2(t)}(s_1(t) - s_2(t))(1 - (1 - s_2(t))^{L_p}), \tag{2.11}$$

for $i \geq 2$, where $\frac{ds_i(t)}{dt} = \lambda(s_{i-1}(t) - s_i(t))$ if $s_2(t) = 0$ and $i \geq 2$. The intuition is similar as for the push strategy, where we note that $(s_1(t) - s_2(t))$ is the rate at which probes are generated, $(1 - (1 - s_2(t))^{L_p})$ is the probability that one of the probes is successful and $(s_i(t) - s_{i+1}(t))/s_2(t)$ is the probability that the accepting busy queue has length $i$, for $i \geq 2$.

The system given by (2.10) and (2.11) also has a unique fixed point $\tilde{\pi} = (\tilde{\pi}_1, \tilde{\pi}_2, \ldots)$ with $\sum_{i \geq 1} \tilde{\pi}_i < \infty$ . As $\tilde{\pi}_1 = \lambda$, $\tilde{\pi}_2$ is found as the unique positive real root of

$$g(x) = \lambda(1 - \lambda) - (\lambda - x)(1 - x)^{L_p} = 0,$$

with $x \in [0, \lambda]$. The uniqueness follows by noting that $dg(x)/dx = (1 - x)^{L_p - 1}(1 - x + L_p(\lambda - x))$ is strictly positive on $[0, \lambda]$, while $g(0) < 0$ and $g(\lambda) > 0$ (as $\lambda < 1$). Finally, $\tilde{\pi}_i$, for $i > 2$, is given by

$$\tilde{\pi}_{i+1} = \tilde{\pi}_i - \frac{\lambda(\tilde{\pi}_{i-1} - \tilde{\pi}_i)}{1 + (\lambda - \tilde{\pi}_2)(1 - (1 - \tilde{\pi}_2)^{L_p})/\tilde{\pi}_2}.$$

This recurrence can be solved to

$$\tilde{\pi}_i = \frac{(\lambda - \tilde{\pi}_2)(\lambda\tilde{\pi}_2)^i \alpha^{2-i} + \lambda\tilde{\pi}_2(\alpha - \lambda^2)}{\lambda\tilde{\pi}_2(\alpha - \lambda\tilde{\pi}_2)},$$

with $\alpha = \lambda - (1 - \tilde{\pi}_2)^{L_p}(\lambda - \tilde{\pi}_2)$.

For the traditional pull strategy, every node with exactly one job will send on average

$$1 + \sum_{i=1}^{L_p - 1} (1 - \tilde{\pi}_2)^i = \frac{1 - (1 - \tilde{\pi}_2)^{L_p}}{\tilde{\pi}_2}$$

probes each time the server becomes idle (as a probe fails with probability $(1 - \tilde{\pi}_2)$), meaning that the overall probe rate $\tilde{R}$ for the traditional pull strategy equals

$$\tilde{R} = (\tilde{\pi}_1 - \tilde{\pi}_2)\frac{1 - (1 - \tilde{\pi}_2)^{L_p}}{\tilde{\pi}_2}.$$

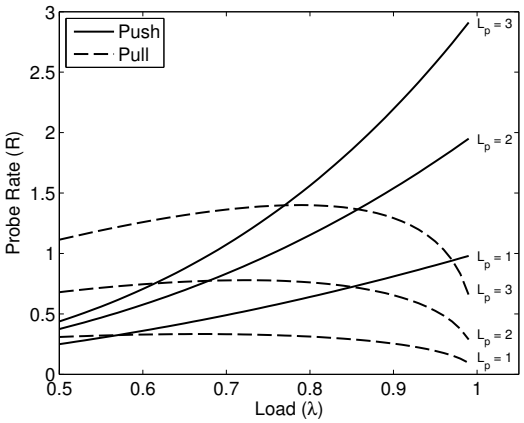


Figure 2.6: Mean overall probe rate for the traditional pull and push strategy as a function of $\lambda$ for $L_p = 1, 2$ and 3.

From the relationship $R = (1 - \pi_1)r_{pull}$, we observe that a rate-based pull strategy with

$$r_{pull} = \frac{\tilde{\pi}_1 - \tilde{\pi}_2}{1 - \pi_1} \frac{1 - (1 - \tilde{\pi}_2)^{L_p}}{\tilde{\pi}_2}$$

amounts to the same overall probe rate $\tilde{R}$. Substituting the rate $r_{pull}$ in Theorem 2.2 allows us to conclude that the rate-based pull strategy matches the queue length distribution of the traditional variant, provided that we match the overall probe rate $R$.

The mean overall probe rate $\hat{R}$ and $\tilde{R}$ for the traditional push and pull strategy respectively is shown in Figure 2.6. As the probe rate of both strategies differs significantly, it is sometimes hard to compare these strategies in a fair manner.

## 2.7 Conclusion

In this chapter we compared the ability of the push and pull strategy to reduce the mean delay in a homogeneous distributed system given an overall probe rate $R$. We showed that the push strategy outperforms the pull strategy if and only if $\lambda < \sqrt{(R+1)^2 + 4(R+1)}/2 - (R+1)/2$ in the infinite system model and showed, by simulation, that this formula is also quite accurate for small finite systems, e.g., systems with $N = 25$ nodes. We further demonstrated that a hybrid strategy is always inferior to the pure push or pull strategy when the overall probe rate $R$ is not allowed to increase. Some technical issues to formally prove the convergence of the steady state measures of the finite system model to the infinite system model were identified and proven. Finally, we showed that rate-based strategies matching the probe rate of traditional strategies, also match the queue length distribution.

Chapter 3

# Improved Rate-Based Pull and Push Strategies in Large Distributed Networks

*Large distributed systems benefit from the ability to exchange jobs between nodes to share the overall workload. To exchange jobs, nodes rely on probe messages that are either generated by lightly-loaded or highly-loaded nodes, which corresponds to a so-called pull or push strategy. A key quantity of any pull or push strategy, that has often been neglected in prior studies, is the resulting overall probe rate. If one strategy outperforms another strategy in terms of the mean delay, but at the same time requires a higher overall probe rate, it is unclear whether it is truly more powerful.*

*In this chapter*[1] *we introduce a new class of rate-based pull and push strategies that can match any predefined maximum allowed probe rate, which allows one to compare the pull and push strategy in a fair manner. We derive a closed form expression for the mean delay of this new class of strategies in a homogeneous network with Poisson arrivals and exponential job durations under the infinite system model. We further show that the infinite system model is the proper limit process over any finite time scale as the number of nodes in the system tends to infinity and that the convergence extends to the stationary regime.*

*Simulation experiments confirm that the infinite system model becomes more accurate as the number of nodes tends to infinity, while the observed error is already around* 1% *for systems with as few as* 100 *nodes.*

[1] This chapter is based on work published in MASCOTS 2013 as "Improved Rate-Based Pull and Push Strategies in Large Distributed Networks" by Wouter Minnebo and Benny Van Houdt [35].

## 3.1 Introduction

To evaluate and compare the different strategies considered in this chapter we introduce an infinite system model, the evolution of which is described by a set of ordinary differential equations (ODEs) as in [22] and Chapter 2. To assess the mean delay and overall probe rate of a strategy, we define a set of ODEs, give an explicit expression for its unique fixed point and express the mean delay and probe rate using this fixed point. To guarantee all trajectories converge to the fixed point, we prove that the fixed point is a global attractor. We also show that the set of ODEs captures the evolution of the limit process of a family of density dependent Markov chains as introduced by Kurtz in [30, 36]. Simulation experiments confirm that the infinite system model becomes exact as the number of nodes in the system tends to infinity, while the error is about 1% for systems with as few as 100 nodes.

This chapter focuses on the following points:

1. We introduce a more general class of rate-based pull and push strategies that rely on two parameters $T$ and $r$ and that coincide with the strategies introduced in Chapter 2 when $T = 1$. Closed form results for the mean delay of this new class of pull and push strategies are presented (under the infinite system model).

2. We show that setting $T > 1$ reduces the mean delay of the rate-based push strategy (for larger $\lambda$ and smaller $R$ values). This is in contrast to earlier findings for the traditional strategy [8, 19], for which smaller $T$ values result in higher probe rates, making the comparison biased. For the rate-based pull strategy we show that setting $T = 1$ is optimal.

3. We introduce the so-called max-push strategy and derive a closed form expression for its mean delay (under the infinite system model). We show that the max-push strategies further reduce the mean delay of the best rate-based pull and push strategies with $T \geq 1$ for certain combinations of $(\lambda, R)$.

4. Finally, we prove that the infinite system models introduced in this chapter are the proper limit processes of the finite stochastic systems with $N$ nodes as $N$ tends to infinity over any finite time scale. In addition, we prove that the convergence extends to the stationary regime (i.e., the ODEs have a global attractor).

The chapter is structured as follows. Section 3.2 introduces the rate-based pull and push strategies. For the rate-based strategies with $T \geq 1$ we present the infinite system model in Section 3.3. In Section 3.4 we validate this model using simulation results and present some numerical examples that compare the performance of the rate-based pull and push strategy. Section 3.5 introduces the max-push strategy and its infinite system model, while numerical results for the the max-push strategy are presented in Section 3.6.

## 3.2 Pull and Push Strategies

We consider a continuous-time system consisting of $N$ queues, where each queue consists of a single server with an infinite buffer. As in [7, 9, 19, 22], jobs arrive locally according to a Poisson process with rate $\lambda < 1$, and have an exponentially distributed duration with mean 1. Servers process jobs in a first-come first-served order. Servers can send probe messages to each other to query for queue length information, and to transfer jobs. We assume that the time required to transfer probe messages and jobs is sufficiently small in comparison with the processing time of a job, i.e., transfer times are considered zero.

We consider the following load balancing strategies that all make use of two parameters: an integer $T \geq 1$ and a real number $r > 0$. We note that the rate-based strategies with $T = 1$ were initially introduced in Chapter 2.

1. *Rate-based Push:* As soon as the queue length exceeds $T$, a server starts to generate probe messages according to a Poisson process with rate $r$. Whenever the queue length drops below $T$, this process is interrupted until the queue length exceeds $T$ again. The node that is probed is selected at random and is only allowed to accept a job if it is idle.

2. *Rate-based Pull:* Whenever a server is idle it generates probe messages according to a Poisson process with rate $r$. This process is interrupted whenever the server is busy. The node that is probed is selected at random and is only allowed to transfer one of its jobs if its queue length exceeds $T$.

For each of the above strategies transferred jobs are immediately served by the accepting node, hence any job transfer results in a reduction of the mean delay. To make the comparison fair the mean overall probe rate $R$ should be identical. The rate $R$ is defined as the mean number of probes that is sent by a server per time unit irrespective of its queue length, where $R$ is clearly less than $r$. Further on we will show that $r$ can be set in such a manner that it can match any predefined $R \geq 0$.

## 3.3 Rate-based strategies with $T \geq 1$

In this section we introduce the infinite system model to assess the performance of the rate-based strategies with $T \geq 1$. This model, the evolution of which is captured by a set of ODEs, is validated by simulation in Section 3.4, while in Section 3.7 it is argued to be the proper limit process of the stochastic finite system model with $N$ nodes as $N$ tends to infinity.

The evolution of both the rate-based pull and push strategy model is given by a set of ODEs denoted as $\frac{d}{dt}x(t) = F(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$. As explained below, this set of ODEs can be written as

$$\frac{dx_1(t)}{dt} = (\lambda + rx_{T+1}(t))(1 - x_1(t)) - (x_1(t) - x_2(t)), \tag{3.1}$$

$$\frac{dx_i(t)}{dt} = \lambda(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)), \tag{3.2}$$

for $2 \leq i \leq T$ and

$$\frac{dx_i(t)}{dt} = \lambda(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)) - r(1 - x_1(t))(x_i(t) - x_{i+1}(t)), \quad (3.3)$$

for $i > T$. The terms $\lambda(x_{i-1}(t) - x_i(t))$ and $x_i(t) - x_{i+1}(t)$, for $i \geq 1$, correspond to arrival and service completions, respectively. Under the pull strategy probes are sent at rate $r(1 - x_1(t))$ and a probe is successful with probability $x_{T+1}(t)$, while under the push strategy the probe rate equals $rx_{T+1}(t)$ and a probe is successful with probability $(1 - x_1(t))$. Hence, for both strategies queues of length 1 are created by job transfers at rate $rx_{T+1}(t)(1 - x_1(t))$. Similarly, job transfers reduce the number of queues with exactly $i$ jobs, for $i > T$, at rate $r(1 - x_1(t))(x_i(t) - x_{i+1}(t))$ under both strategies.

Let $E = \{(x_1, x_2, \ldots) | x_i \in [0, 1], x_i \geq x_{i+1}, i \geq 1, \sum_{j \geq 1} x_j < \infty\}$. The next two theorems show that this set of ODEs is Lipschitz on $E$ and it has a unique fixed point in $E$.

**Theorem 3.1.** *The function $F$ is Lipschitz on $E$.*

*Proof.* $F$ is Lipschitz provided that for all $x, y \in E$ there exists an $L > 0$ such that $|F(x) - F(y)| \leq L\,|x - y|$. By definition of $F(x)$ one finds

$$|F(x) - F(y)| \leq 2(\lambda + 1 + 2r)\,|x - y| + 2r \sum_{i>T} |x_1(x_i - x_{i+1}) - y_1(y_i - y_{i+1})|\,.$$

The above sum can be bounded by

$$\sum_{i>T} |(x_1 - y_1)(x_i - x_{i+1}) + y_1(x_i - x_{i+1} - y_i + y_{i+1})|\,,$$

which is bounded by $2\,|x - y|$ on $E$. Hence, $F$ is Lipschitz by letting $L = 2\lambda + 2 + 8r$. □

As $E$ is a Banach space the Lipschitz condition of $F$ suffices to guarantee that the set of ODEs $\frac{d}{dt}x(t) = F(x(t))$, with $x(0) \in E$, has a unique solution[2] $\phi_t(x(0))$ [27, Section 1.1].

**Theorem 3.2.** *The set of ODEs given by* (3.1) *to* (3.3) *has a unique fixed point $\bar{\pi} = (\bar{\pi}_1, \bar{\pi}_2, \ldots)$ with $\sum_{i \geq 1} \bar{\pi}_i < \infty$. Let $\eta_i = \bar{\pi}_i - \bar{\pi}_{i+1}$ and $\eta_0 = 1 - \lambda$, then the fixed point can be expressed as*

$$\begin{aligned}
\eta_1 &= \frac{\lambda(1 + (1 - \lambda)r - \lambda)}{1 + (1 - \lambda^T)r}, \\
\eta_i &= \eta_1 \lambda^{i-1}, && 2 \leq i \leq T, \\
\eta_i &= \eta_T \left(\frac{\lambda}{1 + (1 - \lambda)r}\right)^{i-T}, && i > T.
\end{aligned}$$

[2]The solution $\phi_t(x)$ belongs to the class of continuously differentiable functions as in the finite dimensional case.

*Proof.* Assume $\bar{\pi}$ is a fixed point with $\sum_{i\geq 1} \bar{\pi}_i < \infty$, meaning $F_i(\bar{\pi}) = 0$ for $i \geq 1$, where $F(x) = (F_1(x), F_2(x), \ldots)$. When $\sum_{i\geq 1} \pi_i < \infty$, we can simplify $\sum_{i\geq 1} F_i(\pi) = 0$ to $\lambda - \bar{\pi}_1 = 0$. Hence, $\bar{\pi}_1$ must equal $\lambda$. The expressions for $\eta_i$ then readily follow from the conditions $F_i(\bar{\pi}) = 0$, for $i \geq 1$. □

This fixed point is also the unique solution of the Kolmogorov differential equation for a state dependent M/M/1 queue with $\lambda_0 = \lambda + r\bar{\pi}_{T+1}$, $\lambda_i = \lambda$, for $i \geq 1$, $\mu_i = 1$, for $i = 1, \ldots, T$, and $\mu_i = 1 + (1-\lambda)r$, for $i \geq T+1$. The arrival process of such an M/M/1 queue is Poisson with rate $\lambda_i$ and the service is exponential with rate $\mu_i$ whenever the queue length equals $i$.

The set of ODEs in (3.1) to (3.3) describes the transient evolution of the infinite system, while we are in fact interested in its behavior as $t$ goes to infinity. Thus, we are interested in the limit of all the trajectories of this set of ODEs.

**Theorem 3.3.** *All the trajectories of the set of ODEs given by* (3.1) *to* (3.3)*, starting from $x \in E$ converge towards the unique fixed point $\pi$.*

The proof follows the same reasoning as the proof of Theorem 2.4, and can be found in Appendix A.

Due to the above theorem, we can now express the main performance measures of the push and pull strategies with $T \geq 1$ via Theorem 3.2:

**Corollary 3.1.** *The mean delay $D$ of a job under the push or pull strategy equals*

$$D_{both} = \frac{1}{1-\lambda} - \frac{r\lambda^T \left(\frac{\lambda}{(1-\lambda)(1+r)} + T\right)}{1 + r(1-\lambda^T)}.$$

*Proof.* Using Theorem 3.2, one can apply Little's law to express the mean response time $D$ as $\sum_{i\geq 1} \pi_i/\lambda$. □

**Corollary 3.2.** *Given a predefined maximum allowed probe rate $R$, the rate $r$ must be set as*

$$r_{pull} = \frac{R}{1-\lambda}, \tag{3.4}$$

$$r_{push} = \frac{R}{\lambda^{T+1} - (1-\lambda^T)R}, \tag{3.5}$$

*with $r_{push} = \infty$ for $R > \lambda^{T+1}/(1-\lambda^T)$. Hence, if the predefined value of $R$ exceeds $\lambda^{T+1}/(1-\lambda^T)$, the rate $r_{push}$ can be set arbitrarily high.*

*Proof.* From the relationships $R = (1-\bar{\pi}_1)r_{pull}$ and $R = r_{push}\bar{\pi}_{T+1}$, we find

$$R = (1-\lambda)r_{pull},$$

and

$$R = \frac{\lambda^{T+1}}{(1-\lambda^T) + 1/r_{push}}.$$

Here we observe that as $r_{push}$ tends to infinity, $R$ remains finite. In this case $\bar{\pi}_{T+1}$ tends to zero (see Theorem 3.2 or Equation 4.1). □

**Theorem 3.4.** *The mean response time $D$ of a job under the push stategy equals*

$$D_{push} = \frac{1-R}{1-\lambda} - \frac{RT}{\lambda} + \frac{R^2}{\lambda^T(1-\lambda)(\lambda+R)},$$

*if $R \leq \lambda^{T+1}/(1-\lambda^T)$, while for $R > \lambda^{T+1}/(1-\lambda^T)$ the rate $r_{push} = \infty$, and the mean delay $D_{push}$ is given by:*

$$D_{push|r=\infty} = \frac{1}{1-\lambda} - \frac{T\lambda^T}{1-\lambda^T}.$$

*Remark, $D_{push|r=\infty} = 1$ for $T = 1$.*
*Under the pull strategy the mean response time equals*

$$D_{pull} = \frac{1}{1-\lambda} - \frac{R\lambda^T(\frac{\lambda}{1-\lambda+R} + T)}{1-\lambda+R(1-\lambda^T)}.$$

*Proof.* The expressions for $D_{push}$ and $D_{pull}$ are found using Corollary 3.1, by plugging in the appropriate value for $r$, given by Corollary 3.2, in the expression for $D_{both}$. □

**Theorem 3.5.** *The optimal choice for the rate-based pull strategy is $T = 1$.*

*Proof.* It can be verified that increasing $T$ by one will increase $D_{pull}$ if and only if

$$\frac{\lambda/(1-\lambda+R)+T}{1-\lambda+R(1-\lambda^T)} \geq \lambda\frac{\lambda/(1-\lambda+R)+T+1}{1-\lambda+R(1-\lambda^{T+1})},$$

which is equivalent to stating

$$T(1-\lambda+R)(1-\lambda) \geq \lambda R(1-\lambda^T).$$

This condition can be rewritten as

$$T(1-\lambda+R) \geq R\sum_{i=1}^{T}\lambda^i,$$

which holds as $\lambda^i < 1$, for $i = 1, \ldots, T$. Hence, increasing $T$ by one always increases the mean delay of the rate-based pull strategy. □

Although it may at first seem sensible to steal jobs from *long* queues only if the maximum allowed probe rate $R$ is low, the pull strategy is in some sense blind as it also needs to send probes to locate these *long* queues. This is contrary to the push strategy, where nodes will only probe if their queue is *long*, as a result setting $T = 1$ is not always optimal for the rate-based push strategy (see numerical results in Section 3.4).

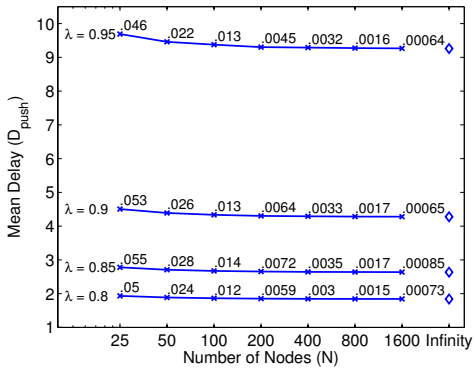


Figure 3.1: Simulated mean delay for a finite system of varying size, using a rate-based push strategy with $T = 2$, matching an overall request rate of $R = 1$. The relative error, shown above a simulated point, becomes smaller when simulating larger systems. In addition, the infinite system model approximates systems of moderate size well.

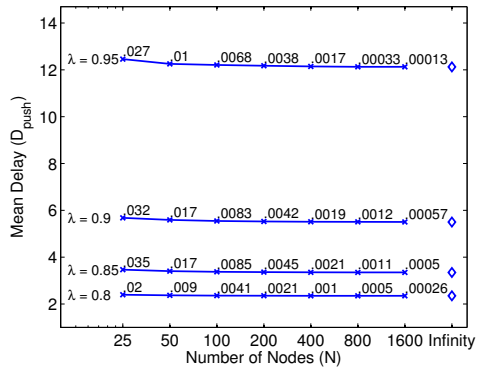


Figure 3.2: Simulated mean delay for a finite system of varying size, using a rate-based push strategy with $T = 4$, matching an overall request rate of $R = 0.5$. The relative error, shown above a simulated point, becomes smaller when simulating larger systems.

## 3.4 Numerical Results for Strategies with $T \geq 1$

### 3.4.1 Validation

In this section we present validation results for the rate-based push strategy with $T \geq 2$ as the model for both rate-based strategies with $T = 1$ was already validated in Chapter 2 and the mean delay of the pull strategy is minimized for $T = 1$. The infinite system model and simulation setup only differ in the system size. The rate $r_{push}$ in the simulation experiments is independent of $N$ and was determined by $\lambda$ and $R$ using the expression for $R$ in (3.5). Each simulated point in the figures represents the average value of 25 simulation runs. Each run has a length of $10^6$ (where the service time is exponentially distributed with mean 1) and a warm-up period of length $10^6/3$.

Figure 3.1 depicts the mean delay as a function of $N$ for $T = 2$, $R = 1$ and $\lambda = 0.8, 0.85, 0.9$ and $0.95$, while Figure 3.2 depicts the same for $T = 4$, $R = 0.5$. In both cases the relative error shown above the simulation results decreases as $N$ tends to infinity. The error for a system with as few as 100 nodes is only slightly above 1% when $T = 2$. We should note that the actual overall probe rate observed in the finite system exceeds $R$ for smaller $N$ values as shown in Figure 3.3 and 3.4. In other words, the relation between $R$ and $r_{push}$ given by (3.5) is not very accurate for small $N$ values as the infinite model is optimistic with respect to the queue length distribution. However, as soon as the system consists of several hundred nodes, there is a fairly good agreement. Similar results were observed for other parameter settings.

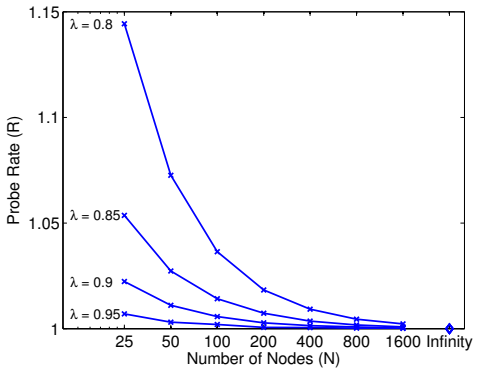


Figure 3.3: Request rate for the finite system using a rate-based push strategy with $T = 2$.

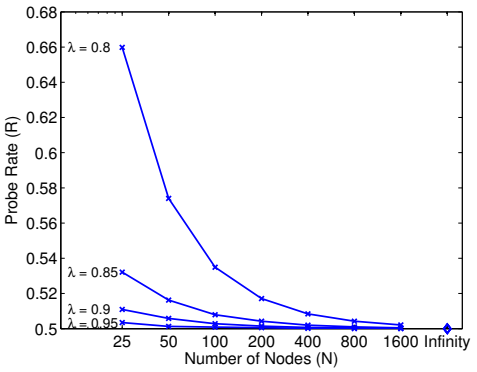


Figure 3.4: Request rate for the finite system using a rate-based push strategy with $T = 4$.

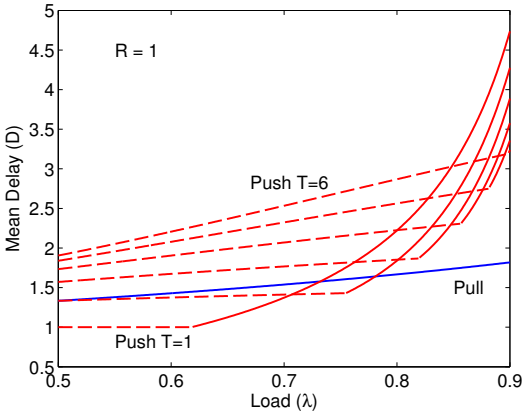


Figure 3.5: The mean delay for the rate-based pull ($T = 1$) and push strategy ($T = 1, \ldots, 6$) with $R = 1$. For the push strategy the part of the curve with $r = \infty$ is dashed.

### 3.4.2 Comparison of Push and Pull Strategy

The mean delay of the rate-based push (for $T = 1, \ldots, 6$) and pull (for $T = 1$) strategy is shown in Figure 3.5 as a function of $\lambda$ for a mean overall probe rate $R = 1$. The curves for the push strategy consist of two parts and $r_{push} = \infty$ for the dashed part of the curve. For these loads $\lambda$ the rate $r_{push}$ can be set arbitrarily high without violating the maximum allowed probe rate $R = 1$. The results indicate that the optimal $T$ value for the push strategy increases as $\lambda$ increases (while $R$ remains fixed). This is as expected as small $T$ values allow queues with a length below average to probe for idle servers, using part of the available probe rate. For the same reason smaller $R$ values also give rise to larger optimal $T$ values (for fixed $\lambda$). Figure 3.5 also indicates that setting $T > 1$ implies that the rate-based push strategy can outperform the pull strategy for a larger range of loads $\lambda$.

The rate-based strategy (with variable $T$) that minimizes the mean delay for $\lambda \in [0.5, 0.9]$ and $R \in [0, 2]$ is depicted in Figure 3.6. The pull strategy is superior for loads above 75 to 80%. For lower loads the push strategy prevails and lower maximum allowed probe rates $R$ give rise to larger optimal $T$ values.

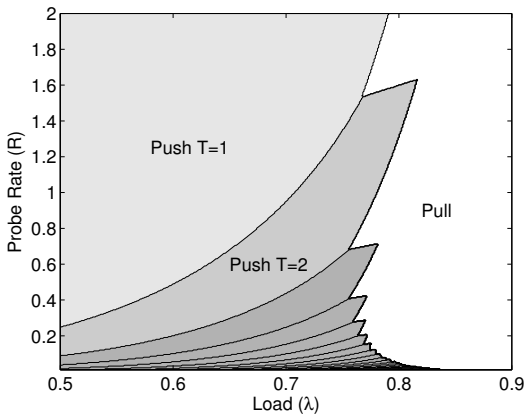


Figure 3.6: The optimal rate-based strategy with variable $T$ as a function of the load $\lambda$ and the overall probe rate $R$. The pull strategy is optimal for high loads. The optimal $T$ for the push strategy increases as $R$ decreases.

## 3.5 The Max-push Strategy

The mean delay under the rate-based push strategy given in Theorem 3.4 can be further reduced as follows. Recall, whenever $R > \lambda^{T+1}/(1-\lambda^T)$, the rate $r_{push}$ can be chosen arbitrarily large (i.e., $r_{push} = \infty$). In other words, even if requests are sent at infinite rate when the queue length exceeds $T$, the overall probe rate remains below $R$. Hence, in order to use this remaining request rate, we introduce the max-push strategy when $T > 1$ and

$$\lambda^{T+1}/(1-\lambda^T) < R < \lambda^T/(1-\lambda^{T-1}). \tag{3.6}$$

Note, for any $R > 0$ and $0 < \lambda < 1$, there exists a single $T > 0$ such that the above relationship holds. Under the max-push strategy we send probes at an infinite rate whenever the queue length exceeds $T$ and at rate $r < \infty$ if the queue length equals $T$. Note, under this strategy jobs are instantaneously transferred to another queue if the queue length equals $T$ upon arrival (at the expense of a number of probe messages). The evolution of the infinite system model for this strategy is also readily formulated as a set of ODEs $\frac{d}{dt}x(t) = G(x(t))$, where $x(t) = (x_1(t), \ldots, x_T(t))$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$:

$$\frac{dx_1(t)}{dt} = \lambda(1 - x_1(t) + x_T(t)) - (x_1(t) - x_2(t)) + rx_T(t)(1 - x_1(t)) \tag{3.7}$$

$$\frac{dx_i(t)}{dt} = \lambda(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)), \tag{3.8}$$

for $2 \leq i \leq T-1$ and

$$\frac{dx_T(t)}{dt} = \lambda(x_{T-1}(t) - x_T(t)) - x_T(t)(1 + r(1 - x_1(t))). \tag{3.9}$$

The terms of the form $\lambda(x_{i-1}(t) - x_i(t))$ and $(x_i(t) - x_{i+1}(t))$, for $1 \leq i \leq T$, are again due to arrival and service completion events, respectively. Additionally,

queues of length 1 are created at rate $\lambda x_T(t)$ due to the instantaneous job transfers and rate $rx_T(t)(1 - x_1(t))$ due to successful probes sent by a queue of length $T$, while the latter event also reduces the number of queues of length $T$ by one.

Let $E_T = \{(x_1, \ldots, x_T) | 1 \geq x_1 \geq x_2 \geq \ldots x_T \geq 0\}$. The next two theorems show that this set of ODEs is Lipschitz on $E_T$ and it has a unique fixed point in $E_T$.

**Theorem 3.6.** *The function $G$ is Lipschitz on $E_T$.*

*Proof.* $G$ is Lipschitz provided that for all $x, y \in E_T$ there exists an $L > 0$ such that $|G(x) - G(y)| \leq L\,|x - y|$. By definition of $G(x)$ one finds

$$|G(x) - G(y)| \leq 2(2\lambda + 1 + 2r)\,|x - y|\,.$$

Hence, $G$ is Lipschitz by letting $L = 4\lambda + 2 + 4r$. □

As $E_T$ is a finite dimensional space the Lipschitz condition of $G$ suffices to guarantee that the set of ODEs $\frac{d}{dt}x(t) = G(x(t))$, with $x(0) \in E_T$, has a unique solution (due to the Picard Lindelöf theorem).

**Theorem 3.7.** *The set of ODEs given by* (3.7)-(3.9) *has a unique fixed point $\dot{\pi} = (\dot{\pi}_1, \ldots, \dot{\pi}_T)$ in $E_T$ that can be expressed as*

$$\dot{\pi}_i = \lambda^i \frac{1 + (\frac{\lambda}{1-\lambda} + r)(1 - \lambda^{T-i})}{1 + (\frac{\lambda}{1-\lambda} + r)(1 - \lambda^{T-1})},$$

*for $1 \leq i \leq T$.*

*Proof.* Assume $\dot{\pi}$ is a fixed point with $\sum_{i\geq 1} \dot{\pi}_i \leq T$, meaning $G_i(\dot{\pi}) = 0$ for $i \geq 1$, where $G(x) = (G_1(x), G_2(x), \ldots, G_T(x))$. When $\sum_{i\geq 1} \dot{\pi}_i \leq T$, we can simplify $\sum_{i\geq 1} G_i(\dot{\pi}) = 0$ to $\lambda - \dot{\pi}_1 = 0$. Hence, $\dot{\pi}_1$ must equal $\lambda$. The expressions for $\dot{\pi}_i$ then follow from the condition $G_i(\dot{\pi}) = 0$. □

In Appendix B we prove the following theorem, using the same reasoning as the proof of Theorem 2.4:

**Theorem 3.8.** *All the trajectories of the set of ODEs given by* (3.7)-(3.9)*, starting from $x \in E_T$ converge towards the unique fixed point $\dot{\pi}$.*

Due to the above theorem, we can now express the main performance measures of the max-push strategy via Theorem 3.7:

**Corollary 3.3.** *The mean delay $D_{mp}$ of a job under the max-push strategy equals*

$$D_{mp} = \frac{1 - \lambda^T + (\frac{\lambda}{1-\lambda} + r)(1 - T\lambda^{T-1} + (T-1)\lambda^T)}{1 + r(1-\lambda)(1-\lambda^{T-1}) - \lambda^T}.$$

*A predefined overall probe rate $R$ can be matched by setting*

$$r_{mp} = \frac{R}{\lambda^{T-1}(R + \lambda) - R} - \frac{\lambda}{1 - \lambda}, \tag{3.10}$$

*where $0 \leq r_{mp} < \infty$ for $\lambda^{T+1}/(1 - \lambda^T) \leq R < \lambda^T/(1 - \lambda^{T-1})$.*

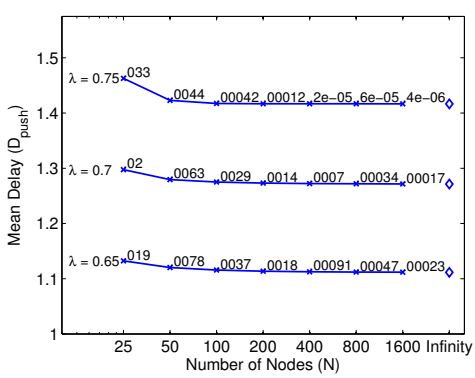


Figure 3.7: Simulated mean delay and relative error for a finite system of varying size, using a max-push strategy with $T = 2$ and $R = 1$.

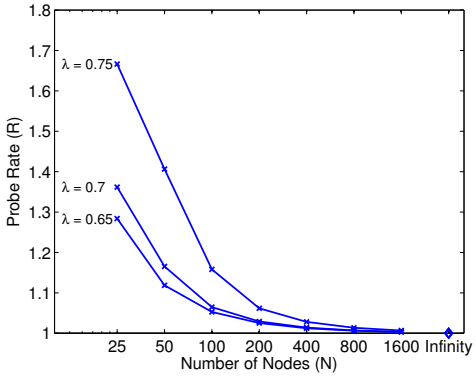


Figure 3.8: Observed probe rate for the finite system using a max-push strategy with $T = 2$.

*Proof.* The mean response time $D$ can be expressed as $\sum_{i\geq 1}^{T} \pi_i/\lambda$ by Little's law. For the max-push strategy the overall probe rate $R$ equals

$$R = \dot{\pi}_T \left( \frac{\lambda}{1-\lambda} + r_{mp} \right),$$

as the instantaneous transfer of an arrival to a queue with $T$ jobs requires $1/(1-\lambda)$ probe messages on average. □

## 3.6 Numerical Results for the Max-Push Strategy

### 3.6.1 Validation

In this section we validate the infinite system model for the max-push strategy using the same approach as in Section 3.4.1 for the rate-based push strategy with $T > 1$. The rate $r_{mp}$ in the simulation was determined using the relationship in (3.10).

The mean delay as a function of the number of nodes $N$ and the relative error are shown in Figure 3.7 for $T = 2$, $R = 1$ and $\lambda = 0.65, 0.7$ and $0.75$. Notice, the max-push strategy with $T = 2$ and $R = 1$ is only properly defined for $\lambda \in [0.6180, 0.7549]$ due to (3.6), larger $\lambda$ values would result in the choice of a larger $T$ value. The relative errors are small (below 1% for $N = 100$ nodes) and decrease as $N$ increases. We should note that the probe rates observed during the simulation are well above 1 for small $N$ as illustrated in Figure 3.8. Hence, the relationship in (3.10) for the max-push strategy is less accurate than (3.5) for the rate-based push strategy for small $N$. Nonetheless, the observed probe rate still seems to decrease to 1 as $N$ tends to infinity.

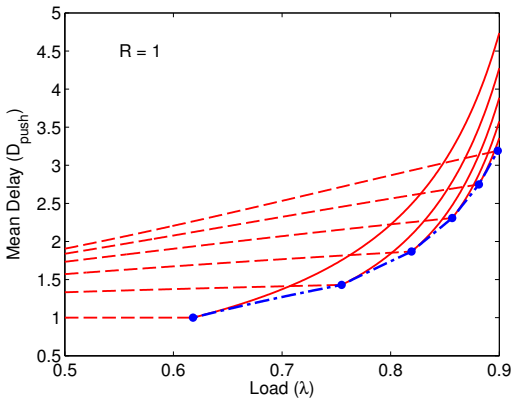


Figure 3.9: The mean delay for the rate-based push strategy (full lines) for $T = 1, \ldots, 6$ with $R = 1$. For each strategy, the probe rate $r = \infty$ when $\lambda$ is below the load marked by a dot. The max-push strategy interconnects the dots, as shown by dash-dotted curves.

### 3.6.2 Comparison of Pull and Max-Push Strategy

The mean delay of the max-push strategy is depicted in Figure 3.9. The dots represent the points where $r = \infty$ for the rate-based push strategy, i.e., the positive real roots of $\lambda^{T+1} + (\lambda^T - 1)R = 0$. The max-push strategy nicely interconnects these points as it utilizes the remaining probe rate.

The combination of $(\lambda, R)$ values for which the pull strategy is outperformed by the rate-based push strategy with $T = 1$, by the rate-based push strategy with $T \geq 1$ and by the max-push strategy, respectively, is shown in Figure 3.10. The pull strategy is still the most effective for larger loads $\lambda$, however, for a large range of $(\lambda, R)$ values the delay of the pull strategy can be reduced using a rate-based push strategy with $T > 1$ or a max-push strategy.

## 3.7 Finite Versus Infinite System Model

As in Chapter 2 for the rate-based strategies with $T = 1$, we can define a family of density dependent Markov chains [30] to describe the behavior of the stochastic finite systems with $N$ nodes for both the rate-based pull/push and max-push strategy. In case of the max-push strategy convergence towards the infinite system model over finite time scales follows from Kurtz's well-known theorem [30] and the convergence extends to the stationary regime as we showed that the set of ODEs given by (3.7)-(3.9) has a unique global attractor in $E_T$, due to a result by Benaïm [37].

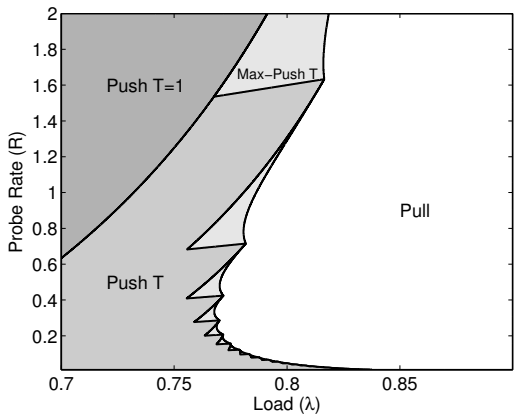


Figure 3.10: The different areas identify the $(\lambda, R)$ combinations for which the rate-based pull strategy (with $T = 1$) is outperformed by the rate-based push strategy with $T = 1$, by the rate-based push strategy with $T \geq 1$ and by the max-push strategy.

For the rate-based pull/push strategy with $T \geq 1$ we can rely on the following generalization of Kurtz's theorem [23, Theorem 3.13]:

**Theorem 3.9** (Kurtz)**.** *Consider a family of density dependent CTMCs, with $F$ Lipschitz. Let $\lim_{N\to\infty} X^{(N)}(0) = \tilde{x}$ a.s. and let $\phi_t(\tilde{x})$ be the unique solution to the initial value problem $\frac{d}{dt}x(t) = F(x(t))$ with $x(0) = \tilde{x}$. Consider the path $\{\phi_t(\tilde{x}), t \leq T\}$ for some fixed $T \geq 0$ and assume that there exists a neighborhood $K$ around this path satisfying*

$$\sum_{\ell \in L} |\ell| \sup_{x \in K} \beta_\ell(x) < \infty, \tag{3.11}$$

*then*

$$\lim_{N\to\infty} \sup_{t \leq T} \left| X^{(N)}(t) - \phi_t(\tilde{x}) \right| = 0 \quad a.s.$$

For the rate-based pull/push strategy, the above condition (3.11) corresponds to showing that there exists an environment $K$ such that $\sum_{i\geq 2} \sup_{x\in K}(x_i - x_{i+1}) < \infty$. Such an environment can be shown to exist by repeating the argument for $T = 1$ from Theorem 2.7. To show that the convergence extends to the stationary regime, we can make use of a theorem by Benaïm and Le Boudec [32] as in the $T = 1$ case, where the required proof for the tightness of the measures can be proven as in Chapter 2.

## 3.8 Conclusion

In this chapter we introduced a new class of rate-based pull and push strategies that can match any predefined maximum allowed probe rate $R$. This class relied on a threshold parameter $T$ such that jobs can only be exchanged between idle nodes and nodes with a queue length exceeding $T$, where the class of strategies introduced in Chapter 2 corresponds to setting $T = 1$. We derived a closed form expression for the mean delay of this new class of strategies in a homogeneous network with Poisson arrivals and exponential job durations under the so-called infinite system model.

We showed that setting $T = 1$ is optimal for the pull strategies considered, while for the push strategy setting $T > 1$ may reduce the mean delay for some values of $(\lambda, R)$, i.e., for larger $\lambda$ and smaller $R$ values. We further introduced the max-push strategy, which utilizes the remaining probe rate capacity in case $R > \lambda^{T+1}/(1-\lambda^T)$, derived a close form expression for its mean delay and (numerically) identified the $(\lambda, R)$ region where it outperforms the pull strategy.

We proved that the infinite system models of both the rate-based strategies with $T > 1$ and the max-push strategy, are the proper limit processes of the finite stochastic systems with $N$ nodes as $N$ tends to infinity over any finite timescale. Moreover, the convergence was shown to extend to the stationary regime by proving that the ODEs have a global attractor. We validated these theoretical results by simulation, and have shown that the infinite model is an accurate approximation for finite systems of moderate size.

Chapter 4

# Analysis of Rate-Based Pull and Push Strategies with Limited Migration Rates in Large Distributed Networks

*In this chapter*[1] *we analyze the performance of pull and push strategies in large homogeneous distributed systems where the number of job transfers per time unit is limited. Job transfer strategies which rely on lightly-loaded servers to attract jobs from heavily-loaded servers are known as pull strategies, whereas for push strategies the heavily loaded servers initiate the job transfers to lightly loaded servers. To this end, servers transmit probe messages to discover other servers that are able to take part in a job transfer.*

*Previous work on rate-based pull and push strategies focused on the impact of the probe rate on the mean job response time. In this chapter we also limit the overall migration rate and show that any predefined migration rate can be matched by both the rate-based pull and push strategies. We present closed form formulas for the mean response time (as a function of the allowed probe and migration rate) and validate their accuracy by simulation.*

*We also introduce and analyze a new pull strategy and show that under high loads it is superior to the push strategies considered, while the push strategies offer only a very limited gain for medium to low load scenarios.*

[1] This chapter is based on work published in VALUETOOLS 2015 as "Analysis of Rate-Based Pull and Push Strategies with Limited Migration Rates in Large Distributed Networks" by Wouter Minnebo and Benny Van Houdt [38].

## 4.1 Introduction

The previous chapters assumed zero cost for job transfers, which is not always realistic. When jobs are difficult to migrate, it would be desirable to be able to limit migrations to a predefined overall migration rate $M$, while not exceeding the predefined overall probe rate $R$.

This chapter focuses on the following points:

1. We indicate how to set the parameter $r$ (and $T$) of the push, pull and max-push strategy to match any predefined migration rate $M$.

2. We argue that setting $T = 1$ for the pull strategy is no longer optimal when an overall migration limit $M$ is considered, as was the case in Chapter 3 and introduce a new pull strategy, called the conditional-pull strategy.

3. We show that the conditional-pull strategy is equivalent in stationary queue length distribution to the max-push variant when the overall probe rate $R$ tends to infinity, i.e., when only an overall migration limit $M$ is considered.

4. We consider a system where both an overall probe limit ($R$) and an overall migration limit ($M$) are imposed. For this system we compare the performance of push and pull strategies. We find that even for moderate $R$ the conditional-pull performs almost as well as the max-push for low to moderate loads, and performs significantly better for higher loads.

The chapter is structured as follows. Section 4.2 summarizes the rate-based strategies considered in this chapter. In Section 4.3 we briefly summarize earlier work concerning rate-based pull and push strategies, and introduce an overall migration limit $M$ for these strategies. Also, we derive an expression for the corresponding maximum probe rate $r_{both|M}$ for the rate-based push and pull, and rewrite the mean delay in an equivalent form. In Section 4.4 we adapt the max-push strategy to match $M$ by finding the corresponding probe rate $r_{mp|M}$, after summarizing earlier work. Section 4.5 considers pull strategies with $T > 1$, and introduces the new conditional-pull strategy. It is shown that the conditional-pull is equivalent to the max-push strategy in case there is no probe limit $R$ and only a migration limit $M$. In addition, the infinite system model describing the evolution of the conditional pull strategy is numerically validated, and argued to be the proper limiting process as the system size tends to infinity. Finally, we compare the mean delay of max-push and conditional-pull in Section 4.6.

## 4.2 Rate Based Strategies

We consider a continuous-time system of $N$ queues, where each queue has a single server and infinite buffer. Each queue operates under Poisson job arrivals with rate $\lambda < 1$, and exponential service time with mean 1. Jobs are processed in a first-come-first-served order.

Traditional strategies send a maximum of $L_p$ probes the instant a server's last job completes or the instant a job arrives when the server is already busy. In contrast, under rate-based strategies probes are no longer sent at job arrival or

completion times but at a fixed rate $r$ as long as the server is idle (pull) or has at least $T$ jobs waiting (push). More formally, probe messages are transmitted by a server according to an interrupted Poisson process with rate $r$.

The strategies considered in this chapter can be summarized as follows:

1. *Rate-based Push:* As soon as the queue length exceeds $T$, a server starts to generate probe messages according to a Poisson process with rate $r$. Whenever the queue length drops below $T$, this process is interrupted until the queue length exceeds $T$ again. The node that is probed is selected at random and is only allowed to accept a job if it is idle.

2. *Rate-based Pull:* Whenever a server is idle it generates probe messages according to a Poisson process with rate $r$. This process is interrupted whenever the server is busy. The node that is probed is selected at random and is only allowed to transfer one of its jobs if its queue length exceeds $T$.

3. *Max-Push:* The instant a new job arrives at a queue with length $T$, probes are sent at an infinite rate. When $\lambda < 1$ this corresponds to stating that the job is instantaneously transferred to an empty server. A server with $T$ jobs in its queue, generates probe messages according to a Poisson process with rate $r$. Whenever the queue length drops to $T-1$, this process is interrupted as long as the queue length remains below $T$. The node that is probed is selected at random and is only allowed to accept a job if it is idle.

4. *Conditional-Pull:* Whenever a node is idle, the node will generate probe messages according to a Poisson process with rate $r$. This process is interrupted whenever the server becomes busy. The probed node is selected at random and the probe is always successful if there are at least $T$ jobs waiting to be served, and successful with some probability $p$ (matching $M$, see (4.30)) if there are exactly $T-1$ jobs waiting to be served.

We do not consider hybrid strategies, which combine both push and pull behavior. These were proven to be inferior to a pure push or pull strategy when $T = 1$ in Theorem 2.6.

## 4.3 Pull and Push Strategies

Infinite system models and closed form solutions for both pull and push strategies were introduced in Chapter 2 and 3. Before introducing new constraints and strategies, we briefly summarize the main findings of those chapters.

The evolution of both the rate-based pull and push strategy under the infinite system model is described by a set of ODEs denoted as $\frac{d}{dt}x(t) = F(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$.

From Theorems 3.2 and 3.3, it is known that $\frac{d}{dt}x(t) = F(x(t))$ has a unique fixed point $\bar{\pi} = (\bar{\pi}_1, \bar{\pi}_2, \ldots)$ with $\sum_{i\geq 1} \bar{\pi}_i < \infty$ that is a global attractor, given by

$$\bar{\pi}_i = \frac{\lambda\left((1+r)\lambda^{i-1} - r\lambda^T\right)}{1 + r(1-\lambda^T)} \qquad 1 \leq i \leq T+1, \tag{4.1}$$

$$\bar{\pi}_i = \bar{\pi}_{T+1}\left(\frac{\lambda}{1+(1-\lambda)r}\right)^{i-T-1} \qquad i > T+1. \tag{4.2}$$

This fixed point is used in conjunction with Little's Law in Corollary 3.1 to formulate the mean delay $D_{both}$ of a job under the push or pull strategy:

$$D_{both} = \frac{1}{1-\lambda} - \frac{r\lambda^T\left(\frac{\lambda}{(1-\lambda)(1+r)} + T\right)}{1 + r(1-\lambda^T)}. \tag{4.3}$$

From the relationships $R = (1-\bar{\pi}_1)r_{pull|R}$ and $R = r_{push|R}\bar{\pi}_{T+1}$, we find

$$R = (1-\lambda)r_{pull|R}, \tag{4.4}$$

and

$$R = \frac{\lambda^{T+1}}{(1-\lambda^T) + 1/r_{push|R}}. \tag{4.5}$$

It follows that whenever $R > \lambda^{T+1}/(1-\lambda^T)$, the rate $r_{push|R}$ can be chosen arbitrarily large (i.e., $r_{push|R} = \infty$).

### 4.3.1 Limiting the Overall Migration Rate

When the overall migration rate is limited, the choice of $r$ must satisfy this constraint. We first indicate how to set $r$ to match $M$, and rewrite the formula for the mean delay. Then we show whether $R$ or $M$ is the strictest constraint for a given load $\lambda$.

**Theorem 4.1.** *Both rate-based pull and push strategies match a predefined migration rate $M$ by letting the probe rate $r = r_{both|M}$, with $r_{both|M}$:*

$$r_{both|M} = \frac{M}{(\lambda(1-\lambda) + M)\lambda^T - M}. \tag{4.6}$$

*For this setting, both strategies achieve the same mean delay.*

*Proof.* The relationship (4.6) readily follows from the formulation of the overall migration rate for both rate-based strategies:

$$M_{both} = \frac{r(1-\lambda)\lambda^{T+1}}{r(1-\lambda^T) + 1}. \tag{4.7}$$

From a push perspective this equation describes the fraction of nodes with queue length larger than or equal to $T+1$ ($\bar{\pi}_{T+1} = \lambda^{T+1}/(1 + r(1-\lambda^T))$ from (4.1)) sending probes at rate $r$, succeeding with probability $(1-\lambda)$. For a pull strategy the overall migration rate is expressed as the fraction of empty queues $(1-\lambda)$ sending probes at rate $r$ and succeeding when probing a queue of length $T+1$ or longer ($\bar{\pi}_{T+1} = \lambda^{T+1}/(1 + r(1-\lambda^T))$ from (4.1)). □

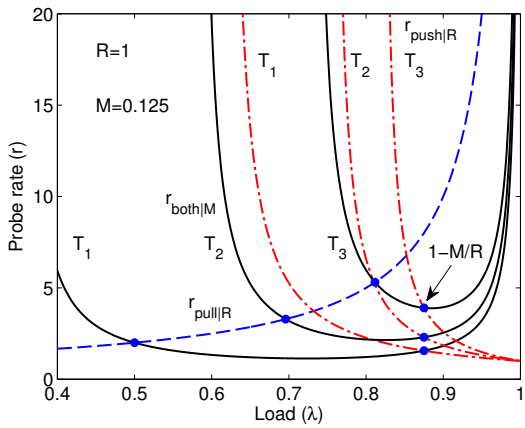


Figure 4.1: The probe rates $r$ imposed by either the probe limit $R = 1$ for push (dot-dashed) and pull (dashed), or migration limit $M = 1/8$ (full), for $T = 1, 2, 3$. Note that $r_{pull|R}$ is independent of $T$.

**Theorem 4.2.** *The mean delay of rate-based pull and push strategies can be expressed as*

$$D_{both} = \frac{1}{1-\lambda}\left(1 - \frac{M_{both}}{\lambda}\left(T + \frac{\lambda}{(1-\lambda)(1+r)}\right)\right). \tag{4.8}$$

*Proof.* Equation (4.8) follows by rewriting (4.7) to

$$\frac{r\lambda^T}{1 + r(1-\lambda^T)} = \frac{M_{both}}{\lambda(1-\lambda)},$$

and substituting this expression in (4.3). □

The previous theorem shows that the improvement in mean delay compared to a standard M/M/1 queue, can be expressed as a migration frequency ($M/\lambda$) times a migration gain ($T + \lambda/((1-\lambda)(1+r))$). The migration frequency denotes how many migrations per job take place on average. The migration gain quantifies the number of places in the queue the migrating job skips. All migrating jobs skip at least $T$ places by construction of the strategy, and skip more places depending on the queue length of the job sender. The average number of places skipped above $T$ equals the average number of customers in an M/M/1 queue with service rate $1 + r(1-\lambda)$, which equals $\lambda/((1-\lambda)(1+r))$.

Both the overall migration limit $M$ and the overall probe limit $R$ impose a maximum on $r$. In case $R \leq M$, all probes are allowed to generate a migration, so $r$ will only be constrained by $R$. In any practical setting there will be more probes allowed than migrations, and the probe rate will be constrained by $R$ or $M$ depending on $\lambda$. An overview of the probe rates matching $R$ or $M$, for both push and pull strategies with $T = 1, 2, 3$, is given in Figure 4.1.

To determine the range of $\lambda$ in which each constraint is most strict, we determine the intersection of $r_{both|M}$ with $r_{push|R}$ and $r_{pull|R}$.

**Lemma 4.1.** *The probe rates $r_{push|R}$ and $r_{both|M}$ intersect at $\lambda = 0$ and $1 - M/R$ only, and both rates are positive for $\lambda = 1 - M/R$ if and only if $\left(1 - \frac{M}{R}\right)^T > \frac{R^2}{R-M+R^2}$.*

*Proof.* We look at the roots of

$$\begin{aligned} f_3(\lambda) &= r_{push|R} - r_{both|M} \\ &= \frac{R}{\lambda^{T+1} - R(1-\lambda^T)} - \frac{M}{(\lambda(1-\lambda)+M)\lambda^T - M} \\ &= \frac{\lambda^{T+1}(M - R(1-\lambda))}{(\lambda^{T+1} - R(1-\lambda^T))((\lambda(1-\lambda)+M)\lambda^T - M)}, \end{aligned} \tag{4.9}$$

and observe that $\lambda = 0$ is a root with multiplicity $T+1$ and $\lambda = 1 - M/R$ is a root with multiplicity 1.

By substituting $\lambda = 1 - M/R$ in the expressions for $r_{push|R}$ and $r_{both|M}$, we find that both are positive if and only if

$$\frac{1}{\frac{(R-M+R^2)\left(1-\frac{M}{R}\right)^T}{R^2} - 1} > 0.,$$

which can be simplified to

$$\left(1 - \frac{M}{R}\right)^T > \frac{R^2}{R - M + R^2}.$$

□

**Theorem 4.3.** *For the rate-based push strategy $r$ should be set as follows in order to respect both the probe rate $R$ and migration rate $M$:*

1. *$R \geq \lambda^{T+1}/(1-\lambda^T)$ and $M \geq \lambda^{T+1}(1-\lambda)/(1-\lambda^T)$: $r$ can be arbitrarily large.*
2. *$R \geq \lambda^{T+1}/(1-\lambda^T)$ and $M < \lambda^{T+1}(1-\lambda)/(1-\lambda^T)$: $r$ can be at most $r_{both|M}$.*
3. *$R < \lambda^{T+1}/(1-\lambda^T)$ and $M \geq \lambda^{T+1}(1-\lambda)/(1-\lambda^T)$: $r$ can be at most $r_{push|R}$.*
4. *$R < \lambda^{T+1}/(1-\lambda^T)$ and $M < \lambda^{T+1}(1-\lambda)/(1-\lambda^T)$: Let $\tau = \left(1 - \frac{M}{R}\right)^T$ and $\upsilon = \frac{R^2}{R-M+R^2}$.*
   - *If $\tau > \upsilon$, $r$ can be at most $r_{both|M}$ if $\lambda < 1 - M/R$ and at most $r_{push|R}$ otherwise.*
   - *If $\tau \leq \upsilon$, $r$ can be at most $r_{push|R}$.*

*Proof.* By taking the limit for $r \to \infty$ of Equation (4.5), we obtain the maximal attainable probe rate $\lambda^{T+1}/(1-\lambda^T)$. If $R$ is larger than this value, $r$ can be chosen arbitrarily large without exceeding $R$. By taking the limit for $r \to \infty$ of Equation (4.7), we obtain the maximal attainable migration rate $\lambda^{T+1}(1-\lambda)/(1-\lambda^T)$. If $M$ is larger than this value, $r$ can be chosen arbitrarily large without exceeding $M$. The first three points follow immediately.

From Lemma 4.1 we know that if $\tau < \upsilon$, $r_{both|M}$ and $r_{push|R}$ are negative at $1 - M/R$. As that was the only non-zero root of $f_3(\lambda)$ and $\lim_{\lambda\to 1} r_{both|M} = \infty$ is larger than $\lim_{\lambda\to 1} r_{push|R} = R$, it follows that $r_{both|M} > r_{push|R}$ where both are positive. In this case $R$ is the strictest constraint, so $r$ can be at most $r_{push|R}$.

If $\tau = \upsilon$, both $r_{both|M}$ and $r_{push|R}$ are infinite at $1 - M/R$. As $1 - M/R$ was the only positive root of $f_3(\lambda)$ and $\lim_{\lambda\to 1} r_{both|M} > \lim_{\lambda\to 1} r_{push|R}$, it follows that $r_{both|M} > r_{push|R}$ in the interval $(1 - M/R, 1)$. In this case $R$ is the strictest constraint so $r$ can be at most $r_{push|R}$.

If $\tau > \upsilon$, $r_{both|M}$ and $r_{push|R}$ are positive at the intersection $\lambda = 1 - M/R$. As $1 - M/R$ was the only positive root of $f_3(\lambda)$ and $\lim_{\lambda\to 1} r_{both|M} > \lim_{\lambda\to 1} r_{push|R}$, it follows that $r_{both|M} > r_{push|R}$ in the interval $(1 - M/R, 1)$ and $r_{both|M} < r_{push|R}$ otherwise. In this case $r$ can be at most $min(r_{both|M}, r_{push|R})$. □

**Lemma 4.2.** *If $M < \frac{R}{1+TR}$, $r_{both|M} - r_{pull|R}$ has a unique root $\lambda_T$ in $(0, 1)$, otherwise it has no roots in $(0, 1)$.*

*Proof.* We look for the root of

$$f_4(\lambda) = r_{pull|R} - r_{both|M} \tag{4.10}$$

$$= \frac{R}{1-\lambda} - \frac{M}{(\lambda(1-\lambda) + M)\lambda^T - M}, \tag{4.11}$$

which can be rewritten as

$$f_4(\lambda) = \frac{\alpha(\lambda)}{\beta(\lambda)} = \frac{(R\lambda^{T+1} - M)(1-\lambda) - MR(1-\lambda^T)}{(1-\lambda)((\lambda(1-\lambda) + M)\lambda^T - M)}.$$

Clearly, $\alpha(1) = 0$ and $\alpha(0) = -M(1 + R) < 0$. The derivative of $\alpha(\lambda)$ is given by

$$\frac{d}{d\lambda}(\alpha(\lambda)) = R(T+1)(1-\lambda)\lambda^T - R\lambda^{T+1} + M + MRT\lambda^{T-1}, \tag{4.12}$$

where we see that $d\alpha(0)/d\lambda = M$ for $T > 1$, and $d\alpha(0)/d\lambda = M(1 + R)$ for $T = 1$. For $d\alpha(1)/d\lambda = M - R + MRT$ we distinguish three cases:

1. If $M < R/(1 + TR)$, $d\alpha(1)/d\lambda < 0$.
2. If $M = R/(1 + TR)$, $d\alpha(1)/d\lambda = 0$.
3. If $M > R/(1 + TR)$, $d\alpha(1)/d\lambda > 0$.

Further, $d\alpha(\lambda)/d\lambda \geq 0$ in $(0, 1)$ for cases 2 and 3. This can be seen by separating positive and negative terms in (4.12), and requiring that

$$R(2 + T)\lambda^{T+1} \leq M + MRT\lambda^{T-1} + R(1 + T)\lambda^T.$$

Dividing by $R\lambda^{T+1}$ yields

$$2 + T \leq \frac{M}{R\lambda^{T+1}} + \frac{MT}{\lambda^2} + \frac{1+T}{\lambda}.$$

Since $\lambda$ is at most one, $\frac{1+T}{\lambda}$ is at least $T + 1$, it suffices that

$$1 \leq \frac{M}{R\lambda^{T+1}} + \frac{MT}{\lambda^2}.$$

The right hand side will be minimal for $\lambda = 1$, so $d\alpha(\lambda)/d\lambda \geq 0$ if

$$1 \leq M/R + MT.$$

This requirement is equivalent to $M \geq R/(1+TR)$.

In case 2 and 3, it follows from $\alpha(0) < 0$ and $\alpha(1) = 0$ and $d\alpha(\lambda)/d\lambda \geq 0$ in $(0,1)$, that $\alpha(\lambda)$ has no roots in $(0,1)$.

In case 1, $\alpha(\lambda)$ has at least one root in $(0,1)$ since $\alpha(0) < 0$ and $\alpha(1) = 0$ and $d\alpha(1)/d\lambda < 0$. By Decartes' rule of signs one finds that $\alpha(\lambda)$ has either 0 or 2 positive roots. Since $\alpha(1) = 0$, we may conclude that we have exactly one root in $(0,1)$ in case 1. □

For $T = 1$, the unique root $\lambda_T$ of Lemma 4.2 reduces to $\lambda_1 = \sqrt{M + M/R}$. However, there seems to be no closed form expression for general $T$.

**Theorem 4.4.** *For the rate-based pull strategy $r$ should be set as follows in order to respect both the probe rate $R$ and migration rate $M$:*

1. *$M \geq \lambda^{T+1}(1-\lambda)/(1-\lambda^T)$: $r$ can be at most $r_{pull|R}$.*
2. *$M < \lambda^{T+1}(1-\lambda)/(1-\lambda^T)$:*
   - *If the migration limit is sufficiently high ($M \geq \frac{R}{1+TR}$), then $r$ can be at most $r_{pull|R}$.*
   - *If the migration limit is sufficiently low ($M < \frac{R}{1+TR}$), $r$ can be at most $r_{pull|R}$ if $\lambda < \lambda_T$, and at most $r_{both|M}$ otherwise.*

*Proof.* By taking the limit for $r \to \infty$ of Equation (4.7), we obtain the maximal attainable migration rate $\lambda^{T+1}(1-\lambda)/(1-\lambda^T)$. If $M$ is larger than this value, $r$ can be chosen arbitrarily large without exceeding $M$. Case 1 follows immediately.

From Lemma 4.2 we know that if $M > \frac{R}{1+TR}$, $r_{both|M}$ and $r_{pull|R}$ do not intersect in the interval $(0,1)$. Since $r_{both|M}$ has an asymptote going to $+\infty$ for a $\lambda$ in $(0,1)$, it follows that $r_{both|M} > r_{push|R}$ in the interval $(0,1)$. A similar argument holds for $M = \frac{R}{1+TR}$.

If $M < \frac{R}{1+TR}$, $r_{both|M}$ and $r_{pull|R}$ intersect at a unique point $\lambda_T \in (0,1)$. As both $r_{both|M}$ and $r_{pull|R}$ are infinite at $\lambda = 1$, $r_{both|M}$ is U-shaped and $r_{pull|R}$ is strictly increasing, it follows that for $\lambda < \lambda_T$, $r_{both|M} > r_{pull|R}$ and for $\lambda > \lambda_T$, $r_{both|M} < r_{push|R}$. □

## 4.4 Max-Push

The rate-based push is unable to reach an overall request rate higher than $\lambda^{T+1}/(1-\lambda^T)$ for any $T$. When the overall probe limit $R$ exceeds this value, it is possible to use the remaining request rate by using the max-push variant as introduced in Chapter 3. This strategy lets nodes with a queue length of $T$ send probes at a finite rate $r_{mp|R}$, and migrates all new arrivals to queues with length $T$ by sending probes at an infinite rate until an empty server is found. As $\lambda^{T+1}/(1-\lambda^T)$ is an increasing function in $\lambda$ and decreasing in $T$, the unique solution for $\lambda$ to $\lambda^{T+1}/(1-\lambda^T) = R$ is increasing in $T$. Therefore, there is a unique $T > 1$ satisfying

$$\lambda^{T+1}/(1-\lambda^T) \leq R < \lambda^T/(1-\lambda^{T-1}). \tag{4.13}$$

For this $T$, the evolution of the max-push strategy can be described by a set of ODEs $\frac{d}{dt}x(t) = G(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$.

Theorems 3.7 and 3.8 show that the set of ODEs has a unique fixed point $\dot{\pi} = (\dot{\pi}_1, \ldots, \dot{\pi}_T)$ that is a global attractor, and can be expressed as

$$\dot{\pi}_i = \lambda^i \frac{1 + (\frac{\lambda}{1-\lambda} + r)(1 - \lambda^{T-i})}{1 + (\frac{\lambda}{1-\lambda} + r)(1 - \lambda^{T-1})}, \tag{4.14}$$

for $1 \leq i \leq T$. The mean delay $D_{mp}$ of a job under the max-push strategy is given by Corollary 3.3:

$$D_{mp} = \frac{1 - \lambda^T + (\frac{\lambda}{1-\lambda} + r)(1 - T\lambda^{T-1} + (T-1)\lambda^T)}{1 + r(1-\lambda)(1-\lambda^{T-1}) - \lambda^T}. \tag{4.15}$$

For the max-push strategy the overall probe rate $R$ equals

$$R = \dot{\pi}_T \left( \frac{\lambda}{1-\lambda} + r \right), \tag{4.16}$$

as the instantaneous transfer of an arrival to a queue with $T$ jobs requires $1/(1-\lambda)$ probe messages on average. Therefore, a predefined overall probe rate $R$ can be matched by setting

$$r_{mp|R} = \frac{R}{\lambda^{T-1}(R+\lambda) - R} - \frac{\lambda}{1-\lambda}, \tag{4.17}$$

where $0 \leq r_{mp|R} < \infty$ for $\lambda^{T+1}/(1-\lambda^T) \leq R < \lambda^T/(1-\lambda^{T-1})$.

### 4.4.1 Limiting the Overall Migration Rate

As the rate-based push strategy cannot exceed the probe rate $\lambda^{T+1}/(1-\lambda^T)$, it is unable to exceed an overall migration rate of $(1-\lambda)\lambda^{T+1}/(1-\lambda^T)$. It follows that when $M > (1-\lambda)\lambda^{T+1}/(1-\lambda^T)$, queues with a length of at least $T+1$ can probe at an arbitrarily high rate without exceeding the migration limit $M$, effectively reducing all queues to a length of at most $T$. Queues with length $T$ can then send probes with a finite $r$ to match $M$. In other words, in order to match $M$ (instead of $R$ as in the previous section), set $T$ such that

$$\frac{(1-\lambda)\lambda^{T+1}}{1-\lambda^T} \leq M < \frac{(1-\lambda)\lambda^T}{1-\lambda^{T-1}}. \tag{4.18}$$

and determine the probe rate $r_{mp|M}$ when the queue length equals $T$ by the following theorem:

**Theorem 4.5.** *The max-push strategy matches a predefined migration rate $M$ by letting the probe rate $r = r_{mp|M}$ with*

$$r_{mp|M} = \lambda \left( \frac{M}{((1-\lambda)\lambda + M)\,\lambda^T - \lambda M} - \frac{1}{1-\lambda} \right). \tag{4.19}$$

*When matching $M$ this way, $\dot{\pi}_i$ for $i \leq T$ reduces to*

$$\dot{\pi}_i = \lambda^i - \frac{M(1-\lambda^{i-1})}{1-\lambda}, \tag{4.20}$$

*Proof.* The relationship (4.19) follows from the formulation of the overall migration rate. The migrations of new arrivals in queues with length $T$ are given by $\dot{\pi}_T\lambda$. The migrations resulting from a successful probe sent by queues with length $T$ are given by $\dot{\pi}_T r(1-\lambda)$. Probes are successful if they locate an empty server, which they do with probability $1-\lambda$. Therefore, the overall migration rate can be expressed as

$$\begin{aligned} M_{mp} &= \dot{\pi}_T \left( \frac{\lambda}{1-\lambda} + r \right) (1-\lambda) \\ &= \frac{(1-\lambda)((1-\lambda)r + 2\lambda)\lambda^{T+1}}{\lambda(r(1-\lambda)+1) - ((1-\lambda)r+\lambda)\lambda^T}. \end{aligned} \tag{4.21}$$

The reduction of $\dot{\pi}_i$ to (4.20) follows from substitution of (4.19) in (4.14), and shows the improvement over an M/M/1 queue directly. □

**Theorem 4.6.** *For the max-push strategy $r$ and $T$ should be set as follows in order to respect both the probe rate $R$ and migration rate $M$:*

- *If $\lambda < 1 - M/R$, $T$ must be chosen according to* (4.18) *and $r$ can be at most $r_{mp|M}$.*
- *If $\lambda > 1 - M/R$, $T$ must be chosen according to* (4.13) *and $r$ can be at most $r_{mp|R}$.*
- *If $\lambda = 1 - M/R$, both constraints are equivalent.*

*Proof.* When matching $R$, the max-push strategy realizes an overall migration rate of $R(1-\lambda)$. This follows from substitution of (4.17) in (4.21). When matching $M$, the max-push strategy realizes an overall probe rate of $M/(1-\lambda)$. This follows from substitution of (4.19) in (4.16). Therefore, both constraints are equivalent provided that $\lambda = 1 - M/R$. If $\lambda < 1 - M/R$, then $R(1-\lambda) > M$, so $M$ is the strictest constraint. If $\lambda > 1 - M/R$, then $M/(1-\lambda) > R$, so $R$ is the strictest constraint. □

**Theorem 4.7.** *The mean delay of the max-push strategy can be expressed as*

$$D_{mp} = \frac{1}{1-\lambda}\left(1 - \frac{M_{mp}}{\lambda}\left(\frac{\alpha+\beta}{M_{mp}}\right)\right), \tag{4.22}$$

*with $\alpha = \dot{\pi}_T\lambda T$ and $\beta = \dot{\pi}_T r(1-\lambda)(T-1)$. When $r = r_{mp|M}$, the mean delay $D_{mp}$ reduces to*

$$D_{mp|M} = \frac{1}{1-\lambda} + \frac{M\left(1-\lambda^T\right)}{(1-\lambda)^2\lambda} - \frac{MT+\lambda^{T+1}}{(1-\lambda)\lambda}. \tag{4.23}$$

*Proof.* The improvement in mean delay compared to a standard M/M/1 queue, can be expressed as a migration frequency ($M_{mp}/\lambda$) times a migration gain. The migration frequency denotes how many migrations per job take place on average. The migration gain quantifies the number of places in the queue the migrating job skips. The fraction of migrating jobs *arriving* at a queue with length $T$ ($\dot{\pi}_T\lambda/M_{mp}$) skip $T$ places in the queue. The fraction of migrating jobs from queues with length $T$, being $\pi_T r(1-\lambda)/M_{mp}$, skip $T-1$ places in the queue. Hence, the migration gain is $(\alpha+\beta)/M_{mp}$.

The reduction to $D_{mp|M}$ is found by applying Little's Law to the expression for $\dot{\pi}$ in (4.20), and shows the improvement over an M/M/1 queue explicitly. □

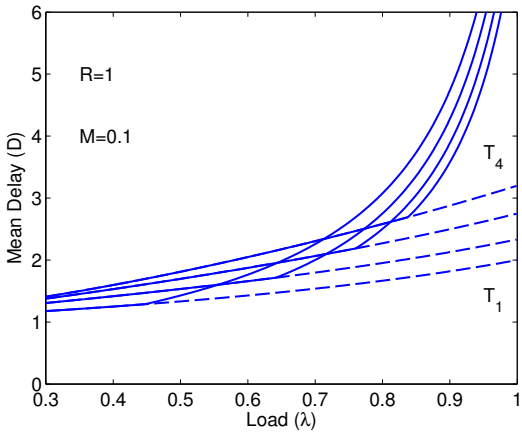


Figure 4.2: The mean delay of the pull strategy for $T = 1, ..., 4$. The probe rate $r$ is constrained by both $R$ and $M$ (full lines). The delay shown in dashed lines is achieved when there is no migration limit, and only the probe limit is in effect. Choosing $T = 1$ is no longer optimal when a maximum migration rate is imposed.

## 4.5 Conditional Pull

When only considering a maximum allowed probe rate $R$, the optimal choice for a pull strategy is to let $T = 1$ as shown in Theorem 3.5. This is no longer the case when taking a maximum allowed migration rate $M$ into account, as shown in Figure 4.2. Intuitively, when the migration limit is small, it is best to pull jobs from longer queues only, resulting in a lower mean delay.

To reduce the mean delay of the rate-based pull strategy, we introduce the conditional pull strategy that can match both $R$ and $M$. Empty servers send probes according to an interrupted Poisson process with rate $r$. Under the conditional pull strategy, empty nodes always accept jobs from queues with length of at least $T + 1$ and also accept jobs from a queue with length $T$ with some probability $p$. This strategy relies on the choice of $p$ to match the migration rate $M$, and lets the probe rate $r$ be determined by $R$, i.e. $r = r_{pull|R} = R/(1 - \lambda)$. Thus, *both* $R$ and $M$ are matched, this is in contrast with the previous strategies, where $r$ was always chosen as large as possible without exceeding $R$ and $M$.

First we note that one can easily see that for $\lambda_T$, being the unique root defined in Lemma 4.2, $\lambda_T < \lambda_{T+1}$ (as $M/((\lambda(1-\lambda)+M)\lambda^T - M)$ increases in $T$ and $\frac{R}{1-\lambda}$ increases in $\lambda$ independent of $T$). Given $\lambda$, the conditional pull strategy sets $T$ such that

$$\lambda_{T-1} \leq \lambda < \lambda_T, \tag{4.24}$$

with $\lambda_0 = 0$. To analyze the response time of a job under the conditional pull strategy we introduce a set of ODEs $\frac{d}{dt}x_i(t) = H(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$. As explained below, the set of ODEs $H(x(t))$ describing the time evolution of

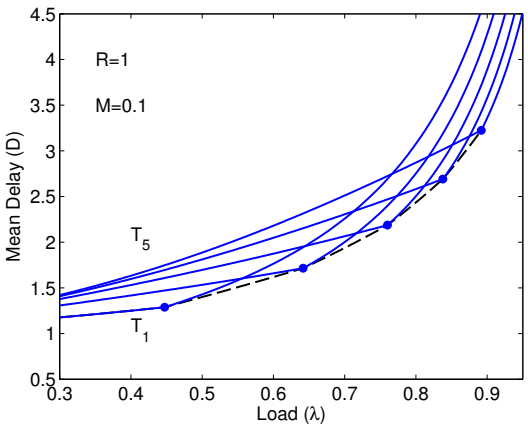


Figure 4.3: Showing the mean delay of the pull strategy with $T > 1$, respecting a migration limit $\dot{M}$. The conditional pull variant is shown in dashed lines.

the queue lengths under the conditional pull strategy is defined as

$$\frac{dx_1(t)}{dt} = -(x_1(t) - x_2(t)) + (\lambda + rx_{T+1}(t) + rp(x_T(t) - x_{T+1}(t)))(1 - x_1(t)) \tag{4.25}$$

$$\frac{dx_i(t)}{dt} = \lambda(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)), \tag{4.26}$$

for $1 < i < T$, and

$$\frac{dx_i(t)}{dt} = \lambda(x_{i-1}(t) - x_i(t)) - (1 + rp^{1[i=T]}(1 - x_1(t)))(x_i(t) - x_{i+1}(t)), \tag{4.27}$$

for $i \geq T$, where $1[A] = 1$ if $A$ is true and $1[A] = 0$ otherwise. The terms $\lambda(x_{i-1}(t) - x_i(t))$ and $x_i(t) - x_{i+1}(t)$, for $i \geq 1$, correspond to arrival and service completions, respectively. Queues of length 1 are created by job transfers at rate $(rx_{T+1}(t) + rp(x_T(t) - x_{T+1}(t)))(1 - x_1(t))$ as the fraction of empty nodes $(1 - x_1(t))$ probe at rate $r$, and a probe is successful with probability $x_{T+1}(t) + p(x_T(t) - x_{T+1}(t))$. Similarly, migrating jobs reduce the number of queues with exactly $i$ jobs, for $i > T$, at rate $r(1 - x_1(t))(x_i(t) - x_{i+1}(t))$ and at rate $rp(1 - x_1(t))(x_T(t) - x_{T+1}(t))$ for $i = T$.

The next theorem shows that this set of ODEs has a unique fixed point with $\sum_{i\geq 1} \hat{\pi}_i < \infty$. In Appendix C we briefly argue why this fixed point can be used to approximate the queue length distribution of a node as the number of nodes becomes large. The argument is similar to the one used in Chapter 2. We also validate the accuracy of this approximation by simulation in Section 4.5.1.

**Theorem 4.8.** *The set of ODEs* $\frac{d}{dt}x(t) = H(x(t))$ *has a unique fixed point* $\hat{\pi} = (\hat{\pi}_1, \hat{\pi}_2, \ldots)$ *with* $\sum_{i\geq 1} \hat{\pi}_i < \infty$. *The fixed point can be expressed as:*

$$\hat{\pi}_i = \frac{\lambda^i \left((1-\lambda)r\left(\sum_{j=0}^{T-i} \lambda^j + p(r+1)\left(1 - \lambda^{T-i}\right)\right) + 1\right)}{(1-\lambda)r\left(\sum_{j=0}^{T-1} \lambda^j + p(r+1)\left(1 - \lambda^{T-1}\right)\right) + 1} \tag{4.28}$$

*for* $1 \leq i \leq T$*, and for* $i > T$ *as*

$$\hat{\pi}_i = \pi_T \left( \frac{\lambda}{1 + r(1-\lambda)} \right)^{i-T} \tag{4.29}$$

*Proof.* Assume $\hat{\pi}$ is a fixed point with $\sum_{i\geq 1} \hat{\pi}_i < \infty$, meaning $H_i(\hat{\pi}) = 0$ for $i \geq 1$, where $H(x) = (H_1(x), H_2(x), \ldots)$. When $\sum_{i\geq 1} \hat{\pi}_i < \infty$, we can simplify $\sum_{i\geq 1} H_i(\pi) = 0$ to $\lambda - \hat{\pi}_1 = 0$. Hence, $\hat{\pi}_1$ must equal $\lambda$. The expressions for $\hat{\pi}_i$ then readily follow from the conditions $H_i(\hat{\pi}) = 0$, for $i \geq 1$. □

**Theorem 4.9.** *A predefined overall migration rate* $M$ *can be matched by setting* $p = p_{cp|M}$*, with*

$$\begin{aligned} p_{cp|M} &= \frac{M - \bar{\pi}_{T+1} r(1-\lambda)}{(\bar{\pi}_T - \bar{\pi}_{T+1}) r(1-\lambda)} \\ &= \frac{\lambda \left( r\left(-\lambda^2 + \lambda + M\right) \lambda^T - M(r+1) \right)}{(1-\lambda) r(r+1) \left( \lambda M - \left(-\lambda^2 + \lambda + M\right) \lambda^T \right)}. \end{aligned} \tag{4.30}$$

*When matching* $M$ *by setting* $p = p_{cp|M}$*,* $\hat{\pi}_i$ *reduces to*

$$\hat{\pi}_i = \lambda^i - \frac{M(1-\lambda^{i-1})}{1-\lambda}, \tag{4.31}$$

*for* $i \leq T$*.*

*Proof.* The fraction of empty queues $(1-\lambda)$ send probes at rate $r$. Probes are successful with probability 1 if they locate a queue with length at least $T+1$ $(\bar{\pi}_{T+1})$. Probes are successful with probability $p$ if they locate a queue with length equal to $T$ $(\bar{\pi}_T - \bar{\pi}_{T+1})$. In other words:

$$M_{cp} = r(1-\lambda)(\bar{\pi}_{T+1} + p(\bar{\pi}_T - \bar{\pi}_{T+1}), \tag{4.32}$$

from which (4.30) follows by algebraic manipulation. The reduction of $\hat{\pi}_i$ to (4.31) is found by substituting (4.30) in (4.28). □

**Theorem 4.10.** *The mean delay* $D_{cp}$ *of a job under the conditional pull strategy equals*

$$D_{cp} = \frac{1}{1-\lambda} \left( 1 - \frac{M_{cp}}{\lambda} (\alpha - \beta) \right), \tag{4.33}$$

*with*

$$\alpha = T + \frac{\lambda}{(1-\lambda)(1+r)} \quad \textit{and} \quad \beta = \frac{r(1-\lambda)p\bar{\pi}_T}{M_{cp}}.$$

*Proof.* The improvement in mean delay compared to a standard M/M/1 queue, can be expressed as a migration frequency $(M_{cp}/\lambda)$ times a migration gain $(\alpha - \beta)$. The fraction of migrating jobs where the job is pulled from a queue with length at least $T+1$, $(r(1-\lambda)\bar{\pi}_{T+1}/M_{cp})$ skip $\alpha$ places in the queue: The same remarks as in Theorem 4.2 apply. The other jobs $((r(1-\lambda)p(\bar{\pi}_T - \bar{\pi}_{T+1}))/M_{cp})$ are pulled from a queue with length equal to $T$, thus skipping exactly $T-1$ places. In other words, the migration gain can be expressed as:

$$\frac{\alpha r(1-\lambda)\bar{\pi}_{T+1} + (T-1)(r(1-\lambda)p(\bar{\pi}_T - \bar{\pi}_{T+1}))}{M_{cp}},$$

which can be rewritten as $\alpha - \beta$. □

Figure 4.3 shows the mean delay of the conditional pull strategy. The dots represent $\lambda_T$, i.e., the intersection points of $r_{pull|R}$ and $r_{both|M}$. The conditional pull strategy achieves a lower mean delay compared to the rate-based pull strategies with $T > 1$, as it transfers more jobs.

**Theorem 4.11.** *When $R = +\infty$ and $M$ is finite, the max-push and conditional pull strategies have the same stationary queue length distribution.*

*Proof.* When there is no probe limit $R$, the parameter $r$ is allowed to be arbitrarily large for the conditional pull strategy. In this case the maximum queue length will be $T$ as $\lim_{r\to\infty} \hat{\pi}_i = 0$ for $i > T$, see (4.29). We therefore know from Theorems 4.5 and 4.9 that both the max-push and conditional pull strategy have the same queue length distribution when only matching $M$, if they use the same $T$. What remains to be shown is that both strategies make use of the same $T$.

Recall that $\lambda_T$ was defined as the solution in $(0, 1)$ of $r_{pull|R} - r_{both|M}$, that is,

$$\frac{R}{1-\lambda} = \frac{M}{(\lambda(1-\lambda)+M)\lambda^T - M}.$$

The left hand side tends to infinity as $R$ tends to infinity. Hence, $r_{both|M}$ must tend to infinity, meaning $\lambda_T$ is the solution to $M = (1-\lambda)\lambda^{T+1}/(1-\lambda^T)$. The $\lambda_T$'s are thus exactly the $M$ values where the max-push strategy changes its $T$ value, see (4.18). Hence, both the conditional pull and max-push strategy choose the same $T$. □

### 4.5.1 Model Validation

We validate the infinite system model for the conditional pull strategy by comparing the closed form results of Theorem 4.10 with time consuming simulation results for systems with a finite number of nodes $N$. The infinite and finite system model only differ in the system size. Hence, the rate $r$ and probability $p$ in the simulation experiments is independent of $N$ and was determined by using the expression for $p$ from Equation (4.30) and $r = R/(1-\lambda)$. Each simulated point in the figures represents the average value of 25 simulation runs. Each run has a length of $10^6$ time units (where the service time is exponentially distributed with a mean of 1 time unit) and a warm-up period of length $10^6/3$ time units.

Table 4.1 compares the mean delay in a finite system with $N$ nodes with the mean delay in the infinite system model under the conditional pull strategy with $R = 1$ and $M = 0.1$ for $N = 25, 50, \ldots, 1600$ and $\lambda = 0.5, 0.65, 0.7, 0.75$ and $0.8$. For each combination of $N$ and $\lambda$ we also show the relative error. The error clearly decreases as $N$ grows, and is worse for larger $\lambda$ values.

The observed overall migration rate in the simulation is strictly lower than the predefined $M$, meaning less jobs will be transferred than anticipated. Hence, the mean delay in the simulation experiments is pessimistic. This error is in part due to the choice of $p$, which was determined using (4.30). This choice relies on the infinite system model whereas we are now studying a finite system. The relative error in the observed overall migration rate is nearly load-insensitive and decreases linearly as the system doubles in size, as shown in Table 4.2.

| | | Load ($\lambda$) | | | | |
|---|---|---|---|---|---|---|
| | | 0.5 | 0.65 | 0.7 | 0.75 | 0.8 |
| System Size (N) | 25 | 1.1e-2 | 1.7e-2 | 1.9e-2 | 2.4e-2 | 2.9e-2 |
| | 50 | 5.6e-3 | 8.2e-3 | 9.5e-3 | 5.7e-3 | 7.1e-3 |
| | 100 | 2.8e-3 | 3.9e-3 | 4.6e-3 | 5.7e-3 | 7.1e-3 |
| | 200 | 1.3e-3 | 2.0e-3 | 2.3e-3 | 2.7e-3 | 3.4e-3 |
| | 400 | 7.0e-4 | 1.0e-3 | 1.2e-3 | 1.3e-3 | 1.7e-3 |
| | 800 | 3.5e-4 | 5.0e-4 | 5.6e-4 | 6.6e-4 | 8.1e-4 |
| | 1600 | 1.6e-4 | 2.5e-4 | 2.6e-4 | 3.8e-4 | 4.3e-4 |

Table 4.1: Relative error of mean delay, given by (4.33), for the conditional pull strategy with $R = 1$ and $M = 0.1$ when compared to simulation results.

| N | 25 | 50 | 100 | 200 | 400 | 800 | 1600 |
|---|---|---|---|---|---|---|---|
| Rel. Err. | 4% | 2% | 1% | .5% | 0.25% | .13% | .064% |

Table 4.2: Relative error of the observed overall migration rate for finite system size when compared to the targeted migration rate $M$.

## 4.6 Push Versus Pull Strategies

We compare the performance of the max-push and the conditional pull strategies with a predefined overall probe limit $R$ and migration limit $M$ (using Theorems 4.7 and 4.10). The parameter $T$ is determined by the load $\lambda$, as each strategy is only defined for a specific $T$ given any $\lambda$ (see (4.13), (4.18) and (4.24)). For the max-push, the value for $T$ and $r$ is chosen to match the strictest constraint of either $R$ or $M$ depending on the load (see Theorem 4.6). For the conditional pull all idle servers probe with rate $r = R/(1-\lambda)$, and $p$ is chosen to match $M$ (see Theorem 4.9).

The mean delay of the max-push and conditional pull strategy with $M = 0.1$ and $R = 0.4$ is shown in Figure 4.4. The max-push strategy is limited by the probe limit when $\lambda > 1 - M/R$ and by the migration limit when $\sqrt{M} < \lambda < 1 - M/R$. The mean delay of the push strategy is one in case $\lambda < \sqrt{M}$, as all newly arriving jobs at a busy server can be migrated instantaneously to an empty server without violating the $R$ and $M$ constraints. For the conditional pull strategy the limiting factor is $R$ when $\lambda < \sqrt{M + M/R}$, and $M$ for $\lambda > \sqrt{M + M/R}$.

The intervals where both strategies are constrained by $M$ do not always overlap, i.e. $\sqrt{M + M/R}$ can be larger than $1 - M/R$, as is the case for $R = 1$ and $M = 0.3$. When $\sqrt{M + M/R} < 1 - M/R$ both strategies transfer the same number of jobs when $\sqrt{M + M/R} < \lambda < 1 - M/R$. However, the max-push will outperform the conditional pull as the average migration gain is larger. This is not unexpected as the max-push strategy avoids that queues become larger than $T$, whereas queues with a length exceeding $T$ exist for the conditional pull as it only sends random probes at a finite rate.

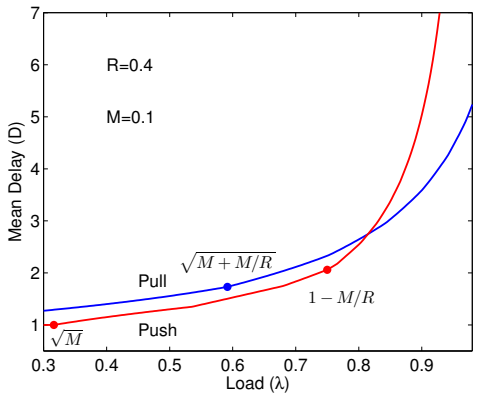


Figure 4.4: Mean delay of the max-push and conditional pull strategies, with $R = 0.4$ and $M = 0.1$.

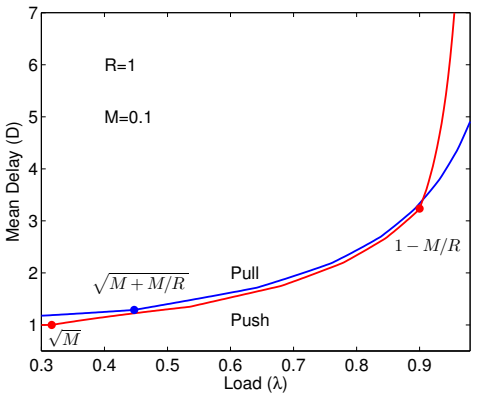


Figure 4.5: Mean delay of the max-push and conditional pull strategies, with $R = 1$ and $M = 0.1$.

As expected from Theorem 4.11, the difference in performance between max-push and conditional-pull becomes smaller when increasing $R$, as shown in Figure 4.5. As the empty queues send probes with rate $r = R/(1-\lambda)$, they are allowed to send more probes as $R$ increases. This increases the odds that a long queue is probed, thus lowering the mean delay. This can also be observed by looking at the values for $T$. By increasing $R$, a larger value for $T$ can be used for the same load. This requires that jobs are pulled from longer queues, increasing the migration gain per transfer.

In conclusion, whenever the maximum allowed probe rate $R$ clearly exceeds the maximum allowed migration rate $M$ (which is the case that is mainly of practical interest), the pull strategy is either clearly superior (for large $\lambda$) or has a similar performance to the max-push strategy (for medium to low $\lambda$).

## 4.7 Conclusion

In this chapter we defined the overall migration rate $M$ for all considered strategies, and showed that a predefined migration rate can be matched by setting the parameter $r$ to the appropriate value. This allowed us to rewrite the mean delay formulas in an equivalent form, where the gain over a simple M/M/1 queue is explicitly expressed.

We have shown that pull strategies with $T = 1$ are no longer optimal as was the case when only the overall probe rate was constrained as in Theorem 3.5. We introduced the new conditional pull strategy, that achieves a lower mean delay than rate-based pull with $T > 1$. For this variant we derived the stationary queue length distribution and mean delay in closed form, for a homogeneous network with Poisson arrivals and exponential service times under the infinite system model. Simulation results confirm that the infinite system model is an accurate approximation for finite systems of moderate size.

Theoretical results from earlier work provide proof that the infinite system model of the conditional pull strategy is the proper limit processes of the finite stochastic systems with $N$ nodes as $N$ tends to infinity over any finite timescale. Moreover, the convergence was shown to extend to the stationary regime by proving that the ODEs have a global attractor.

Comparing the max-push and conditional pull, we identified the $(\lambda, R)$ region where max-push outperforms conditional pull and quantified the difference in mean delay. We find that the push variant performs better in the interval where both strategies are constrained by $M$. The max-push has an advantage over the conditional pull since it can send probes at an infinite rate even for finite $R$, thus eliminating all queues longer than $T$. The conditional pull variant must discover long queues by means of random probing, and can not completely eliminate them. However, we have shown that for infinite $R$, meaning that only the overall probe limit $M$ should be respected, the conditional pull is equivalent in stationary queue length distribution to the max-push.

Chapter 5

# On a Class of Push and Pull Strategies with Single Migrations and Limited Probe Rate

*In this chapter*[1] *we introduce a general class of rate-based push and pull load balancing strategies, assuming there is no central dispatcher and nodes rely on probe messages for communication.*

*Under a pull strategy lightly loaded nodes send random probes in order to discover heavily loaded nodes, if such a node is found one task is transfered. Under a push strategy the heavily loaded nodes attempt to locate the lightly loaded nodes.*

*We show that by appropriately setting its parameters, rate-based strategies can be constructed that are equivalent with traditional or d-choices strategies.*

*Traditional strategies send a batch of $L_p$ probes at task arrival (push) or completion times (pull), whereas rate-based strategies send probes periodically.*

*Under the centralized/distributed d-choices strategy, $d$ or $d-1$ probes are sent in batch at arrival times and the task is transferred to the shortest queue discovered.*

*We derive closed form expressions for the mean delay for all considered strategies assuming a homogeneous network with Poisson arrivals and exponential job durations under the infinite system model.*

*We compare the performance of all strategies given that the same overall probe rate is used. We find that a rate-based push variant outperforms d-choices in terms of mean delay, at the cost of being more complex. A simple pull strategy is superior for high loads.*

[1] This chapter is based on work submitted to Performance Evaluation as "On a Class of Push and Pull Strategies with Single Migrations and Limited Probe Rate" by Wouter Minnebo and Benny Van Houdt.

## 5.1 Introduction

In previous work tasks could only be migrated to an empty server. However, for higher loads it becomes harder to find an empty server. In this situation a migration to a server that is lightly loaded but not empty can further reduce the mean delay. Therefore, we extend both the traditional and rate-based model to allow transfers to lightly loaded nodes, in this case nodes with at most $B$ jobs. Setting $B = 0$ only allows transfers to empty servers, reducing the models and closed form expressions to those found in Chapters 2 and 3, and also [10].

Furthermore, we develop several push models that achieve the same performance as the d-choice strategy when using the same number of probes, but without centralized load balancers.

This chapter makes the following contributions:

1. We introduce a general class of push and pull strategies, and describe its evolution in an infinite system model. We identify several subclasses by restricting the model parameters. For these subclasses, we find the stationary queue length distribution, allowing us to express the mean delay explicitly. Furthermore, we state as conjecture an optimal pull and push strategy for this general class of strategies.

2. We show that rate-based strategies achieve the same level of performance compared to traditional strategies, when using the same overall probe rate. Therefore, systems where it might be desirable to not send the probes at task arrival or completion instants are not at a disadvantage. In addition, rate-based strategies allow for more granular control over the overall probe rate, whereas the number of probes in a batch must be an integer for the traditional strategies.

3. We introduce several distributed versions of d-choices with an overall probe rate of $\lambda^2(1-\lambda^{d-1})/(1-\lambda)$, that are equivalent in performance compared to a centralized d-choices with $d$ probes per task.

4. We show that a rate-based push variant has a lower mean delay than d-choices, and the pull strategy remains best for high loads.

The chapter is structured as follows. In Section 5.2 we give an overview of the strategies considered in this chapter. Section 5.3 presents the infinite system models for a general rate-based push and pull strategy, considers a subclass corresponding to a particular choice of parameters, and covers the max-push strategy. Section 5.4 analyses the traditional pull and push strategies, and shows the equivalence with rate-based strategies. This equivalence was shown in Chapter 2 for $T = 1$ and $B = 0$. In section 5.5 we introduce a distributed version of the d-choices strategy, and derive two rate-based variants that are equivalent to the original d-choices strategy with respect to their stationary distribution. In Section 5.6, the best performing rate-based pull and push strategies are compared to the d-choice strategy.

## 5.2 Problem Description and Overview of Strategies

We consider a continuous-time system consisting of $N$ queues, where each queue consists of a single server with an infinite buffer. As in [7, 9, 19, 22], jobs arrive locally according to a Poisson process with rate $\lambda < 1$, and have an exponentially distributed duration with mean 1. Servers process jobs in first-come first-served order. Servers can send probe messages to each other to query for queue length information and to transfer jobs. We assume that the time required to transfer probe messages and jobs is sufficiently small in comparison with the processing time of a job, i.e., transfer times are considered zero. For a discussion on the impact of communication delay and transfer time we refer to [39, 40].

1. *Rate-based Push/Pull:* Whenever a server has $i$ tasks it generates probe messages according to a Poisson process with rate $r_i$. The node with length $j$ that is probed is selected at random and the transfer of one job from the highest to the lowest loaded server is allowed if $a_{i,j} = 1$, while no transfer takes place if $a_{i,j} = 0$. We will study several subclasses of this class.

2. *Traditional Push:* For every task arrival that would bring the queue length above $T$, the server first sends up to $L_p$ probes in sequence. The task is forwarded to the first discovered server with queue length $B$ or less. If no such server is found, the task is processed by the original server.

3. *Traditional Pull:* For each task completion that would bring the queue length to $B$ or less, the server first sends up to $L_p$ probes in sequence. A task is migrated from the first discovered server with queue length above $T$. If no such server is found, no further action is taken.

4. *Distributed d-Choices*: Nodes send $d-1$ probes on a task arrival instant and forward the job to the least loaded probed node, or process the task themselves if no shorter queue is found.

5. *Push-d-batch*: All servers that have tasks waiting generate probe events according to a Poisson process with rate $r_i$, where $i$ is the queue length. During each probe event, a batch of $d-1$ probes is sent and a task is migrated to the least loaded probed node if its queue length is smaller than $i-1$.

We study the different strategies using an infinite system model, i.e. as the number of queues in the system ($N$) tends to infinity. In the previous chapters we observed that the infinite system model is an accurate approximation for the finite case. A relative error of a few percent or less was observed when predicting the mean delay for $N \geq 100$. We make similar observations in Sections 5.3.2 and 5.3.4 for the strategies introduced in this paper.

## 5.3 Rate-based Strategies

In this section we introduce the infinite system model to assess the performance of rate-based push and pull strategies. First let us define a general rate-based strategy belonging to the class $\mathcal{S}(\mathbf{r}, A)$, with $\mathbf{r}$ a vector $(r_0, r_1, \ldots)$ and $A$ a matrix with elements $a_{i,j}$. The elements $r_i$ of vector $\mathbf{r}$ indicate at which rate a queue with length $i$ sends random probes. The elements $a_{i,j}$ of matrix $A$ indicate the probability that a probe from a queue with length $i$ to a queue with length $j$ results in a task transfer. This class of strategies only allows the transfer of a single task per probe. We refer to a strategy as a pull strategy if $a_{i,j} > 0$ implies $i < j$, and as a push strategy if $a_{i,j} > 0$ implies $i > j$.

The evolution of the queue lengths under this general strategy is modeled by a set of ODEs denoted as $dx(t)/dt = D(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$. As explained below, this set of ODEs can be written as

$$\frac{dx_i(t)}{dt} = \lambda(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)) + \alpha + \beta - \gamma - \delta \tag{5.1}$$

with $x_0(t) = 1$, and

$$\hat{\alpha} = (x_{i-1}(t) - x_i(t)) \sum_{j=i+1}^{\infty} r_j (x_j(t) - x_{j+1}(t)) a_{j,i-1}$$

$$\hat{\beta} = r_{i-1}(x_{i-1}(t) - x_i(t)) \sum_{j=i+1}^{\infty} (x_j(t) - x_{j+1}(t)) a_{i-1,j}$$

$$\hat{\gamma} = r_i(x_i(t) - x_{i+1}(t)) \sum_{j=0}^{i-2} (x_j(t) - x_{j+1}(t)) a_{i,j}$$

$$\hat{\delta} = (x_i(t) - x_{i+1}(t) \sum_{j=0}^{i-2} (x_j(t) - x_{j+1}(t)) r_j a_{j,i}$$

The terms $\lambda(x_{i-1}(t) - x_i(t))$ and $(x_i(t) - x_{i+1}(t))$ indicate arrivals and completions, respectively. The term $\hat{\alpha}$ indicates incoming transfers to queues with length $i - 1$ resulting from push request by longer queues. The term $\hat{\beta}$ indicates incoming transfers to queues with length $i - 1$ resulting from pull requests by those queues. The term $\hat{\gamma}$ indicates outgoing transfers resulting from push requests made by queues with length $i$. The term $\hat{\delta}$ indicates outgoing transfers resulting from pull requests made by shorter queues to queues with length $i$.

Before further analyzing the performance of these strategies, we point out two technical issues regarding the infinite system model. These issues arise from having an infinite dimensional state space $E$. Replacing the infinite size buffer in each node by a finite large buffer (such that the loss rate can be neglected) would result in a finite dimensional (compact) space $E$ and would resolve most of the issues. This also explains why large finite buffers are often considered as opposed to infinite buffers (see [9, 10]). We only briefly outline the issues here, and refer to Chapter 2 for a complete discussion and the relevant proofs for selected strategies are given in Chapters 2 and 3. We expect similar results can be obtained for the strategies

introduced in this chapter, however a formal treatment is outside the scope of this work.

The first issue is establishing that a given set of ODEs describes the proper limit process of the corresponding finite systems for any finite time horizon $[0, T]$. This can be resolved by relying on the generalization of Kurtz's theorem from [23, Theorem 3.13], and showing that a suitable environment around a sample path exists.

The second issue is determining whether this convergence extends to the stationary regime. Here it suffices to show that the set of ODEs describing the evolution of the system has a unique fixed point that is a global attractor. In addition, it must be shown that the unique stationary measure of the corresponding finite system converges to this fixed point as the size of the system tends to infinity, e.g. by relying on [32, Corollary 1].

For the remainder of this chapter we assume these issues can be resolved appropriately for all considered strategies. In other words, we assume the obtained fixed point of the set of ODEs describing the system is unique and a global attractor, and the infinite system model is the correct limiting process both on a finite timescale and in the stationary regime.

In the next sections we simplify Equation (5.1) by restricting the choice of $\mathbf{r}$ and $A$, resulting in explicit expressions for the unique fixed point and mean delay.

### 5.3.1 Fixed Rate Push and Pull

To restrict the set of pull and push strategies we state that a queue is long if it contains more than $T$ tasks and that a queue is short if it has at most $B$ tasks, with $B < T$. In addition, only one group of queues (be it long or short) sends probes independently of the queue length with rate $r$. We do not consider hybrid strategies as results in Chapter 2 indicate that a pure pull or push strategy is superior. We allow only transfers from long queues to short queues. In other words, for the fixed rate pull strategy

$$\begin{cases} r_i = 0 & \text{if } i > B \\ r_i = r & \text{if } i \leq B \end{cases}$$

and $a_{i,j}$ is one if $i \leq B$ and $j > T$ and zero otherwise. Likewise, for the fixed rate push strategy

$$\begin{cases} r_i = 0 & \text{if } i \leq T \\ r_i = r & \text{if } i > T \end{cases}$$

and $a_{i,j}$ is one if $i > T$ and $j \leq B$ and zero otherwise.

The evolution of both the fixed rate pull and push strategy is modeled by a set of ODEs denoted as $dx(t)/dt = F(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$. This is a simplification of Equation (5.1), and the ODEs can be written as

$$\frac{dx_i(t)}{dt} = (\lambda + r x_{T+1}(t))(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)), \tag{5.2}$$

for $1 \leq i \leq B + 1$ with $x_0(t) = 1$, and

$$\frac{dx_i(t)}{dt} = \lambda(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)), \tag{5.3}$$

for $B+2 \leq i \leq T$, and

$$\frac{dx_i(t)}{dt} = \lambda(x_{i-1}(t)-x_i(t))-(x_i(t)-x_{i+1}(t))-r(1-x_{B+1}(t))(x_i(t)-x_{i+1}(t)) \quad (5.4)$$

for $i > T$.

In the next Theorem we express the fixed point for this set of ODEs.

**Theorem 5.1.** *The set of ODEs given by (5.2-5.4) has a unique fixed point* $\pi = (\pi_1, \pi_2, \ldots)$ *with* $\sum_{i\geq 1} \pi_i < \infty$. *Let* $\eta_i = \pi_i - \pi_{i+1}$ *and* $\eta_0 = 1-\lambda$, *then the fixed point can be expressed as*

$$\eta_i = (1-\lambda)(\lambda + r\pi_{T+1})^i, \qquad 1 \leq i \leq B+1 \quad (5.5)$$

$$\eta_i = \eta_{B+1}\lambda^{i-(B+1)}, \qquad B+2 \leq i \leq T \quad (5.6)$$

$$\eta_i = \eta_T \left(\frac{\lambda}{1+r(1-\pi_{B+1})}\right)^{i-T}, \qquad i > T. \quad (5.7)$$

*Proof.* The expressions for $\eta_i$ readily follow from setting$dx_i(t)/dt = 0$ in Equations (5.2-5.4), and observing that $\pi_1 = \lambda$ due to the requirement $\sum_{i\geq 1} \pi_i < \infty$. □

We can now express the main performance measures of these push and pull strategies via Equations (5.5-5.7). First, we note that the overall probe rate for push strategies equals

$$R_{push} = r_{push}\pi_{T+1}, \quad (5.8)$$

as all queues with length $T+1$ or more send probes with rate $r_{push}$. Similarly, the overall probe rate of the pull strategy equals

$$R_{pull} = r_{pull}(1-\pi_{B+1}), \quad (5.9)$$

as all queues with length $B$ or less send probes with rate $r_{pull}$. Furthermore, the total migration rate is

$$M = r(1-\pi_{B+1})\pi_{T+1}.$$

From a push perspective a fraction of nodes ($\pi_{T+1}$) sends probes at rate $r$, succeeding with probability $(1-\pi_{B+1})$. From a pull perspective the roles of senders and receivers are reversed. Now we can formulate the mean delay:

**Theorem 5.2.** *The mean delay D of a job under the fixed rate push or pull strategy equals*

$$D_{both} = \frac{1}{1-\lambda}\left(1 - \frac{M}{\lambda}\gamma\right),$$

*with*

$$\gamma = T - B + \alpha + \delta,$$

$$\alpha = \sum_{i=T+2}^{\infty} \frac{(i-(T+1))\eta_i}{\pi_{T+1}} = \frac{\lambda}{1-\lambda+r(1-\pi_{B+1})},$$

$$\delta = \sum_{i=0}^{B-1} \frac{(B-i)\eta_i}{1-\pi_{B+1}} = \frac{(1-\lambda)(B(1-\lambda-r\pi_{T+1}) - (\lambda+r\pi_{T+1})(1-(\lambda+r\pi_{T+1})^B))}{(1-\pi_{B+1})(1-\lambda-r\pi_{T+1})^2}.$$

*Proof.* We use a similar argument as in Chapter 4, where we showed that the improvement over the mean delay of an M/M/1 queue can be formulated as a migration frequency ($M/\lambda$) and migration gain ($\gamma$). The migration frequency denotes how many migrations per job take place on average and the migration gain quantifies the number of places in the queue the migrating job skips. Therefore, the total improvement is given by how many migrations take place on average per task multiplied by how many places in the queue a migrating task skips.

Migrating tasks skips on average $\gamma$ places in the queue. All tasks skip $T - B$ places by construction of the strategy. Tasks can skip more places depending on the length of the queue sending the task, accounting for $\alpha$ places on average. We note this equals the average number of customers in an M/M/1 queue with service rate $1 + r(1 - \pi_{B+1})$. Tasks can also skip more places depending on the length of the queue receiving the task, accounting for $\delta$ places on average. □

Although we can express the fixed point in closed form, it is not straightforward to compute as the values for $\pi_{B+1}$ and $\pi_{T+1}$ are unknown a priori. For the pull strategy both $\pi_{B+1}$ and $\pi_{T+1}$ were found numerically with a bisection algorithm, as detailed in Appendix D.

For the push strategy, $\pi_{B+1}$ and $\pi_{T+1}$ can be computed directly. Typically, we want to match a predefined probe rate $R$, which equals $r_{push}\pi_{T+1}$ for the push strategy. So when using a push strategy and matching $R$, we can simply substitute $R$ instead of $r\pi_{T+1}$ and compute the fixed point distribution immediately. However, when $R$ is relatively large this will result in a negative value for $\pi_{T+1}$. This indicates that queues can send probes at an infinite rate without exceeding the overall probe limit $R$, thereby instantly finding migration targets for all tasks from queues with length $T + 1$ or more, and reducing $\pi_{T+1}$ to zero. This is illustrated in Figure 5.1, where the load at which $\pi_{T+1}$ reaches zero is marked with a dot. For all loads lower than this point the substitution we performed (using $R$ instead of $r\pi_{T+1}$) is no longer valid, and computing $\pi_{T+1}$ yields a negative result. In this case the push strategy with the current $B, T$ and $\lambda$ parameters uses less probes than allowed by the overall probe limit $R$, as all tasks that are eligible to migrate are instantly exhausted. The behavior of a push strategy with infinite $r$ is equivalent with the max-push strategy with $r_{mp} = 0$, covered in Section 5.3.3.

**Conjecture 5.1.** *The optimal choice for a rate-based pull strategy in class $\mathcal{S}(\mathbf{r}, A)$ given an overall probe rate $R$ is a fixed rate pull strategy with $B = 0$ and $T = 1$.*

In Theorem 3.5 it was shown that if $B = 0$, setting $T = 1$ is optimal. Intuitively, increasing $T$ makes it less likely that a probe is successful. Similarly, a non-empty server is just as likely to locate a queue with length at least $T$ than an empty server. And the tasks can skip more places in the queue if the request was sent by the empty server. Therefore, intuitively setting $B = 0$ and $T = 1$ appears optimal for the rate-based pull strategy.

For the push strategy setting $B = 0$ is not optimal as shown in Figure 5.2. Increasing $B$ improves the performance of the push under moderate to high loads. We observed that increasing $T$ higher than $B + 2$ is not beneficial, as setting the parameter $B$ to $B + 1$ yields a lower mean delay for that load. Therefore, such settings are not shown.

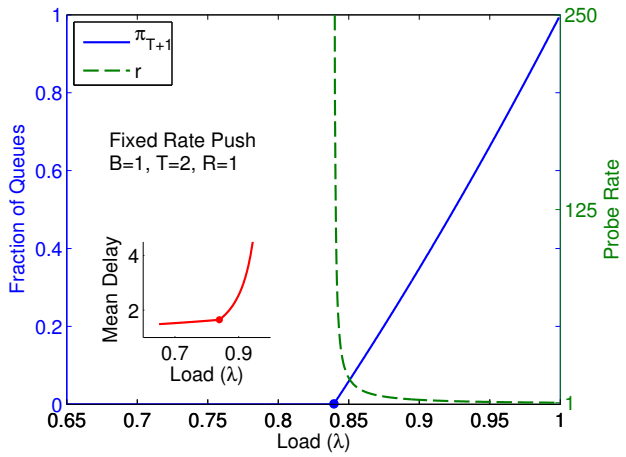


Figure 5.1: The probe rate of individual queues ($r$) and the fraction of queues allowed to send probes ($\pi_{T+1}$), shown for the fixed rate push strategy with $B = 1, T = 2$ and $R = 1$. The probe rate $r$ goes to infinity as the fraction of queues with at least $T+1$ tasks ($\pi_{T+1}$) reaches zero. The load $\lambda$ at which this occurs is marked with a dot. This is also the point where the behavior of the mean delay changes, as shown in the inset plot.

### 5.3.2 Numerical Validation of Fixed Rate Push

| $N$ | $B = 1$ | | | | | $B = 2$ |
|---|---|---|---|---|---|---|
| | $T = 2$ | | | $T = 3$ | | |
| | $\lambda = 0.85$ | $\lambda = 0.90$ | $\lambda = 0.95$ | $\lambda = 0.90$ | $\lambda = 0.95$ | $\lambda = 0.95$ |
| 25 | 4.15e-2 | 8.78e-2 | 1.17e-1 | 5.42e-2 | 1.28e-1 | 1.43e-1 |
| 50 | 1.70e-2 | 4.21e-2 | 5.77e-2 | 2.00e-2 | 6.28e-2 | 6.60e-2 |
| 100 | 7.68e-3 | 2.07e-2 | 2.92e-2 | 7.95e-3 | 3.12e-2 | 3.17e-2 |
| 200 | 3.60e-3 | 1.04e-2 | 1.50e-2 | 3.49e-3 | 1.58e-2 | 1.52e-2 |
| 400 | 1.76e-3 | 5.07e-3 | 7.19e-3 | 1.62e-3 | 7.68e-3 | 7.54e-3 |
| 800 | 8.74e-4 | 2.53e-3 | 3.76e-3 | 7.94e-4 | 3.88e-3 | 3.76e-3 |
| 1600 | 4.22e-4 | 1.25e-3 | 1.79e-3 | 3.96e-4 | 2.04e-3 | 1.94e-3 |

Table 5.1: The relative error of the mean delay $D$ in a finite system with size $N$ using the fixed rate push strategy, compared to the infinite system model. We note that the infinite system model is optimistic with respect to the performance of the finite system.

In this section we present validation results for the fixed rate push strategy with $B \geq 1$ as the model for push and pull strategies with $B = 0$ was already validated in Chapter 3 and we conjecture that the mean delay of the pull strategy is minimized for $B = 0$ and $T = 1$. The infinite system model and simulation setup only differ in the system size. The rate $r_{push}$ in the simulation experiments is independent of $N$ and was determined by $\lambda$ and $R$ using the expression for $R_{push}$ in (5.8), we choose $R = 1$ in all experiments. Each entry in the tables represents the average value of 25 simulation runs. Each run has a length of $10^6$ (where the service time is exponentially distributed with mean 1) and a warm-up period of length $10^6/3$.

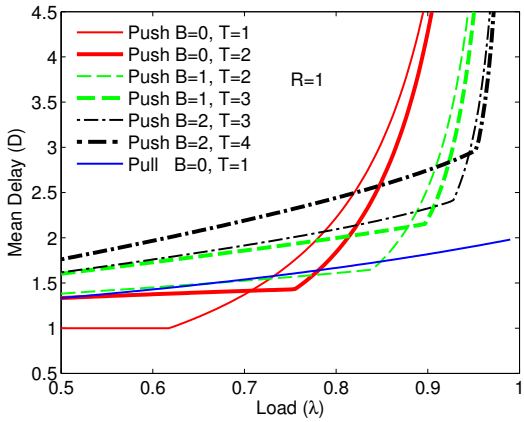


Figure 5.2: Mean delay of the push strategy, with $T = B+1$ and $T = B+2$ for $B = 0, 1, 2$. Also shown is the pull strategy with $B = 0$ and $T = 1$. All strategies use $R = 1$.

| $N$ | $B = 1$ | | | | | $B = 2$ |
|---|---|---|---|---|---|---|
| | $T = 2$ | | | $T = 3$ | | |
| | $\lambda = 0.85$ | $\lambda = 0.90$ | $\lambda = 0.95$ | $\lambda = 0.90$ | $\lambda = 0.95$ | $\lambda = 0.95$ |
| 25 | 4.75e-1 | 1.05e-1 | 2.80e-2 | 1.45e+0 | 7.13e-2 | 2.67e-1 |
| 50 | 1.98e-1 | 5.36e-2 | 1.40e-2 | 5.27e-1 | 3.69e-2 | 1.41e-1 |
| 100 | 8.75e-2 | 2.74e-2 | 7.37e-3 | 1.96e-1 | 1.88e-2 | 7.30e-2 |
| 200 | 4.07e-2 | 1.40e-2 | 3.87e-3 | 8.20e-2 | 9.85e-3 | 3.65e-2 |
| 400 | 1.98e-2 | 6.86e-3 | 1.80e-3 | 3.73e-2 | 4.71e-3 | 1.85e-2 |
| 800 | 9.75e-3 | 3.42e-3 | 9.82e-4 | 1.79e-2 | 2.39e-3 | 9.32e-3 |
| 1600 | 4.78e-3 | 1.71e-3 | 4.62e-4 | 8.80e-3 | 1.28e-3 | 4.81e-3 |

Table 5.2: The relative error of the overall probe rate $R$ in a finite system with size $N$ using the fixed rate push strategy, compared to the infinite system model. We note that when using the $r$ as derived from the infinite system model, the finite system produces a higher overall probe rate than requested.

Table 5.1 shows the relative error in mean delay, observed when comparing a finite system with size $N$ to the infinite system model. As expected, the error decreases as the system grows in size, with at most a few percent relative error as the system reaches 100 nodes. Changing values for $T$ and $B$ can either increase or decrease the error. For example taking $B = 1$ and $\lambda = 0.90$, increasing $T$ from 2 to 3 decreases the error, but with $\lambda = 0.95$ the same change increases the error. The error also increases with the load. The infinite system model is optimistic, underestimating the observed mean delay.

We should note that the actual overall probe rate observed in the finite system exceeds the requested $R$, as shown in Table 5.2. In other words, the relation between $R_{push}$ and $r_{push}$ given by (5.8) is not very accurate for small $N$ values as the infinite model is optimistic with respect to the queue length distribution. However, as the finite system grows in size, the actual overall probe rate converges to the one requested.

### 5.3.3 The Max-Push Strategy

As we noted in the previous section, the fixed rate push strategy can not match the predefined overall probe rate in case $R$ is larger than needed to instantly find migration targets for all tasks from queues with length $T+1$ or more. This effectively eliminates all queues longer than $T$, without using the full $R$ budget. The idea of the max-push strategy is to migrate all new arrivals at a queue with length $T$ instantly to an eligible server, and let the queues with length exactly $T$ probe with rate $r_{mp}$. We later show how to choose $r_{mp}$, $B$ and $T$ such that the resulting overall probe rate matches $R$.

Formally the max-push strategy is a member of $\mathcal{S}(\mathbf{r}, A)$ and defined as follows. Let $r_T = r_{mp}$ and $r_{T+1} = \infty$, with the other entries of $\mathbf{r}$ set to zero. Let $a_{i,j}$ be one in case $i$ is either $T$ or $T+1$, and $j \leq B$.

In Chapter 3 the max-push strategy was introduced for $B = 0$, which we now generalize for $B > 0$. We discern two cases: $T > B+1$ and $T = B+1$.

In case $T > B+1$, the evolution of the max-push strategy is given by a set of ODEs denoted as $dx(t)/dt = H(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$. This is an adaptation of Equation (5.1) and this set of ODEs can be written as

$$\frac{dx_i(t)}{dt} = \left(\lambda + \frac{\lambda x_T(t)}{1 - x_{B+1}(t)} + r x_T(t)\right)(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)), \quad (5.10)$$

for $1 \leq i \leq B+1$, and

$$\frac{dx_i(t)}{dt} = \lambda(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)), \quad (5.11)$$

for $B+2 \leq i < T$, and

$$\frac{dx_T(t)}{dt} = \lambda(x_{T-1}(t) - x_T(t)) - x_T(t)(1 + r(1 - x_{B+1}(t))). \quad (5.12)$$

Note that all new arrivals at queues of length $T$ are migrated to servers with a maximum length of $B$, as indicated by $\lambda x_T(t)$ in (5.10). Probes are sent to random servers with equal probability for each server. Consequently, the migrations from new arrivals at queues of length $T$ are uniformly distributed across servers with length $B$ or less. Therefore, these migrations arrive at a queue with length $i-1$ with probability $(x_{i-1}(t) - x_i(t))/(1 - x_{B+1}(t))$, increasing the fraction of servers with queue length $i \leq B+1$.

For the case $T > B+1$ all migrations have the same target, specifically queues with length at most $B$. This is no longer true if we allow $T = B+1$. The new arrivals at a queue with length $T$ can be migrated to any queue with length at most $B$. However, a probe from a queue with length $T$ should find a target with length at most $B-1$ in order for the migration to result in a delay reduction. Therefore, the evolution of the system is described by a different set of ODEs given below.

In case $T = B+1$, the evolution of the max-push strategy is given by a set of ODEs denoted as $dx(t)/dt = I(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$. As explained below, this set of ODEs can be written as

$$\frac{dx_i(t)}{dt} = \left(\lambda + \frac{\lambda x_T(t)}{1 - x_T(t)} + r x_T(t)\right)(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)), \quad (5.13)$$

for $1 \leq i \leq B$, and

$$\frac{dx_T(t)}{dt} = \lambda(x_{T-1}(t) - x_T(t)) - x_T(t)(1 + r(1 - x_{T-1}(t))) + \frac{\lambda x_T(t)}{1 - x_T(t)}(x_{T-1}(t) - x_T(t)). \tag{5.14}$$

The same remarks as for $H(x(t))$ apply, with a modification in $dx_T(t)/dt$. Queues with length $T$ can now also be created by migrating an arrival in a queue with length $T$, to a queue with length $T-1$. This corresponds with the term $\lambda x_T(t)(x_{T-1}(t) - x_T(t))/(1 - x_T(t))$ in (5.14). Queues with length $T$ ($x_T(t)$) again send probes with rate $r$, and are now successful with probability $1 - x_{T-1}(t)$.

The sets of ODEs $H(x(t))$ and $I(x(t))$ have a unique fixed point $\dot{\pi}$ and $\hat{\pi}$, respectively. We derive the formulas for these fixed points further on, expressing the overall probe rate and migration rate first.

For both cases ($T > B + 1$ and $T = B + 1$) the overall probe rate can be formulated as

$$R_{mp} = \frac{\lambda \breve{\pi}_T}{1 - \breve{\pi}_{B+1}} + r_{mp}\breve{\pi}_T, \tag{5.15}$$

with $\breve{\pi}_i$ equal to $\dot{\pi}$ or $\hat{\pi}$ depending on the value of $T$. This relation states the following: new arrivals at a queue of length $T$ ($\lambda\breve{\pi}_T$) must find a server to migrate to, and find one on average by spending $1/(1 - \breve{\pi}_{B+1})$ probes. Queues with length $T$ ($\breve{\pi}_T$) also send probes at the finite rate $r_{mp}$.

Similarly we can define the migration rate, e.g., the rate at which probes are successful:

$$M_{mp|T>B+1} = \lambda\dot{\pi}_T + r_{mp}\dot{\pi}_T(1 - \dot{\pi}_{B+1})$$
$$M_{mp|T=B+1} = \lambda\hat{\pi}_T + r_{mp}\hat{\pi}_T(1 - \hat{\pi}_B).$$

For both cases ($T > B + 1$ and $T = B + 1$) new arrivals at a queue with length $T$ ($\lambda\breve{\pi}_T$) are migrated. The rest of the migrations are due to probes sent at rate $r_{mp}$ by queues with length $T$ ($\breve{\pi}_T$). These are successful with probability $(1 - \dot{\pi}_{B+1})$ in case $T > B + 1$ and with probability $(1 - \hat{\pi}_B)$ in case $T = B + 1$.

Having expressed the overall probe rate and migration rate, the fixed points are given in the next two theorems.

**Theorem 5.3.** *The set of ODEs given by (5.10-5.12) has a unique fixed point $\dot{\pi} = (\dot{\pi}_1, \ldots, \dot{\pi}_T)$ with $\sum_{i\geq 1} \dot{\pi}_i < \infty$ that can be expressed by letting $\dot{\eta}_i = \dot{\pi}_i - \dot{\pi}_{i+1}$ and $\dot{\eta}_0 = 1 - \lambda$. Then one finds the following expressions for the fixed point*

$$\dot{\eta}_i = (1 - \lambda)(\lambda + R_{mp})^i, \qquad 1 \leq i \leq B + 1$$
$$\dot{\eta}_i = \dot{\eta}_{B+1}\lambda^{i-(B+1)}, \qquad B + 2 \leq i < T.$$

*Proof.* From the requirement $\sum_{i\geq 1} \dot{\pi}_i < \infty$ it is clear that $\dot{\pi}_1 = \lambda$. Then setting $dx_i(t)/dt = 0$ in Equations (5.10-5.12), directly yields the expressions for $\dot{\eta}_i$. □

**Theorem 5.4.** *The set of ODEs given by (5.13-5.14) has a unique fixed point $\hat{\pi} = (\hat{\pi}_1, \ldots, \hat{\pi}_T)$ with $\sum_{i\geq 1} \hat{\pi}_i < \infty$. that can be expressed by letting $\hat{\eta}_i = \hat{\pi}_i - \hat{\pi}_{i+1}$ and $\hat{\eta}_0 = 1 - \lambda$. Then we can express the fixed point as*

$$\hat{\eta}_i = (1 - \lambda)(\lambda + R_{mp})^i \qquad 1 \leq i \leq B.$$

*Proof.* We note that $\hat{\pi}_1 = \lambda$ due to the requirement $\sum_{i\geq 1} \hat{\pi}_i < \infty$. Then we find the expressions for $\eta_i$ from setting $dx_i(t)/dt = 0$ in Equations (5.13-5.14). □

The stationary queue length distribution can be computed directly when matching a predefined $R$. In case $R$ is not given in advance, for example if only $r_{mp}$ is given, a similar procedure to the algorithm in Appendix D can be used to find $\breve{\pi}_T$ and $\breve{\pi}_{B+1}$.

From the formulation of the max-push strategy it is clear that there is a requirement on $R$, for the strategy to be well-defined. If $R$ is too low, not all new arrivals at a queue of length $T$ can be migrated. If $R$ is too high, queues with length $T$ will be exhausted and we face the same problem as before. We provide the following method to determine a valid parameter set: Let $\Gamma(B, T, R, \lambda)$ be the value of $\pi_{T+1}$ as calculated by (5.5-5.7). Now we discern two cases. If $T > B+1$, then for a given $B$ and $\lambda$, $T$ must be chosen such that $\Gamma(B, T-1, R, \lambda) > 0$ and $\Gamma(B, T, R, \lambda) < 0$. If $T = B+1$, for a given $\lambda$, $T$ must be chosen such that $\Gamma(B, T+1, R, \lambda) < 0$, and $\Gamma(B-1, T, R, \lambda) > 0$.

We can now express the main performance measures of the max-push strategy via Theorems 5.3 and 5.4:

**Theorem 5.5.** *The mean delay $D$ of a job under the max-push strategy with $T \geq B+1$, equals*

$$D_{mp} = \frac{1}{1-\lambda}\left(1 - \frac{M_{mp}}{\lambda}\delta\right),$$

*with*

$$\delta = T - B - 1 + \frac{\lambda\breve{\pi}_T}{M_{mp}} + \beta \qquad\qquad \beta = \frac{\sum_{i=0}^{B-1}(B-i)\breve{\eta}_i}{1-\breve{\pi}_{B+1}}$$

*Proof.* Here, $\breve{\pi}$ and $\breve{\eta}$ is used to denote $\dot{\pi}$ and $\dot{\eta}$ or $\hat{\pi}$ and $\hat{\eta}$, in case $T > B+1$ or $T = B+1$ respectively. Also $M_{mp}$ is to be substituted with $M_{mp|T>B+1}$ or $M_{mp|T=B+1}$, depending on the values for $T$ and $B$.

The reasoning is the same as in Theorem 5.2. Migrating tasks skip on average $\delta$ places in the queue.

All tasks skip $T - B - 1$ places by construction of the strategy. The fraction of migrating arrivals at a queue of length $T$ skips one extra place ($\lambda\breve{\pi}_T/M_{mp}$). Tasks can skip more places depending on the length of the queue receiving the task, accounting for $\beta$ places on average. □

Figures 5.3 and 5.4 show the mean delay of the max-push strategy, for $T > B+1$ and $T = B+1$, respectively. The max-push connects the points where the push can no longer match $R$. Connected points all use the same value for parameter $B$.

**Conjecture 5.2.** *The optimal choice for a rate-based push strategy in class $\mathcal{S}(\mathbf{r}, A)$ is a max-push strategy with $T = B+1$.*

Intuitively, it appears desirable to let the longer queues spend as much of the probe budget as possible. The choice of $T = B+1$ indicates that a task is transfered if the transfer results in a lower mean delay without further constraints on how much this gain should be.

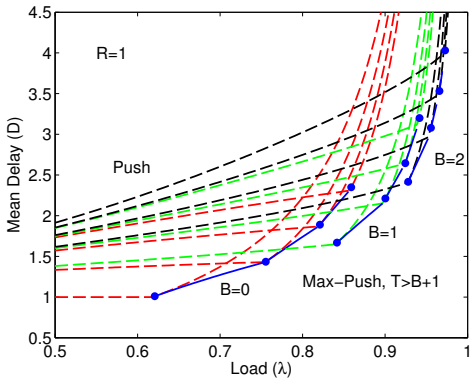


Figure 5.3: Mean delay of the max-push strategy, with $T > B+1$ and for $B = 0, 1, 2$, using $R = 1$.

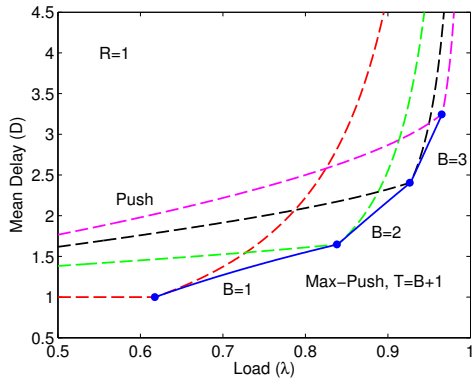


Figure 5.4: Mean delay of the max-push strategy, with $T = B+1$ for $B = 1, 2$, using $R = 1$.

| $N$ | $B=1$ | | | | $B=2$ | |
|---|---|---|---|---|---|---|
| | $T=2$ $\lambda=0.80$ | $T=3$ $\lambda=0.875$ | $T=4$ $\lambda=0.915$ | $T=5$ $\lambda=0.935$ | $T=3$ $\lambda=0.915$ | $T=4$ $\lambda=0.95$ |
| 25 | 2.07e-2 | 4.10e-2 | 6.19e-2 | 7.94e-2 | 6.25e-2 | 1.17e-1 |
| 50 | 7.93e-3 | 1.44e-2 | 2.03e-2 | 2.66e-2 | 2.22e-2 | 4.16e-2 |
| 100 | 3.70e-3 | 6.18e-3 | 7.44e-3 | 8.85e-3 | 9.04e-3 | 1.47e-2 |
| 200 | 1.81e-3 | 2.92e-3 | 3.26e-3 | 3.60e-3 | 4.14e-3 | 5.88e-3 |
| 400 | 9.20e-4 | 1.47e-4 | 1.60e-3 | 1.74e-3 | 2.06e-3 | 2.64e-3 |
| 800 | 4.25e-4 | 7.25e-4 | 7.61e-4 | 8.42e-4 | 1.04e-3 | 1.25e-3 |
| 1600 | 2.15e-4 | 3.74e-4 | 4.11e-4 | 4.35e-4 | 5.10e-4 | 6.26e-4 |

Table 5.3: The relative error of the mean delay $D$ in a finite system with size $N$ using the max-push strategy, compared to the infinite system model. We note that the infinite system model is optimistic with respect to the performance of the finite system.

### 5.3.4 Numerical Validation of Max-Push

We compare the predictions of the infinite system model with respect to a finite system using the max-push strategy with $B \geq 1$ in this section. The setting $B = 0$ was already discussed in Chapter 3. The experimental setup is the same as in Section 5.3.2, we choose $R_{mp} = 1$ for all experiments and determined $r_{mp}$ using (5.15).

In Table 5.3 we show the relative error of the mean delay observed in the finite system, compared to the infinite system model. The error decreases as the system grows larger, and is smaller for lower loads. Overall, the mean delay is accurately predicted with a relative error of at most a few percent as the system size reaches 50 nodes. The infinite system model is optimistic, predicting a lower mean delay than observed in a finite system.

The relative error of the overall probe rate is shown in Table 5.4. In all cases the finite system uses more probes than the requested overall probe rate $R$. Again

| $N$ | $B = 1$ | | | | $B = 2$ | |
|---|---|---|---|---|---|---|
| | $T = 2$ $\lambda = 0.80$ | $T = 3$ $\lambda = 0.875$ | $T = 4$ $\lambda = 0.915$ | $T = 5$ $\lambda = 0.935$ | $T = 3$ $\lambda = 0.915$ | $T = 4$ $\lambda = 0.95$ |
| 25 | 4.20e-1 | 8.17e-1 | 1.25e+0 | 1.53e+0 | 1.56e+0 | 2.69e+0 |
| 50 | 1.45e-1 | 3.67e-1 | 7.48e-1 | 1.06e+0 | 7.09e-1 | 2.00e+0 |
| 100 | 5.23e-2 | 1.16e-1 | 2.78e-1 | 4.84e-1 | 1.79e-1 | 8.75e-1 |
| 200 | 2.37e-2 | 4.58e-2 | 8.59e-2 | 1.46e-1 | 5.60e-2 | 2.04e-1 |
| 400 | 1.15e-2 | 2.14e-2 | 3.60e-2 | 5.22e-2 | 2.55e-2 | 5.73e-2 |
| 800 | 5.51e-3 | 1.04e-2 | 1.69e-2 | 2.38e-2 | 1.23e-2 | 2.52e-2 |
| 1600 | 2.78e-3 | 5.18e-3 | 8.45e-3 | 1.15e-2 | 6.01e-3 | 1.21e-2 |

Table 5.4: The relative error of the overall probe rate $R$ in a finite system with size $N$ using the max-push strategy, compared to the infinite system model. We note that when using the $r_{mp}$ as derived from the infinite system model, the finite system produces a higher overall probe rate than requested.

the error decreases as the system grows in size. However, for high loads and a small system size we observe that the observed overall probe rate is much larger than requested, with a relative error as high as 2.69. This is due to the fact that in a small system there is a higher probability that there will be some periods that all nodes have $B$ or more tasks. If that happens, a new arrival at a queue with length $T$ can not find an instantaneous transfer target, but will spend many probes trying. In the infinite system model this is never a problem, but in a finite system it does occur. In our simulation we allow $N$ probes (without replacement) for such a task, so all queues have been sampled. And if no eligible migration target is found, the queue where the task originally arrived still accepts the task. As the system becomes larger this situation occurs less frequently or not at all.

## 5.4 Traditional Strategies

In this section we analyze the traditional strategies, where probes are not sent periodically but only on task arrival or completion instants. Probes are sent sequentially until an eligible target for migration is found, or the maximum of $L_p$ probes is reached.

We also show that fixed rate strategies as discussed in Section 5.3 can be constructed that use the same overall probe rate and result in the same stationary queue length distribution as the traditional strategies.

### 5.4.1 Traditional Push

In the traditional push variant, up to $L_p$ probes are sent when a new task arrives at a queue with length at least $T$. The task is migrated to the first node discovered that has at most $B$ tasks. A similar setup was studied in [18] using birth-death models, with the constraint that $T = B + 1$.

The evolution of the traditional push strategy is modeled by a set of ODEs denoted as $dx(t)/dt = J(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents

the fraction of the number of nodes with at least $i$ jobs at time $t$. As explained below, this set of ODEs can be written as

$$\frac{dx_i}{dt} = \lambda(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)) + \lambda x_T(t)(1 - x_{B+1}(t)^{L_p})\frac{x_{i-1}(t) - x_i(t)}{1 - x_{B+1}(t)}, \tag{5.16}$$

for $1 \leq i \leq B+1$. For $B+2 \leq i \leq T$ we have

$$\frac{dx_i}{dt} = \lambda(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)), \tag{5.17}$$

and for $i > T$ we have

$$\frac{dx_i}{dt} = \lambda(x_{i-1}(t) - x_i(t))x_{B+1}(t)^{L_p} - (x_i(t) - x_{i+1}(t)). \tag{5.18}$$

An arrival at a queue with length at least $T$ is not transfered if no lightly loaded node is found with $L_p$ probes, this occurs with probability $x_{B+1}(t)^{L_p}$. So with probability $1 - x_{B+1}(t)^{L_p}$ a new arrival at a queue with length at least $T$ (occurring at rate $\lambda x_T(t)$) is migrated to a lightly loaded node. Since each server has the same probability of being probed, the migrating tasks are distributed uniformly over the lightly loaded nodes $((x_{i-1}(t) - x_i(t))/(1 - x_{B+1}(t)))$.

The set of ODEs $J(x(t))$ has a unique fixed point $\tilde{\pi}$. We derive its closed form further on, expressing the overall probe rate first.

We assume probes are sent sequentially, and a task is migrated to the first discovered eligible node. So at least one probe is sent, and another probe follows if all previous probes failed to locate a lightly loaded node. This results in an average of $1 + \sum_{i=1}^{L_p - 1} \pi_{B+1}^i$ probes sent. Since probes are sent for each arrival (with rate $\lambda$) at a queue of length $T$ or more ($\pi_T$), the resulting overall probe rate equals

$$R_{trad.push} = \lambda \tilde{\pi}_T \frac{1 - \tilde{\pi}_{B+1}^{L_p}}{1 - \tilde{\pi}_{B+1}}. \tag{5.19}$$

Having expressed the overall probe rate, the fixed point is given in the next theorem.

**Theorem 5.6.** *The set of ODEs given by (5.16-5.18) has a unique fixed point $\tilde{\pi} = (\tilde{\pi}_1, \tilde{\pi}_2, \ldots)$ with $\sum_{i \geq 1} \tilde{\pi}_i < \infty$. Let $\tilde{\eta}_i = \tilde{\pi}_i - \tilde{\pi}_{i+1}$ and $\tilde{\eta}_0 = 1 - \lambda$, then we express the fixed point as*

$$\tilde{\eta}_i = (1 - \lambda)(\lambda + R_{trad.push})^i, \qquad 1 \leq i \leq B+1 \tag{5.20}$$

$$\tilde{\eta}_i = \tilde{\eta}_{B+1}\lambda^{i-(B+1)}, \qquad B+2 \leq i \leq T \tag{5.21}$$

$$\tilde{\eta}_i = \tilde{\eta}_T(\lambda \tilde{\pi}_{B+1}^{L_p})^{i-T}, \qquad i > T. \tag{5.22}$$

*Proof.* As it is required that $\sum_{i \geq 1} \tilde{\pi}_i < \infty$, it is clear that $\tilde{\pi}_1 = \lambda$. Then the expressions for $\tilde{\eta}_i$ are obtained from Equations (5.16-5.18) by setting $dx_i(t)/dt = 0$. □

Instead of providing an explicit formula for the mean delay, we show the following equivalence.

**Theorem 5.7.** *When using the same parameters $B$ and $T$, and matching the $R_{trad.push}$ generated by the traditional push, the fixed rate push strategy has the same fixed point, resulting in an equivalent performance.*

*Proof.* From (5.5-5.7) and (5.20-5.22), it is clear that $\eta_i$ and $\tilde{\eta}_i$ are identical for $i \leq T$ as $R_{trad.push} = r\pi_{T+1}$. What remains to be shown is that

$$\lambda \tilde{\pi}_{B+1}^{L_p} = \frac{\lambda}{1 + r_{push}(1 - \pi_{B+1})}.$$

First we rewrite to

$$\tilde{\pi}_{B+1}^{L_p}(1 + r_{push}(1 - \pi_{B+1})) = 1 \tag{5.23}$$

Since we assume both strategies match the same $R$, we use relations (5.19) and (5.8) to establish that $R_{trad.push} = R_{push}$ or

$$\lambda \tilde{\pi}_T \frac{1 - \tilde{\pi}_{B+1}^{L_p}}{1 - \tilde{\pi}_{B+1}} = r_{push}\pi_{T+1}.$$

We use this equality to derive

$$\tilde{\pi}_{B+1}^{L_p} = 1 - \frac{r_{push}\pi_{T+1}(1 - \tilde{\pi}_{B+1})}{\tilde{\pi}_T \lambda}.$$

Substituting this $\tilde{\pi}_{B+1}^{L_p}$ in (5.23) we find that we have an equality of the form $(1-a)(1+b) = 1$ which can be written as $a = b/(1+b)$, this yields

$$\frac{r_{push}\pi_{T+1}(1 - \tilde{\pi}_{B+1})}{\lambda \tilde{\pi}_T} = \frac{r_{push}(1 - \pi_{B+1})}{1 + r_{push}(1 - \pi_{B+1})}.$$

As $\tilde{\pi}_{B+1} = \pi_{B+1}$, dividing by $r_{push}(1 - \pi_{B+1})$ results in

$$\frac{\pi_{T+1}}{\tilde{\pi}_T} = \frac{\lambda}{1 + r_{push}(1 - \pi_{B+1})}.$$

As $\tilde{\pi}_T = \pi_T$, and denoting $\lambda/(1 + r_{push}(1 - \pi_{B+1}))$ as $\tau$, then we have

$$\begin{aligned}\frac{\pi_{T+1}}{\pi_T} &= \frac{\sum_{j=T+1}^{\infty} \eta_j}{\eta_T + \sum_{j=T+1}^{\infty} \eta_j} = \frac{1}{\frac{\eta_T}{\sum_{j=T+1}^{\infty} \eta_j} + 1} = \frac{1}{\frac{1}{\sum_{j=T+1}^{\infty} \tau^{j-T}} + 1} \\ &= \frac{1}{\frac{1}{\frac{1}{1-\tau} - 1} + 1} = \frac{1}{\frac{1-\tau}{\tau} + 1} = \frac{\lambda}{1 + r_{push}(1 - \pi_{B+1})}.\end{aligned}$$

□

### 5.4.2 Traditional Pull

In the traditional pull variant, whenever a node with queue length at most $B + 1$ has processed a task, it sends out at most $L_p$ probes to locate a highly loaded node. The first node found with a queue length larger than $T$, migrates a task to the probing node. A similar setup was studied in [7] using birth-death models, with the constraint that $T = B + 1$.

The evolution of the traditional pull strategy is modeled by a set of ODEs denoted as $dx(t)/dt = K(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$. As explained below, this set of ODEs can be written as

$$\frac{dx_i}{dt} = \lambda(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t))(1 - x_{T+1}(t))^{L_p}, \tag{5.24}$$

for $1 \leq i \leq B+1$. For $B+2 \leq i \leq T$ we have

$$\frac{dx_i}{dt} = \lambda(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)), \tag{5.25}$$

and for $i > T$ we have

$$\begin{aligned}\frac{dx_i}{dt} &= \lambda(x_{i-1}(t) - x_i(t)) - (x_i(t) - x_{i+1}(t)) \\ &- (x_1(t) - x_{B+2}(t))(1 - (1 - x_{T+1}(t))^{L_p})\frac{x_i(t) - x_{i+1}(t)}{x_{T+1}(t)}.\end{aligned} \tag{5.26}$$

The queue length of nodes with at most $B+1$ tasks only decreases if they fail to find a long queue to migrate a task from, this happens with probability $(1 - x_{T+1})^{L_p}$. The extra negative term in (5.26) indicates migrations to the lightly loaded nodes. For every completion of a queue with length at most $B+1$ (rate $(x_1 - x_{B+2})$), the probes are successful with probability $(1 - (1 - x_{T+1})^{L_p})$, and the probability for discovery of a long queues with length $i$ is uniformly distributed over all long queues $((x_i(t) - x_{i+1}(t))/(x_{T+1}(t)))$.

The set of ODEs $K(x(t))$ has a unique fixed point $\mathring{\pi}$. We first express the overall probe rate, and then describe $\mathring{\pi}$ explicitly.

We assume probes are sent sequentially, and a task is migrated from the first discovered eligible node. Thus, at least one probe is sent, and extra probes follow if all previous attempts were unsuccessful. This results in an average of $1 + \sum_{i=1}^{L_p - 1}(1 - \mathring{\pi}_{T+1})^i$ probes sent. Since probes are sent for each completion at a queue with a length of at most $B+1$, the resulting overall probe rate equals:

$$R_{trad.pull} = (\mathring{\pi}_1 - \mathring{\pi}_{B+2})\frac{1 - (1 - \mathring{\pi}_{T+1})^{L_p}}{\mathring{\pi}_{T+1}}. \tag{5.27}$$

Having expressed the overall probe rate, the fixed point is given in the next theorem.

**Theorem 5.8.** *The set of ODEs given by (5.24-5.26) has a unique fixed point* $\mathring{\pi} = (\mathring{\pi}_1, \mathring{\pi}_2, \ldots)$ *with* $\sum_{i \geq 1} \mathring{\pi}_i < \infty$. *Let* $\mathring{\eta}_i = \mathring{\pi}_i - \mathring{\pi}_{i+1}$ *and* $\mathring{\eta}_0 = 1 - \lambda$, *then the fixed point is given by*

$$\mathring{\eta}_i = (1 - \lambda)\left(\frac{\lambda}{(1 - \mathring{\pi}_{T+1})^{L_p}}\right)^i, \qquad 1 \leq i \leq B+1 \tag{5.28}$$

$$\mathring{\eta}_i = \mathring{\eta}_{B+1}\lambda^{i-(B+1)}, \qquad B+2 \leq i \leq T \tag{5.29}$$

$$\mathring{\eta}_i = \mathring{\eta}_T\left(\frac{\lambda}{1 + R_{trad.pull}}\right)^{i-T}, \qquad i > T. \tag{5.30}$$

*Proof.* As $\sum_{i\geq 1} \mathring{\pi}_i < \infty$ implies $\mathring{\pi}_1 = \lambda$, setting $dx_i(t)/dt = 0$ in Equations (5.24-5.24) and rearranging terms results in the expressions for $\mathring{\eta}_i$. □

Instead of providing an explicit formula for the mean delay, we show the following equivalence.

**Theorem 5.9.** *When using the same parameters $B$ and $T$, and matching the $R_{trad.pull}$ generated by the traditional pull, the fixed rate pull strategy has the same fixed point distribution, resulting in an equivalent performance.*

*Proof.* From (5.5-5.7) and (5.28-5.30), it is clear that $\eta_i$ and $\mathring{\eta}_i$ are identical iff

$$\frac{\lambda}{(1-\mathring{\pi}_{T+1})^{L_p}} = \lambda + r_{pull}\pi_{T+1}, \tag{5.31}$$

as $R_{trad.pull} = r(1-\pi_{B+1})$. Since we assume both strategies match the same $R$, we use relations (5.27) and (5.9) to establish that $R_{trad.pull} = R_{pull}$ or

$$(\lambda - \mathring{\pi}_{B+2})\frac{1-(1-\mathring{\pi}_{T+1})^{L_p}}{\mathring{\pi}_{T+1}} = r_{pull}(1-\pi_{B+1}).$$

From this relation we find $r_{pull}$ and substitute in (5.31), yielding

$$\frac{\lambda}{(1-\mathring{\pi}_{T+1})^{L_p}} = \lambda + \frac{\lambda - \mathring{\pi}_{B+2}}{1-\pi_{B+1}}\left(1-(1-\mathring{\pi}_{T+1})^{L_p}\right).$$

This expression is equivalent to

$$\frac{\lambda - \mathring{\pi}_{B+2}}{1-\pi_{B+1}} = \frac{\lambda}{(1-\mathring{\pi}_{T+1})^{L_p}}. \tag{5.32}$$

From (5.28) we find

$$\mathring{\pi}_{B+2} = \lambda - (1-\mathring{\pi}_{B+1})\left(\frac{\lambda}{(1-\mathring{\pi}_{T+1})^{L_p}}\right).$$

We rewrite this relation as

$$\frac{\lambda - \mathring{\pi}_{B+2}}{1-\mathring{\pi}_{B+1}} = \frac{\lambda}{(1-\mathring{\pi}_{T+1})^{L_p}}.$$

As $\mathring{\pi}_{B+1} = \pi_{B+1}$, (5.32) follows. □

## 5.5 d-Choices Strategies

In this section we study variants of the d-choices strategy. The original strategy was introduced in [4], where an infinite system model was used to describe its behavior. Let $x(t) = (x_1(t), x_2(t), \ldots)$, where $x_i(t)$ represents the fraction of nodes with at least $i$ jobs at time $t$. Then the evolution of queue lengths under the d-choices strategy is formulated as the following set of ODEs denoted as $dx(t)/dt = L(x(t))$:

$$\frac{dx_i(t)}{dt} = \lambda(x_{i-1}(t)^d - x_i(t)^d) - (x_i(t) - x_{i+1}(t)). \tag{5.33}$$

Results in [4] show that all trajectories converge to a unique fixed point

$$\bar{\pi}_i = \lambda^{\frac{d^i-1}{d-1}}. \tag{5.34}$$

As explained further on an equivalent distributed variant requires fewer than $d$ probes per task. Additionally, we construct equivalent rate-based variants that send either single probes or batches of probes periodically instead of on task arrival instants.

### 5.5.1 Distributed d-Choices

The original d-choices as introduced in [4] assumes that a central dispatcher sends $d$ probes for every task arrival. When assuming a central dispatcher, other approaches are known to perform better with less probes [13]. We assume that tasks originate at the nodes themselves.

In a sense this setup provides the information of exactly one probe message, that is the queue length of the queue where the task arrives. Therefore, an equivalent strategy to a central dispatcher sending $d$ probes is to let the nodes send $d-1$ probes on a task arrival instant. The task is then forwarded to the least loaded probed node, or stays at the originating node if no shorter queue is found.

The evolution of the distributed d-choices strategy is given by a set of ODEs denoted as $dx(t)/dt = M(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$. As explained below, this set of ODEs can be written as

$$\begin{aligned}\frac{dx_i(t)}{dt} =& \lambda(x_{i-1}(t) - x_i(t))x_{i-1}(t)^{d-1} - (x_i(t) - x_{i+1}(t)) \\ &+ \lambda x_i(t)(x_{i-1}(t)^{d-1} - x_i(t)^{d-1}),\end{aligned} \tag{5.35}$$

for $i > 0$, with $x_0(t) = 1$. Queues of length $i$ are created by arrivals in a queue with length $i-1$ ($\lambda(x_{i-1}(t) - x_i(t))$), only if $d-1$ probes could not find a shorter queue (probability $x_{i-1}^{d-1}(t)$). Additionally, queues of length $i$ are created if an arrival at a queue with length at least $i$ ($\lambda x_i(t)$), sends $d-1$ probes and finds a queue with length $i-1$ the shortest (probability $x_{i-1}^{d-1}(t) - x_i^{d-1}(t)$)

Algebraic manipulation on (5.35) immediately shows the equivalence with the original formulation of the d-choices strategy in (5.33).

Using fixed point from (5.34) we can formulate the mean delay in terms of migrations in the next theorem.

**Theorem 5.10.** *The mean delay of both the distributed and centralized d-choices strategy can be formulated as*

$$\frac{1}{1-\lambda}(1 - \frac{\alpha}{\lambda}),$$

*with*

$$\alpha = \lambda \sum_{i=1}^{\infty} (\bar{\pi}_i - \bar{\pi}_{i+1}) \sum_{j=0}^{i-1} (\bar{\pi}_j^{d-1} - \bar{\pi}_{j+1}^{d-1})(i-j).$$

*Proof.* The improvement over the mean delay of an M/M/1 queue can be formulated as the average number of places a task will skip in the queue due to a migration. Here, for every arrival ($\lambda$) at a queue of length $i$ ($\bar{\pi}_i - \bar{\pi}_{i+1}$), the $d-1$ probes could find a shorter queue. The shortest queue found is of length $j$ with probability ($\bar{\pi}_j^{d-1} - \bar{\pi}_{j+1}^{d-1}$), in which case the task skips $(i-j)$ places. □

Although there is an infinite sum in $\alpha$ of the above theorem, the terms quickly become small as $\bar{\pi}$ decreases doubly exponentially.

We note that the required overall request rate of the distributed d-choices can be lower than $\lambda(d-1)$. First, if a task originates at an empty server, no probes need to be send as no shorter queue can be found. Similarly, the $d-1$ probes could be sent sequentially and stop once an empty server is found. Thus only servers with at least one job need to send probes at task arrival instants until either an empty server is found or the maximum of $d-1$ probes is reached. Analytically, this results in an overall probe rate of

$$R_{dChoices} = \bar{\pi}_1 \lambda \left(1 + \sum_{i=1}^{d-2} \bar{\pi}_1^i\right),$$

where $\bar{\pi}_1\lambda$ is the rate of probe events (arrivals at busy servers), and $(1 + \sum_{i=1}^{d-1} \bar{\pi}_1)$ is the number of probes per event. At least one probe is sent, and a next probe follows if all previous probes found busy servers, up to a maximum of $d-1$ probes in total. Since $\bar{\pi}_1$ equals $\lambda$ we can simplify the expression to

$$R_{dChoices} = \frac{\lambda^2(1-\lambda^{d-1})}{1-\lambda}. \tag{5.36}$$

We will match this probe rate in the following sections to create equivalent strategies.

### 5.5.2 Rate-based Variant Sending Probes in Batch

Instead of sending out $d-1$ probes at task arrival instants, we can adapt the strategy to send batches of probes according to a Poisson process with rate $r$. We will call the sending of a batch of probes a probe event. It is our aim to find a strategy equivalent to the d-choices strategy, i.e. one that achieves the same stationary distribution when using the same overall probe rate.

The first attempt at finding such a strategy lets queues with two or more jobs send out batches of probes periodically with a rate $r$ that is independent of the queue's length. The evolution of such a strategy is modeled by the set of ODEs denoted as $dx(t)/dt = N(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$. As explained below, this set of ODEs can be written as

$$\frac{dx_1}{dt} = \lambda(1 - x_1(t)) + r x_2(t)(1 - x_1(t)^{d-1}) - (x_1(t) - x_2(t)), \tag{5.37}$$

and for $i \geq 2$ we have

$$\begin{aligned}\frac{dx_i}{dt} =& \lambda(x_{i-1}(t) - x_i(t)) + r x_{i+1}(t)(x_{i-1}(t)^{d-1} - x_i(t)^{d-1}) \\ & - (x_i(t) - x_{i+1}(t))(1 + r(1 - x_{i-1}(t)^{d-1})).\end{aligned} \tag{5.38}$$

Queues with length one are created by new arrivals and probes to an empty server. Tasks from all queues with tasks waiting ($x_2(t)$) are eligible for transfer to an empty server, and those queues generate probe events with rate $r$. An empty server is located by $d-1$ probes with probability $(1-x_1(t)^{d-1})$. In general, queues of length $i$ are created when a probe event of a queue with length at least $i+1$ identifies a queue with length $i-1$ as shortest among the $d-1$ probed servers. Likewise, the fraction of queues with length $i$ decreases if the probe events (which occur at rate $r$) locate a queue with length lower than $i-1$ (with probability $(1-x_{i-1}(t)^{d-1})$).

From (5.36) we note that the rate of probe events must be $\lambda^2$, as you send $(1-\lambda^{d-1})/(1-\lambda)$ probes on average per event. In the system above, all servers with tasks waiting generate probe events at the same rate. Therefore, in order for the system to be equivalent with d-choices, we have the condition $r\bar{\pi}_2 = \lambda^2$. In other words, $r$ would need to be $1/\lambda^{d-1}$. Unfortunately, when using this $r$ in conditions (5.38) and setting $dx_i(t)/dt = 0$, $\bar{\pi}$ is not a solution to the resulting set of equations. In other words, it is impossible to create such a strategy that has the same fixed point as the d-choices strategy.

However, when we let each queue send at a rate $r_i$ depending on its length $i$, we can find a strategy equivalent with d-choices by choosing $r_i$ appropriately. We call this strategy push-d-batch. The evolution of such a strategy is modeled by the set of ODEs denoted as $dx(t)/dt = P(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$. As explained below, this set of ODEs can be written as

$$\begin{aligned}\frac{dx_1}{dt} =&\lambda(1-x_1(t)) - (x_1(t)-x_2(t)) \\ &+ (1-x_1(t)^{d-1})\sum_{j=2}^{\infty} r_j(x_j(t)-x_{j+1}(t)),\end{aligned} \tag{5.39}$$

and for $i \geq 2$ we have

$$\begin{aligned}\frac{dx_i}{dt} =&\lambda(x_{i-1}(t)-x_i(t)) \\ &- (x_i(t)-x_{i+1}(t))(1+r_i(1-x_{i-1}(t)^{d-1})) \\ &+ (x_{i-1}(t)^{d-1} - x_i(t)^{d-1})\sum_{j=i+1}^{\infty} r_j(x_j(t)-x_{j+1}(t)).\end{aligned} \tag{5.40}$$

The same remarks as for $N(x(t))$ apply. The difference here is that queues with length $i$ generates probe events with rate $r_i$, so we now have to sum the $r_i$ over the queue lengths: $\sum_{j=i+1}^{\infty} r_j(x_j(t)-x_{j+1}(t))$.

We aim to achieve the same stationary distribution as d-choice, so we will use $\bar{\pi}$ from (5.34) to denote the fixed point. When substituting $x_i$ with $\bar{\pi}_i$ in (5.39), the expression reduces to zero as required. We also aim to use the same rate of probe events, therefore

$$\sum_{j=2}^{\infty} r_j(\bar{\pi}_j(t) - \bar{\pi}_{j+1}(t)) = \lambda^2.$$

Achieving both objectives is accomplished by the choice of $r_i$. As we know the fixed point of the d-choices strategy ($\bar{\pi}$), we can find $r_i$ from $dx_i/dt$ in (5.40) by rewriting

the sum term as the known total sum ($\lambda^2$) minus the missing terms. For example we find $r_2$ from

$$\begin{aligned}\frac{dx_2}{dt} = 0 =&\lambda(\bar{\pi}_1 - \bar{\pi}_2) - (\bar{\pi}_2 - \bar{\pi}_3)(1 + r_2(1 - \bar{\pi}_1^{d-1})) \\ &+ (\bar{\pi}_1^{d-1} - \bar{\pi}_2^{d-1})(\lambda^2 - r_2(\bar{\pi}_2 - \bar{\pi}_3)),\end{aligned}$$

where all terms are known except $r_2$. Repeating this procedure for $i \geq 2$ we find the general expression

$$r_{i|batch} = \frac{-\lambda(1 - \lambda^{d^i - 1})}{(1 - \lambda^{d^i})(1 - \bar{\pi}_i^{d-1})} - \frac{1 - \lambda^{1-d^{i-1}}}{(1 - \lambda^{d^i})(1 - \bar{\pi}_{i-1}^{d-1})}.$$

By allowing queues to generate probe events at a rate dependent on the queue length, we have shown that a rate-based variant equivalent to d-choices can be constructed for which probe events need not be at task arrival instants. In this formulation probes are still sent in batch, and therefore this strategy is not a member of the class $\mathcal{S}(\mathbf{r}, A)$. In the next section we construct an equivalent rate-based variant where probe events consist of a single probe, thus belonging to the class $\mathcal{S}(\mathbf{r}, A)$.

### 5.5.3 Rate-based Variant Sending Single Probes

In the previous section we showed that generating probe events as a Poisson process can be just as effective as sending probes at arrival instants. In this section we demonstrate that sending probes in batch is also not required to achieve the same stationary distribution as d-choice.

Again our aim is to construct a strategy with an equivalent performance compared to d-choice, while using the same number of probes. Now a probe event consists of sending a single probe. A migration is initiated if the probe finds a queue of at least two tasks shorter, so all transfers lower the mean queue length but tasks can be migrated multiple times. Each queue with length $i$ generates probe events at rate $r_i$, and the overall probe rate is equal to (5.36). We will call the strategy described here push-d-single.

Formally this strategy is a member of the class $\mathcal{S}(\mathbf{r}, A)$ and is defined as follows. The elements $a_{i,j}$ are one if $i > j + 1$. The explicit values for $r_i$ are introduced further on.

The evolution of push-d-single is modeled by the set of ODEs denoted as $dx(t)/dt = Q(x(t))$, where $x(t) = (x_1(t), x_2(t), \ldots)$ and $x_i(t)$ represents the fraction of the number of nodes with at least $i$ jobs at time $t$. This is a simplified version of Equation (5.1), and the ODEs can be written as

$$\frac{dx_1}{dt} = \lambda(1 - x_1(t)) - (x_1(t) - x_2(t)) + (1 - x_1(t)) \sum_{j=2}^{\infty} r_j(x_j(t) - x_{j+1}(t)), \quad (5.41)$$

and for $i \geq 2$ we have

$$\begin{aligned}\frac{dx_i}{dt} =&\lambda(x_{i-1}(t) - x_i(t)) \\ &- (x_i(t) - x_{i+1}(t))(1 + r_i(1 - x_{i-1}(t))) \\ &+ (x_{i-1}(t) - x_i(t)) \sum_{j=i+1}^{\infty} r_j(x_j(t) - x_{j+1}(t)).\end{aligned} \tag{5.42}$$

When substituting $x_i(t)$ with $\bar{\pi}_i$ of (5.34) in (5.41) and using

$$\sum_{j=2}^{\infty} r_j(x_j(t) - x_{j+1}(t)) = \frac{\lambda^2(1 - \lambda^{d-1})}{1 - \lambda},$$

the expression reduces to zero as required, indicating that this strategy could have the same fixed point as the d-choices strategy. In order to find a suitable $r_i$ we employ the same method as in the previous section, we rewrite the sum $\sum_{j=i+1}^{\infty} r_j(x_j(t) - x_{j+1}(t))$ as the known total ($R_{dChoices}$) minus the missing terms. Then, we find $r_i$ by substituting $\bar{\pi}$ of (5.34) in (5.42) and requiring that $dx_i/dt = 0$. For example $r_2$ is found from

$$\begin{aligned}\frac{dx_2}{dt} = 0 =&\lambda(\bar{\pi}_1 - \bar{\pi}_2) - (\bar{\pi}_2 - \bar{\pi}_3)(1 + r_2(1 - \bar{\pi}_1)) \\ &+ (\bar{\pi}_1 - \bar{\pi}_2)\left(\frac{\lambda^2(1 - \lambda^{d-1})}{1 - \lambda} - r_2(\bar{\pi}_2 - \bar{\pi}_3)\right).\end{aligned}$$

In general, we find that $r_i$ for $i \geq 2$ must be equal to

$$r_{i|single} = \frac{\lambda^{\frac{1}{d-1}}\left(\frac{\left(\lambda^{\frac{d^{i-1}}{d-1}} - \lambda^{\frac{d^i}{d-1}}\right)\left(\lambda^{\frac{d^i}{d-1}} - \lambda^{\frac{d}{d-1}}\right)}{\left(\lambda^{\frac{1}{d-1}} - \lambda^{\frac{d^{i-1}}{d-1}}\right)\left(\lambda^{\frac{d^{i+1}}{d-1}} - \lambda^{\frac{d^i}{d-1}}\right)} - 1\right)}{\lambda^{\frac{1}{d-1}} - \lambda^{\frac{d^i}{d-1}}},$$

in order for the stationary distribution to match $\bar{\pi}$ while using an overall probe rate of $\sum_{i=2}^{\infty} r_i(\bar{\pi}_i - \bar{\pi}_{i+1}) = R_{dChoices}$.

## 5.6 Performance Evaluation

As we know the overall probe rate of the distributed d-choices strategy from (5.36), we can compare the considered strategies fairly. That is to say, we compare the mean delay given that all strategies use the same overall probe rate. We choose to compare the d-choices strategy due to its popularity, and compare it with the strategies of the class $\mathcal{S}(\mathbf{r}, A)$ that we expect to be optimal as indicated in Conjectures 5.1 and 5.2.

We let the d-choices strategy determine the overall probe rate, and make sure the max-push and fixed rate pull strategy match this rate by setting $T, B$ and $r$ appropriately. Figures 5.5 and 5.6 summarize the performance comparison.

The fixed rate pull strategy is clearly superior for high loads. Also notable is that its mean delay stays finite as the load $\lambda$ tends to one, specifically the delay

approaches $d/(d-1)$ with $R = R_{dChoices}$. However, the probe rate will be infinite in this case, as it is given by $r = R/(1-\lambda)$. For lower loads the pull strategy is not optimal, but adopting this strategy independently of the system load might be an option as the performance is still reasonable and the simplicity of not having to switch strategies depending on the system load keeps the implementation straightforward. Furthermore, the only parameter that would have to be adjusted at runtime depending on the system load is the probe rate, as we conjecture that setting $T = 1$ and $B = 0$ is optimal.

The mean delay of the d-choices and max-push strategy are almost identical for low loads, with the max-push achieving a slightly lower mean delay. This region extends to medium loads as $d$ increases. For higher loads the max-push strategy only slightly outperforms d-choices. This close match in mean delay is notable because we conjecture that max-push is the optimal push strategy within the class $\mathcal{S}(\mathbf{r}, A)$. This suggests that d-choices achieves a close to optimal result with a far simpler approach. The only parameter d-choices has to select is $d$, whereas the max-push has to adjust $B, T$ and $r$ depending on the system load. Furthermore, the assumption that a node can probe at an infinite rate might not hold, and will in practice be replaced by some high but finite rate. Moreover, in a setting with a finite number of servers it can occur that all queues are temporarily longer than $B$ and thus no transfers can be made, yet new arrivals at queues with length $T$ expect an immediate transfer. In addition, sending a batch of probes might be preferable if the latency is non-negligible in order to avoid waiting for the results of multiple sequential probes. In conclusion, d-choices is far more practical than max-push and still achieves a comparable performance.

To better understand why the performance of the d-choices and max-push strategies is so similar for low to moderate loads, we show in Table 5.5 several probe rates $r_i$ used by push-d-single. Clearly $r_i$ increases with $i$ and $d$, but decreases with $\lambda$. The increase with $i$ is fast, so that for low loads the push-d-single and the max-push behave almost the same. They both require that queues with length at least $i$ send probes at a practically infinite rate, and that most of the remaining probes are send by the queues with length $i-1$.

| $(\lambda, d)$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ |
|---|---|---|---|---|
| $(0.5, 2)$ | 1.6 | 7.53 | 1.28e+2 | 3.28e+4 |
| $(0.5, 4)$ | 1.35e+1 | 3.38e+4 | 9.22e+18 | 5.79e+76 |
| $(0.75, 2)$ | 8.53e−1 | 1.8 | 6.81 | 7.41e+1 |
| $(0.75, 4)$ | 4.56 | 9.61e+1 | 7.45e+7 | 7.23e+31 |
| $(0.95, 2)$ | 5.53e−1 | 6.43e−1 | 8.61e−1 | 1.5 |
| $(0.95, 4)$ | 1.92 | 3.88 | 3.59e+1 | 4.85e+5 |

Table 5.5: The first probe rates $r_i$ of push-d-single. We note that $r_i$ increases rapidly with $i$ and $d$, and decreases with $\lambda$.

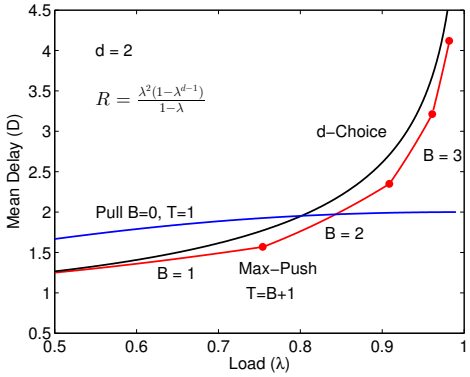


Figure 5.5: Mean delay of the d-choices with $d = 2$, max-push and pull strategy. All strategies produce the same overall probe rate $R$.

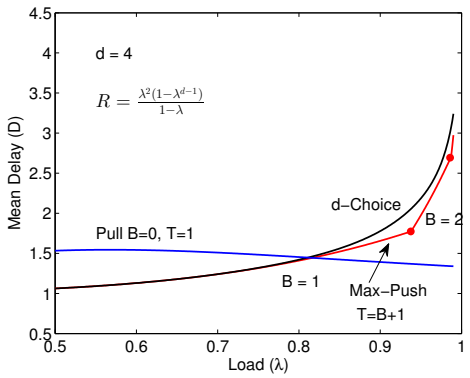


Figure 5.6: Mean delay of the d-choices with $d = 4$, max-push and pull strategy. All strategies produce the same overall probe rate $R$.

## 5.7 Conclusion

In this chapter we have studied several load balancing strategies. We introduced an infinite system model of a general push and pull framework, and indicated the strategies that we expect to be optimal for this class.

We have extended the infinite system model for the fixed rate push and pull, max-push, and the traditional push and pull, to include the parameter $B$ describing the maximal queue length of a lightly loaded server. For a push strategy increasing this $B$ can lead to better performance, whereas for a pull strategy setting $B = 0$ appears best. In addition, we have shown that traditional and fixed rate strategies are equivalent if both use the same overall probe rate $R$.

Furthermore, we have revisited the popular d-choices strategy and have shown that the required overall probe rate is smaller than $d$ probes per task, specifically $\lambda^2(1-\lambda^{d-1})/(1-\lambda)$. In the original formulation probes are sent in batch and on task arrival instants. We have shown that equivalent rate-based push strategies exist that send either single probes or a batch of probes periodically according to a Poisson process with rate $r_i$ dependent on the queue length $i$.

Finally, we compared the performance of the best performing rate-based push and pull strategy with d-choices, given that the same overall probe rate is used. The pull strategy is the best choice for high loads, but its simplicity and reasonable performance for low to moderate loads makes it a viable solution in case the system must use a single strategy. For low loads the max-push and d-choices performance is nearly equivalent, with the max-push achieving a slightly lower mean delay for medium to high loads. Still, it is remarkable that the simple d-choices strategy performs so close to the more complicated max-push which we conjecture to be an optimal push strategy.

Chapter 6

# Conclusions

In this thesis we introduced and explored a class of load balancing strategies, with a focus on comparing the performance in terms of mean delay given that strategies produce the same overall probe rate. We also showed that the infinite system model is the correct limiting process as the number of nodes tends to infinity, which required resolving some technical issues that result from allowing infinite buffers. In addition, we numerically assessed the relative error in mean delay and overall probe rate when using an infinite system model to predict the performance of a finite system.

## 6.1 Main Contributions

We introduce a simple rate-based push and pull strategy in Chapter 2 where probes are sent periodically instead of at arrival or completion times, as is typically the case in related work. Our approach has the advantage that we have fine-grained control over the generated overall probe rate, whereas the traditional strategies are more limited in that regard. This allows us to compare the pull and push strategies given that they produce the same overall probe rate. Moreover, we show that when the rate-based strategies match the overall probe rate produced by the traditional strategies, the resulting performance is equivalent. In addition, we show that a pure pull or push strategy is always superior to a hybrid approach. We start from a finite system model and let the system size tend to infinity, obtaining the correct limit process. The resulting infinite system model consists of a set of ODEs. We proceed to show these ODEs have a unique fixed point that is a global attractor. Furthermore, we provide closed form results for the stationary queue length distribution and mean delay. The accuracy of using the infinity system model to approximate a finite system is numerically validated by means of simulation.

In Chapter 3 we consider a queue *long* if its length exceeded $T$, instead of a queue being considered long if it has any number of tasks waiting. Then we examine the effect of letting only long queues send probes (push) or requiring an empty server to find a long queue to transfer a task from (pull). For the pull strategy we show it is optimal to set $T = 1$, whereas for the push strategy it is beneficial to increase

$T$ as the load increases. We also observe that the push strategy could not always use the full overall probe rate budget. When $R$ is relatively large, long queues can send probes at an infinite rate without exceeding the overall probe limit, effectively eliminating queues longer than $T$. We therefore introduce the max-push strategy which allows sending probes at an infinite rate for queues with length $T + 1$, and uses the remaining probe budget for letting queues with length $T$ send probes periodically. This max-push strategy outperforms the rate-based push strategy for all loads. The theoretical results from Chapter 2 are extended to be applicable to the models introduced in Chapter 3, and numerical simulations validate the accuracy of the infinite system models when approximating finite systems.

The effect of limiting the migration rate $M$ is analyzed in Chapter 4. In this case a strategy has to satisfy both the overall probe rate and the overall migration rate constraint. We provide policies to choose the strategies' parameters in order to achieve this. Furthermore, we observe that setting $T = 1$ is no longer optimal for the pull strategy in this case. When the migration rate is limited, it can be beneficial to only pull tasks from longer queues. To further improve the performance of the pull strategy, we introduce the conditional-pull strategy. Under this strategy a task is always pulled if the target's queue length exceeds $T$, and pulled with probability $p$ if its queue length equals $T$. The conditional-pull strategy outperforms the rate-based pull strategy for all loads, if the overall migration rate is limited. Theoretical results from Chapter 3 also hold for the newly introduced strategy, and we show by simulation that the infinite system model is a good approximation for finite systems.

The model of push and pull strategies is further extended in Chapter 5 to a general class of strategies. The model allows that the probe rate of a queue depends on its length, but we mostly look at strategies where all queues probe at a fixed rate independent of the queue length. In addition, the model can allow transfers between queues of arbitrary length, but we focus on strategies where the only transfers are from long to short queues. We consider a queue *short* if its length is at most $B$. Increasing $B$ is beneficial for the push strategy as the load increases, as short queues are easier to locate than empty servers and such a transfer still lower the mean delay. For the pull strategy increasing $B$ does not improve performance. When comparing the performance of these fixed-rate push and pull strategies with their traditional counterparts where probes are sent in batch, we again find that the performance is equivalent if the rate-based strategies match the overall probe rate produced by the traditional strategies. We also extend the max-push strategy to allow transfers to short queues. For these strategies we assess the accuracy of the infinite system model as an approximation for a finite system by means of simulation. In addition, we develop distributed versions of the popular d-choices strategy that do not rely on a central dispatcher. In particular, we show that a rate-based strategy can be constructed with a probe rate dependent on the queue length, sending either single probes or in batch, that is equivalent in performance with d-choices if the same overall probe rate is used. We also compared the strategies that we conjecture to be optimal within the proposed framework (max-push with $T = B + 1$ and pull with $B = 0$ and $T = 1$) with d-choices, given that all strategies use the same overall probe rate. We find that d-choices performs remarkably well given its low complexity, compared to the more complex max-push strategy. For high loads, the pull strategy remains best.

## 6.2 Future Work

Throughout this thesis we make several assumptions that simplify the analysis, and this allows us to find closed form expressions for many quantities of interest. However, these assumptions are not always realistic in practice, and it would be interesting to see how the results change when those assumptions are relaxed. Although it is expected that some quantities might not be expressible as closed forms anymore when relaxing the assumptions.

We use an infinite system model, and check by simulation how accurate it predicts the performance of a finite system. One could also model such finite system directly, and derive performance measures explicitly. Some technical issues we encountered would be less involved when dealing with a finite system.

A simple extension would be to allow the migration of multiple tasks when an eligible queue is found, instead of only one task transfer per probe. This could increase efficiency, using less probes to achieve the same level of performance. In [9, 22] such an approach is briefly explored, but this could be extended by taking into account the overall probe rate and migration rate.

In our models we assume that probes and transfers do not incur a computational cost, and that they are resolved instantaneously. This assumption can be justified if tasks take a relatively long time to complete, so the other delays are small enough in comparison to ignore them. However, for relatively short tasks the impact of adding a delay or computational cost to either transfers or probes will be significant, and is worth looking into. For related work in this direction we refer to [22, 39, 40].

Another interesting generalization would be phase-type arrival and service times, instead of the Poisson arrival time and exponential service time we use in this thesis. One could add dimensions to the ordinary differential equations to keep track of the phase-type. Then the ODEs can be simulated numerically to find the mean delay. However proving global attraction will probably be more challenging, and closed forms for a fixed point or mean delay will be hard to obtain. Such generalization is considered in [22] and treated thoroughly in [41].

We studied a homogeneous system of nodes, where all queues are connected and serve tasks at the same rate. One could also study the load balancing problem in heterogeneous networks, where nodes can be connected with a specific topology and the service rate can vary per server class, in time, or depending on the load. Several other authors have considered this setting, including [9, 19, 20],

Appendix A

# Proof of Theorem 3.3

We start by proving the following Lemma:

**Lemma A.1.** *Let $x(t)$ be the unique solution of the set of ODEs given by* (3.1) *to* (3.3) *with $x(0) \in E$. The $L_1$-distance to the unique fixed point $\sum_{i\geq 1} |x_i(t) - \bar{\pi}_i|$ does not increase as a function of $t$.*

*Proof.* Define $\epsilon_i(t) = x_i(t) - \bar{\pi}_i$, for $i \geq 1$, such that $\Phi(t) = \sum_{i\geq 1} |\epsilon_i(t)|$ represents the $L_1$-distance. As $\frac{d}{dt}x_i(t) = \frac{d}{dt}\epsilon_i(t)$ and $\bar{\pi}$ is a fixed point of (3.1) to (3.3), we find

$$\frac{d}{dt}\epsilon_1(t) = -\epsilon_1(t)(1+\lambda) - r\epsilon_1(t)(\bar{\pi}_{T+1} + \epsilon_{T+1}(t)) + r\epsilon_{T+1}(t)(1-\bar{\pi}_1) + \epsilon_2(t), \tag{A.1}$$

$$\frac{d}{dt}\epsilon_i(t) = \lambda\epsilon_{i-1}(t) - (1+\lambda)\epsilon_i(t) + \epsilon_{i+1}(t), \tag{A.2}$$

for $2 \leq i \leq T$, and

$$\begin{aligned}\frac{d}{dt}\epsilon_i(t) = {} & \lambda(\epsilon_{i-1}(t) - \epsilon_i(t)) + r\epsilon_1(t)(\bar{\pi}_i - \bar{\pi}_{i+1}) \\ & - (\epsilon_i(t) - \epsilon_{i+1}(t))(1 - r\epsilon_1(t) + r(1-\bar{\pi}_1)),\end{aligned} \tag{A.3}$$

for $i > T$. Assume for now that $\epsilon_i(t) \neq 0$ for all $i$ such that $\frac{d}{dt}\Phi(t)$ is properly defined as

$$\frac{d}{dt}\Phi(t) = \sum_{i:\epsilon_i(t)>0} \frac{d}{dt}\epsilon_i(t) - \sum_{i:\epsilon_i(t)<0} \frac{d}{dt}\epsilon_i(t).$$

If $\epsilon_i(t)$ has the same sign for all $i$, one finds that $\frac{d}{dt}\Phi(t) = -|\epsilon_1(t)|$ by summing (A.1) to (A.3), we will show that this inequality also holds in general. Let $I = \{i_1, i_2, \ldots\}$, with $i_1 < i_2 < \ldots$, be the set of indices where $\epsilon_i(t)$ changes sign, that is, $\epsilon_{i-1}(t)$ and $\epsilon_i(t)$ have a different sign if and only if $i \in I$. Assume $\epsilon_1(t) < 0$ and let $I_k = \{i \in I : i \leq T+1\}$, $I_m = \{i \in I : i > T+1\}$,

By means of (A.1) to (A.3), we find that if $\epsilon_{i-1}(t)$ and $\epsilon_i(t)$ differ in sign, $\frac{d}{dt}\Phi(t)$ contains an extra term given by

$$sign(\epsilon_i(t))2(\lambda\epsilon_{i-1}(t) - \epsilon_i(t)),$$

for $i = 2, \ldots, T+1$,

$$sign(\epsilon_i(t))2[\lambda\epsilon_{i-1}(t) + r\epsilon_1(t)\bar{\pi}_i - \epsilon_i(t)(1 - r\epsilon_1(t) + r(1-\bar{\pi}_1))],$$

for $i > T+1$. Further, if $\epsilon_1(t)$ and $\epsilon_{T+1}(t)$ differ in sign, $\frac{d}{dt}\Phi(t)$ contains an extra term given by

$$sign(\epsilon_{T+1}(t))2r[\epsilon_1(t)(\bar{\pi}_{T+1} + \epsilon_{T+1}(t)) - \epsilon_{T+1}(t)(1-\bar{\pi}_1)].$$

This implies that for $\epsilon_1(t) \leq 0$

$$\begin{aligned}
\frac{d}{dt}\Phi(t) &= \underbrace{\epsilon_1(t)}_{\leq 0} + \alpha \\
&+ 2\sum_{i\in I_m} \underbrace{sign(\epsilon_i(t))\{\lambda\epsilon_{i-1}(t) - \epsilon_i(t)(1 + r(1-\bar{\pi}_1))\}}_{\leq 0} \\
&+ 2\sum_{i\in I_m} sign(\epsilon_i(t))\{r\epsilon_1(t)(\epsilon_i(t) + \bar{\pi}_i)\} \\
&+ 2\sum_{i\in I_k} \underbrace{sign(\epsilon_i(t))\{\lambda\epsilon_{i-1}(t) - \epsilon_i(t)\}}_{\leq 0}
\end{aligned}$$

where $\alpha$ is equal to

$$2r\epsilon_1(t)(\epsilon_{T+1}(t) + \bar{\pi}_{T+1}) - \underbrace{2\epsilon_{T+1}(t)r(1-\bar{\pi}_1)}_{\leq 0}.$$

if $\epsilon_{T+1}(t) > 0$ and zero otherwise.

Hence, $\frac{d}{dt}\Phi(t) \leq \epsilon_1(t)$ provided that

$$\sum_{i\in I_m} sign(\epsilon_i(t))(\epsilon_i(t) + \bar{\pi}_i) = \sum_{i\in I_m} sign(\epsilon_i(t))x_i(t) \geq 0,$$

if $\epsilon_{T+1}(t) \leq 0$ and

$$x_{T+1}(t) + \sum_{i\in I_m} sign(\epsilon_i(t))x_i(t) \geq 0,$$

if $\epsilon_{T+1}(t) > 0$.

Let $I_m = \{i_0, i_1, \ldots\}$. In case $\epsilon_{T+1}(t) \leq 0$, the $sign(\epsilon_{i_n}(t))$ is equal to 1 for $n$ even and $-1$ for $n$ odd. Hence, the condition reduces to

$$\sum_{k\geq 0}(x_{i_{2k}}(t) - x_{i_{2k+1}}(t)) \geq 0,$$

which holds as $x_i(t) \geq x_j(t)$ for $i < j$. Similarly, if $\epsilon_{T+1}(t) > 0$, the $sign(\epsilon_{i_n}(t))$ is equal to $-1$ for $n$ even and 1 for $n$ odd. Hence, the condition reduces to

$$(x_{T+1}(t) - x_{i_0}(t)) + \sum_{k\geq 0}(x_{i_{2k+1}} - x_{i_{2k+2}}) \geq 0,$$

which again holds as $x_i(t) \geq x_j(t)$ for $i < j$.

Hence, $\frac{d}{dt}\Phi(t) \leq -|\epsilon_1(t)|$ if $\epsilon_1(t) \leq 0$. A similar argument can be used for $\epsilon_1(t) \geq 0$.

Finally, the technical issue of defining $\frac{d}{dt}\Phi(t)$ in case $\epsilon_i(t) = 0$ for some $i$ and $t = t_0$, can be resolved as in Chapter 2. □

The above lemma shows that the $L_1$-distance to the fixed point does not increase along any trajectory $x(t)$ in $E$, and can only remain the same whenever $x_1(t) = \bar{\pi}_1$ and there are no sign changes in the $\epsilon_i(t)$'s.

**Lemma A.2.** *The only trajectory $x(t)$ of the ODEs given by (3.1) to (3.3) with $x(0) \in E$ for which the $L_1$-distance does not decrease is given by $x(t) = \bar{\pi}$ for all $t$.*

*Proof.* From the proof of Lemma A.1, we know that $x_1(t) = \bar{\pi}_1 = \lambda$ for all $t$, whenever the $L_1$-distance does not decrease. Equation (3.1) therefore implies that $x_{T+1}(t) = \frac{\lambda^2 - x_2(t)}{r(1-\lambda)}$ on such a trajectory. Hence, if $x_2(t) = \bar{\pi}_2 + c$, then

$$x_{T+1}(t) = \frac{\lambda^2 - \bar{\pi}_2(t)}{r(1-\lambda)} - \frac{c}{r(1-\lambda)} = \bar{\pi}_{T+1} - \frac{c}{r(1-\lambda)}.$$

Hence, $\epsilon_2(t) = x_2(t) - \bar{\pi}_2$ and $\epsilon_{T+1}(t) = x_{T+1}(t) - \bar{\pi}_{T+1}$ differ in sign unless $x_2(t) = \bar{\pi}_2$ and $x_{T+1}(t) = \bar{\pi}_{T+1}$. The fact that $x(t) = \bar{\pi}$ on such a trajectory now follows from (3.1) to (3.3). □

We now recall La Salle's invariance principle for Banach spaces, where a (positively) invariant subset of $K \subset E$ of an ODE defined on $E$ is such that $x(t) \in K$ for all $t$ provided that $x(t)$ is the unique solution of the ODE with $x(0) \in K$.

**Theorem A.1** ( [28])**.** *Let $V(x)$ be a continuous real valued function from $E$ to $\mathbb{R}$ with $\frac{d}{dt}V(x) = \limsup_{t\to 0^+} \frac{1}{t}(V(x(t)) - V(x)) \leq 0$, where $x(t)$ is the unique solution of an ODE with $x(0) = x$. Let $K = \{x \in E | \frac{d}{dt}V(x) = 0\}$ and let $M$ be the largest (positively) invariant subset of $K$. If $x(t)$ is precompact (i.e., remains in a compact set) for $x(0) \in E$, then*

$$\lim_{t\to\infty} dist(x(t), M) = 0,$$

*where $dist(x, M)$ represents the Banach distance between the point $x \in E$ and the set $M \subset E$.*

Using La Salle's invariance principle, Theorem 3.3 can be proven analogously to Theorem 2.4 with $V(x)$ equal to the $L_1$-distance.

Appendix B

# Proof of Theorem 3.8

We start by proving the following Lemma:

**Lemma B.1.** *Let $x(t)$ be the unique solution of the ODEs given by (3.7) to (3.9) with $x(0) \in E_T$. The $L_1$-distance to the unique fixed point $\sum_{i\geq 1} |x_i(t) - \dot{\pi}_i|$ does not increase as a function of $t$.*

*Proof.* Using the same definitions as in Appendix A, we find

$$\frac{d}{dt}\epsilon_1(t) = \lambda(\epsilon_T(t) - \epsilon_1(t)) - (\epsilon_1(t) - \epsilon_2(t)) + r\epsilon_T(t)(1 - (\epsilon_1(t) + \dot{\pi}_1)) - r\dot{\pi}_T\epsilon_1(t), \tag{B.1}$$

$$\frac{d}{dt}\epsilon_i(t) = \lambda\epsilon_{i-1}(t) - (1+\lambda)\epsilon_i(t) + \epsilon_{i+1}(t), \tag{B.2}$$

for $1 < i < T$, and

$$\frac{d}{dt}\epsilon_T(t) = \lambda(\epsilon_{T-1}(t) - \epsilon_T(t)) + r\epsilon_1(t)\dot{\pi}_T - \epsilon_T(t)(1 + r(1 - (\epsilon_1(t) + \dot{\pi}_1))). \tag{B.3}$$

Assume for now that $\epsilon_i(t) \neq 0$ for all $i$ such that $\frac{d}{dt}\Phi(t)$ is properly defined. If $\epsilon_i(t)$ has the same sign for all $i$, one finds that $\frac{d}{dt}\Phi(t) = -|\epsilon_1(t)|$ by summing (B.1) to (B.3), we will show that this inequality also holds in general. If $\epsilon_i(t)$ and $\epsilon_{i-1}(t)$ differ in sign, $\frac{d}{dt}\Phi(t)$ changes by

$$sign(\epsilon_i(t))2(\lambda\epsilon_{i-1}(t) - \epsilon_i(t)),$$

for $i = 2, \ldots, T$, while a difference in sign between the terms $\epsilon_1(t)$ and $\epsilon_T(t)$ creates a term of the form

$$sign(\epsilon_T(t))2\{\lambda\epsilon_T(t) + r\epsilon_1(t)\dot{\pi}_T - \epsilon_T(t)r(1 - (\epsilon_1(t) + \dot{\pi}_1))\}.$$

Assume $\epsilon_1(t) \leq 0$, then we find

$$\frac{d}{dt}\Phi(t) = \underbrace{\epsilon_1(t)}_{\leq 0} + \alpha + 2\sum_{i\in I}\{\underbrace{sign(\epsilon_i(t))(\lambda\epsilon_{i-1}(t) - \epsilon_i(t))}_{\leq 0}\} \tag{B.4}$$

where $\alpha$ is equal to

$$\alpha = \underbrace{-2\lambda\epsilon_T(t)}_{\leq 0}\underbrace{-2r\epsilon_T(t)(1-(\epsilon_1(t)+\dot{\pi}_1))}_{\leq 0}+\underbrace{2r\dot{\pi}_T\epsilon_1(t)}_{\leq 0}. \tag{B.5}$$

if $\epsilon_T(t) > 0$ and $\alpha = 0$ otherwise.

Hence, $\frac{d}{dt}\Phi(t) \leq \epsilon_1(t)$. A similar argument can be used for $\epsilon_1(t) > 0$ by reversing all the signs.

As in Appendix A, the technical issue of defining $\frac{d}{dt}\Phi(t)$ in case $\epsilon_i(t) = 0$ for some $i$ and $t = t_0$ is resolved by relying on the upper right-hand derivative (as in [4, Theorem 3]). □

The above lemma shows that the $L_1$-distance to the fixed point does not increase along any trajectory $x(t)$ in $E_T$, and only remains the same whenever $x_1(t) = \pi_1$ (as $\epsilon_1(t) = 0$ in such a case).

**Lemma B.2.** *The only trajectory $x(t)$ of the ODE given by (3.7) to (3.9) with $x(0) \in E_T$ for which the $L_1$-distance does not decrease is given by $x(t) = \dot{\pi}$ for all $t$.*

*Proof.* If $x_1(t) = \dot{\pi}_1 = \lambda$ for all $t$, then (3.7) implies that $x_T(t) = \frac{\lambda^2 - x2(t)}{\lambda + r(1-\lambda)}$ and the proof proceeds as in Lemma A.2. □

Using La Salle's invariance principle for Banach spaces as given by Theorem A.1, we can now prove Theorem 3.8:

*Proof of Theorem 3.8.* We rely on La Salle's invariance principle for Banach spaces by setting $V(x)$ equal to the $L_1$-distance to the fixed point, i.e., $V(x) = \sum_{i=1}^{T} |x_i - \dot{\pi}_i|$. Lemma B.1 implies that $\frac{d}{dt}V(x) \leq 0$, while Lemma B.2 shows that $M = \{\dot{\pi}\}$ is a singleton. Hence, $\dot{\pi}$ is a global attractor since $E_T$ itself is a compact set and all trajectory are contained within $E_T$ by definition. □

Appendix C

# Fixed Point Approximation

In this appendix we briefly indicate that the set of ODEs $\frac{d}{dt}x_i(t) = H(x(t))$ defined in Section 4.5 is the proper limit process of a family of density dependent Markov processes and that the stationary measure of these processes converges to the unique fixed point of the set of ODEs.

Let $E_C = \{(x_1, x_2, \ldots) | x_i \in [0,1], x_i \geq x_{i+1}, i \geq 1, \sum_{j\geq 1} x_j < \infty\}$. The next two theorems show that this set of ODEs is Lipschitz on $E_C$ and it has a unique fixed point in $E$.

**Proposition C.1.** *The function $H$ is Lipschitz on $E_C$.*

*Proof.* $H$ is Lipschitz provided that for all $x, y \in E_C$ there exists an $L > 0$ such that $|H(x) - H(y)| \leq L\,|x - y|$. By definition of $H(x)$ one finds

$$|H(x) - H(y)| \leq 2(\lambda + 1 + 2r)\,|x - y| + 2r(1+p)\sum_{i>T} |x_1(x_i - x_{i+1}) - y_1(y_i - y_{i+1})|\,.$$

The above sum can be bounded by

$$\sum_{i>T} |(x_1 - y_1)(x_i - x_{i+1}) + y_1(x_i - x_{i+1} - y_i + y_{i+1})|\,,$$

which is bounded by $2\,|x - y|$ on $E_C$. Hence, $H$ is Lipschitz by letting $L = 2\lambda + 2 + 8r + 8rp$. □

As $E_C$ is a Banach space the Lipschitz condition of $H$ suffices to guarantee that the set of ODEs $\frac{d}{dt}x(t) = H(x(t))$, with $x(0) \in E_C$, has a unique solution[1] $\phi_t(x(0))$ [27, Section 1.1].

The set of ODEs $\frac{d}{dt}x(t) = H(x(t))$ describes the transient evolution of the infinite system, while we are in fact interested in its behavior as $t$ goes to infinity. Thus, we are interested in the limit of all the trajectories of this set of ODEs.

[1] The solution $\phi_t(x)$ belongs to the class of continuously differentiable functions as in the finite dimensional case.

**Theorem C.1.** *All the trajectories of the set of ODEs* $\frac{d}{dt}x(t) = H(x(t))$*, starting from* $x \in E_C$ *converge towards the unique fixed point* $\hat{\pi}$*.*

The proof of this Theorem is omitted, as it analogous to the proof of Theorem 3.3.

Similar as in Chapter 2 for the rate-based strategies with $T = 1$, we can define a family of density dependent Markov chains [30] to describe the behavior of the stochastic finite systems with $N$ nodes for the conditional pull strategy. We can rely on the following generalization of Kurtz's theorem [23, Theorem 3.13]:

**Theorem C.2** (Kurtz)**.** *Consider a family of density dependent CTMCs, with* $F$ *Lipschitz. Let* $\lim_{N\to\infty} X^{(N)}(0) = \tilde{x}$ *a.s. and let* $\phi_t(\tilde{x})$ *be the unique solution to the initial value problem* $\frac{d}{dt}x(t) = F(x(t))$ *with* $x(0) = \tilde{x}$*. Consider the path* $\{\phi_t(\tilde{x}), t \leq T\}$ *for some fixed* $T \geq 0$ *and assume that there exists a neighborhood* $K$ *around this path satisfying*

$$\sum_{\ell \in L} |\ell| \sup_{x \in K} \beta_\ell(x) < \infty, \tag{C.1}$$

*then*

$$\lim_{N\to\infty} \sup_{t \leq T} \left| X^{(N)}(t) - \phi_t(\tilde{x}) \right| = 0 \quad a.s.$$

For the conditional pull strategy, the above condition (C.1) corresponds to showing that there exists an environment $K$ such that $\sum_{i\geq 2} \sup_{x\in K}(x_i - x_{i+1}) < \infty$. Such an environment can be shown to exist by repeating the argument for $T = 1$ from Theorem 2.8. To show that the convergence extends to the stationary regime, we can make use of a theorem by Benaïm and Le Boudec [32] as in the $T = 1$ case, where the required proof for the tightness of the measures can be proven as in Chapter 2.

Appendix D

# Bisection Algorithm to Find $\pi_{T+1}$ and $\pi_{B+1}$ from Theorem 5.1

The bisection algorithm is shown in Figure D.3. Figures D.1 and D.2 provide subroutines for the main algorithm. The parameter *prec* denotes the precision and is typically of the order $1e-10$.

```
1: procedure FIND_π_{T+1}(B, λ, r, prec, π_{B+1})
2:     dev ← π_{B+1}/4
3:     π_{T+1} ← π_{B+1}/2
4:     diff ← 1
5:     loop
6:         π ← λ
7:         for i ← 1, B do
8:             η ← (1 − λ) ∗ (λ + r ∗ π_{T+1})^i
9:             π ← π − η
10:        end for
11:        diff ← π − π_{B+1}
12:        if |diff| < prec then
13:            return π_{T+1}
14:        end if
15:        if diff < 0 then
16:            π_{T+1} ← π_{T+1} − dev
17:        else
18:            π_{T+1} ← π_{T+1} + dev
19:        end if
20:        dev ← dev/2
21:        if dev < prec then
22:            return π_{T+1}
23:        end if
24:    end loop
25: end procedure
```

Figure D.1: Bisection over $[0, \pi_{B+1}]$ to find the $\pi_{T+1}$ satisfying the first $B$ fixed point conditions.

```
1: procedure CHECK_π_{T+1}(B, T, λ, r, π_{B+1}, π_{T+1})
2:     η ← (1 − λ) ∗ (λ + r ∗ π_{T+1})^{B+1}
3:     π ← π_{B+1} − η
4:     for i ← (B + 2), T do
5:         η ← η ∗ λ
6:         π ← π − n_i
7:     end for
8:     return π − π_{T+1}
9: end procedure
```

Figure D.2: Checks how well a given $\pi_{T+1}$ value satisfies the fixed point conditions, given that it satisfies the first $B$.

```
1: procedure PULL(B, T, λ, R, prec)
2:     # bisect over [0, λ] to find π_{B+1}
3:     dev ← λ/4
4:     π_{B+1} ← λ/2
5:     diff ← 1
6:     loop
7:         r ← R/(1 − π_{B+1})
8:         # find π_{T+1} satisfying
9:         # the first B fixed point conditions
10:        π_{T+1} ← FIND_π_{T+1}(B, λ, r, prec, π_{B+1})
11:        #check the other conditions
12:        diff ← CHECK_π_{T+1}(B, T, λ, r, π_{B+1}, π_{T+1})
13:        if |diff| < prec then
14:            return π_{T+1}, π_{B+1}, r
15:        end if
16:        if |π_{T+1} − π_{B+1}| < 10 ∗ prec then
17:            π_{B+1} ← π_{B+1} + dev
18:        else
19:            if diff > 0 then
20:                π_{B+1} ← π_{B+1} − dev
21:            else
22:                π_{B+1} ← π_{B+1} + dev
23:            end if
24:        end if
25:        dev ← dev/2
26:        if dev < prec then
27:            return π_{T+1}, π_{B+1}, r
28:        end if
29:    end loop
30: end procedure
```

Figure D.3: This algorithm demonstrates how to compute the values of $\pi_{T+1}, \pi_{B+1}$ and $r$, for the pull strategy.

# Bibliography


[1] L. Gkatzikis and I. Koutsopoulos, "Migrate or not? exploiting dynamic task migration in mobile cloud computing systems," *IEEE Wireless Communications*, vol. 20, no. 3, pp. 24–32, June 2013.

[2] ——, "Mobiles on cloud nine: Efficient task migration policies for cloud computing systems," in *Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on*, Oct 2014, pp. 204–210.

[3] E. Schurman and J. Brutlag, "The user and business impact of server delays, additional bytes and http chunking in web search," OReilly Velocity Web performance and operations conference, June 2009.

[4] M. Mitzenmacher, "The power of two choices in randomized load balancing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 12, pp. 1094–1104, October 2001.

[5] N. Vvedenskaya, R. Dobrushin, and F. Karpelevich, "Queueing system with selection of the shortest of two queues: an asymptotic approach," *Problemy Peredachi Informatsii*, vol. 32, pp. 15–27, 1996.

[6] 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," *Perform. Eval.*, vol. 68, pp. 1056–1071, 2011.

[7] D. Eager, E. Lazowska, and J. Zahorjan, "A comparison of receiver-initiated and sender-initiated adaptive load sharing," *Perform. Eval.*, vol. 6, no. 1, pp. 53–68, 1986.

[8] R. Mirchandaney, D. Towsley, and J. Stankovic, "Analysis of the effects of delays on load sharing," *IEEE Trans. Comput.*, vol. 38, no. 11, pp. 1513–1525, 1989.

[9] N. Gast and B. Gaujal, "A mean field model of work stealing in large-scale systems," *SIGMETRICS Perform. Eval. Rev.*, vol. 38, no. 1, pp. 13–24, 2010.

[10] B. Van Houdt, "Performance comparison of aggressive push and traditional pull strategies in large distributed systems," in *Proceedings of QEST 2011, Aachen (Germany), IEEE Computer Society*, SEP 2011, pp. 265–274.

[11] V. Gupta, M. Harchol-balter, K. Sigman, and W. Whitt, "Analysis of join-theshortest-queue routing for web server farms," in *In PERFORMANCE 2007. IFIP WG 7.3 International Symposium on Computer Modeling, Measurement and Evaluation*, 2007.

[12] M. Mitzenmacher, "How useful is old information?" *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 1, pp. 6–20, Jan 2000.

[13] A. Stolyar, "Pull-based load distribution in large-scale heterogeneous service systems," *Queueing Systems*, vol. 80, no. 4, pp. 341–361, 2015. [Online]. Available: http://dx.doi.org/10.1007/s11134-015-9448-8

[14] D. Gamarnik, J. N. Tsitsiklis, and M. Zubeldia, "Delay, memory, and messaging tradeoffs in distributed service systems," in *Proceedings of the 2016 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science*, ser. SIGMETRICS '16. New York, NY, USA: ACM, 2016, pp. 1–12. [Online]. Available: http://doi.acm.org/10.1145/2896377.2901478

[15] C. Graham, "Chaoticity on path space for a queueing network with selection of the shortest queue among several," *J. Appl. Probab.*, vol. 37, no. 1, pp. 198–211, 03 2000. [Online]. Available: http://dx.doi.org/10.1239/jap/1014842277

[16] M. Bramson, Y. Lu, and B. Prabhakar, "Randomized load balancing with general service time distributions," *SIGMETRICS Perform. Eval. Rev.*, vol. 38, no. 1, pp. 275–286, Jun. 2010. [Online]. Available: http://doi.acm.org/10.1145/1811099.1811071

[17] L. Ying, R. Srikant, and X. Kang, "The power of slightly more than one sample in randomized load balancing," in *Proc. of IEEE INFOCOM*, 2015.

[18] D. Eager, E. Lazowska, and J. Zahorjan, "Adaptive load sharing in homogeneous distributed systems," *Software Engineering, IEEE Transactions on*, vol. SE-12, no. 5, pp. 662 –675, may 1986.

[19] R. Mirchandaney, D. Towsley, and J. A. Stankovic, "Adaptive load sharing in heterogeneous distributed systems," *J. Parallel Distrib. Comput.*, vol. 9, no. 4, pp. 331–346, 1990.

[20] I. Van Spilbeeck and B. Van Houdt, "Performance of rate-based pull and push strategies in heterogeneous networks," *Performance Evaluation*, vol. 91, pp. 2 – 15, 2015, special Issue: Performance 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0166531615000504

[21] M. S. Squillante and R. D. Nelson, "Analysis of task migration in shared-memory multiprocessor scheduling," *SIGMETRICS Perform. Eval. Rev.*, vol. 19, no. 1, pp. 143–155, Apr. 1991. [Online]. Available: http://doi.acm.org/10.1145/107972.107987

[22] M. Mitzenmacher, "Analyses of load stealing models based on families of differential equations," *Theory of Computing Systems*, vol. 34, pp. 77–98, 2001.

[23] ——, "The power of two choices in randomized load balancing," Ph.D. dissertation, University of California, Berkeley, 1996.

[24] A. Mukhopadhyay, A. Karthik, R. R. Mazumdar, and F. Guillemin, "Mean field and propagation of chaos in multi-class heterogeneous loss models," *Performance Evaluation*, vol. 91, pp. 117 – 131, 2015, special Issue: Performance 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0166531615000565

[25] W. Minnebo and B. Van Houdt, "A fair comparison of pull and push strategies in large distributed networks," *IEEE/ACM Transactions on Networking*, 2013.

[26] ——, "Pull versus push mechanism in large distributed networks: Closed form results," in *Proc. of the 24-th International Teletraffic Congress*, Krakau (Poland), 2012.

[27] K. Deimling, *Ordinary Differential Equations in Banach spaces.* Lect. Notes in Math. 596, 1977.

[28] J. Walker, *Dynamical Systems and Evolution Equations. Theory and Applications.* Plenum Press, New York, 1980.

[29] M. Shaked and J. G. Shanthikumar, *Stochastic Orders and their Applications.* Associated Press, 1994.

[30] T. Kurtz, *Approximation of population processes.* Society for Industrial and Applied Mathematics, 1981.

[31] M. Benaïm and J. Le Boudec, "A class of mean field interaction models for computer and communication systems," *Performance Evaluation*, vol. 65, no. 11-12, pp. 823–838, 2008.

[32] ——, "On mean field convergence and stationary regime," *CoRR*, vol. abs/1111.5710, Nov 24 2011.

[33] P. Billingsley, *Probability and Measure.* John Wiley and Sons, 1979.

[34] F. de Blasi and G. Pianigiani, "Uniqueness for differential equations implies continuous dependence only in finite dimension," *Bulletin of the London Mathematical Society*, vol. 18, no. 4, pp. 379–382, 1986.

[35] W. Minnebo and B. Van Houdt, "Improved rate-based pull and push strategies in large distributed networks," in *Proc. of the IEEE 21-th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, San Francisco (USA), 2013.

[36] S. Ethier and T. Kurtz, *Markov processes: characterization and convergence.* Wiley, 1986.

[37] M. Benaïm, "Recursive algorithms, urn processes and chaining number of chain recurrent sets," *Ergodic Theory and Dynamical Systems*, vol. 18, pp. 53–87, 1998.

[38] W. Minnebo and B. Van Houdt, "Analysis of rate-based pull and push strategies with limited migration rates in large distributed networks," in *9th EAI International Conference on Performance Evaluation Methodologies and Tools*. ACM, Jan. 2016.

[39] S. Dhakal, "Load balancing in delay-limited distributed systems," Ph.D. dissertation, The University of New Mexico, 2003.

[40] R. Mirchandaney, "Adaptive load sharing in the presence of delays," Ph.D. dissertation, Yale University, 1988.

[41] R. Aghajani, X. Li, and K. Ramanan, "Mean-field Dynamics of Load-Balancing Networks with General Service Distributions," *ArXiv e-prints*, Dec. 2015.

# Samenvatting

Vroeger werd een computerprogramma uitgevoerd op een enkele machine, zonder enige communicatie met de buitenwereld. Tegenwoordig communiceren vele programma's via een lokaal netwerk of het internet. Op die manier kunnen computers vanop verschillende locaties met elkaar samenwerken. Zo'n groep van computers die onderling verbonden zijn met een netwerk, en hun acties met elkaar coördineren is een *gedistribueerd systeem*.

Het ontwerpen, opzetten en onderhouden van een gedistribueerd systeem is uitdagend, en dit brengt verschillende interessante problemen met zich mee. Veelal wordt een programma opgedeeld in een set van taken die moeten uitgevoerd worden, al dan niet afhankelijk van elkaar. In deze thesis bekijken we hoe we de taken het best verdelen over de beschikbare servers. Specifiek willen we nagaan hoe we het systeem op een gedistribueerde manier de taken kunnen laten herverdelen na een willekeurige initiële toekenning van taken aan servers om zo de responsiviteit van taken te verhogen. Het kunnen herverdelen van taken is een belangrijke eigenschap van hedendaagse gedistribueerde systemen. Dit is het *load balancing* probleem.

In deze thesis bestuderen we gedistribueerde oplossingen voor dit probleem. Deze zijn onder te verdelen in twee categoriën: *push* en *pull* strategiën. Gebruikmakend van een pull strategie gaan de onderbelaste servers op zoek om taken van overbelaste servers over te nemen. Onder de push strategie zijn de rollen omgedraaid en zoeken de overbelaste servers de onderbelaste servers op om taken aan door te geven. Hierbij houden we rekening met hoeveel communicatie een bepaalde strategie vergt, en ontwikkelen we varianten die aan een vooropgestelde communicatie limiet kunnen voldoen. Op deze manier kunnen we strategiën vergelijken onder de voorwaarde dat ze evenveel communiceren. Dit is belangrijk om in het oog te houden, daar een strategie die meer communiceert potentiëel betere prestaties kan boeken. Ook bekijken we het geval waarbij er slechts een beperkt aantal taken verplaatst mag worden bovenop de gelimiteerde communicatie.

Na de introductie in hoofdstuk 1, ontwikkelen en vergelijken we push en pull strategiën die een vooropgesteld communicatie budget gebruiken in hoofdstuk 2. Hierbij vinden we ook dat deze nieuwe strategiën met periodische communicatie equivalent zijn met traditionele strategiën waarbij communicatie op vaste tijdstippen plaatsvindt.

In hoofdstuk 3 verbeteren we de prestaties van de push strategie door op te merken dat het voordelig is enkel servers die al verschillende taken hebben te laten communiceren. Een verdere uitbreiding bestaat erin om taken van servers met lange wachtrijen ogenblikkelijk te migreren naar een lege server, bovenop de periodische communicatie. Voor de pull strategie tonen we aan dat het optimaal is om enkel

servers die geen taken hebben te laten communiceren, en alle taken te accepteren die op dat moment aan het wachten zijn.

In hoofdstuk 4 voeren we een extra limitatie in, namelijk dat het aantal taken dat mag gemigreerd worden ook beperkt is. We tonen aan hoe de parameters van de strategiën te kiezen zodat noch de communicatie, noch de migratie limiet overschreden wordt. In deze setting blijkt het voor de pull strategie ook voordelig om enkel taken te accepteren van servers met langere wachtrijen, en ontwikkelen we een pull variant die beter presteert in deze omstandigheden.

In hoofdstuk 5 veralgemenen we het model verder zodat ook servers die al taken hebben, een migrerende taak mogen accepteren. Net zoals in hoofdstuk 2 blijken deze strategiën met periodische communicatie equivalent zijn met traditionele strategiën waarbij communicatie op vaste tijdstippen plaatsvindt. Binnen dit algemene model tonen we zowel push als pull strategiën waarvan we vermoeden dat ze optimaal zijn. Verder vergelijken we deze vermoedelijk optimale strategiën met een populaire en performante push strategie uit de literatuur, d-choices. Bovendien ontwikkelen we strategiën die equivalent zijn met d-choices, waarbij we tonen dat hetzelfde resultaat ook met periodische communicatie kan bereikt worden.