

# Mean Field Analysis for the Power of $d$ Choices.



Tim Hellemans, Benny Van Houdt

March 21, 2021

# Contents

<b>1</b>	<b>Introduction</b>	<b>13</b>
1.1	Motivation . . . . .	13
1.2	Perks of reading this thesis . . . . .	17
1.3	Overview of results . . . . .	18
1.4	Job size distribution and notation . . . . .	23
1.5	Informal intuition : cavity queue for $SQ(d)$ . . . . .	26
1.5.1	Exponential job sizes . . . . .	26
1.5.2	Phase Type Job Sizes . . . . .	32
1.6	Formal definition of the cavity queue . . . . .	35
1.7	Supplementary material . . . . .	39
<b>2</b>	<b>The power of <math>d</math> choices using least loaded server selection</b>	<b>40</b>
2.1	Introduction . . . . .	41
2.2	Structure of the chapter . . . . .	43
2.3	Informal intuition . . . . .	44
2.3.1	General job sizes . . . . .	44
2.3.2	Exponential job sizes . . . . .	47
2.3.3	Phase type job sizes . . . . .	48
2.4	Model description . . . . .	48
2.5	Cavity process . . . . .	49
2.6	Limiting workload distribution . . . . .	53
2.6.1	Fixed point iteration . . . . .	55
2.7	Exponential job sizes . . . . .	58
2.7.1	Limiting workload distribution . . . . .	58
2.7.2	Limiting waiting and response time distribution . . . . .	61
2.8	Phase-type and deterministic job sizes . . . . .	64
2.9	$LL(d)$ versus $SQ(d)$ . . . . .	66
2.9.1	Exponential job sizes . . . . .	67

2.9.2	Impact of job variability . . . . .	72
2.9.3	Late binding overhead . . . . .	74
2.10	Finite system accuracy . . . . .	75
2.11	Conclusions and future work . . . . .	78
<b>3</b>	<b>Performance of redundancy(<math>d</math>) with identical/independent replicas</b>	<b>79</b>
3.1	Introduction . . . . .	80
3.2	Structure of this chapter . . . . .	81
3.3	Informal intuition . . . . .	82
3.4	Model description . . . . .	85
3.5	Cavity process . . . . .	87
3.6	Equilibrium regime . . . . .	92
3.6.1	$\text{Red}_{\text{eq}}(d)$ . . . . .	92
3.6.2	$\text{Red}_{\text{eq}}(d)$ with delayed cancellation . . . . .	95
3.6.3	$\text{Red}_{\text{iid}}(d)$ . . . . .	96
3.6.4	Numerical considerations . . . . .	98
3.7	Validation of the model . . . . .	102
3.7.1	Finite system accuracy . . . . .	102
3.7.2	Stability region . . . . .	103
3.8	Numerical experiments for $\text{Red}_{\text{eq}}(d)$ . . . . .	104
3.8.1	Mean response time and workload distribution . . . . .	104
3.8.2	Impact of the job size variability . . . . .	106
3.8.3	Stability . . . . .	107
3.8.4	Variance of the response time distribution . . . . .	108
3.8.5	Tail of the Response Time distribution . . . . .	110
3.8.6	$\text{Red}_{\text{eq}}(d)$ with cancellation delay . . . . .	111
3.9	Comparison $\text{Red}_{\text{eq}}(d)$ and $\text{Red}_{\text{iid}}(d)$ . . . . .	111
3.9.1	$\text{Red}_{\text{iid}}(d)$ stochastically outperforms $\text{Red}_{\text{eq}}(d)$ . . . . .	112
3.9.2	Mean response time and workload distribution . . . . .	114
3.9.3	Impact of job size variability . . . . .	115
3.9.4	Stability . . . . .	116
3.9.5	Tail of the response time distribution . . . . .	117
3.10	Future work . . . . .	117
<b>4</b>	<b>Performance analysis of workload dependent load balancing policies</b>	<b>121</b>
4.1	Introduction . . . . .	122

4.2	Structure of this chapter . . . . .	125
4.3	Informal intuition . . . . .	125
4.4	Model description . . . . .	126
4.5	Cavity process . . . . .	127
4.6	Mean-field analysis . . . . .	130
4.6.1	Transient behaviour . . . . .	130
4.6.2	Equilibrium environment . . . . .	132
4.7	Load balancing policies . . . . .	133
4.7.1	Type 1 : Red( $d, k, \delta$ ) . . . . .	135
4.7.2	Type 1 : RTQ( $d, T$ ) . . . . .	138
4.7.3	Type 2 : DR( $d, T$ ) . . . . .	143
4.7.4	Type X : Replicate only small jobs . . . . .	145
4.7.5	Type 3 : LL( $d, k, \delta$ ) . . . . .	147
4.7.6	Type 4 : JTQ( $d, T$ ) . . . . .	155
4.8	Phase type distribution . . . . .	158
4.8.1	Red( $d, k, \delta$ ) . . . . .	158
4.8.2	RTQ( $d, T$ ) . . . . .	160
4.8.3	LL( $d, k, \delta$ ) . . . . .	161
4.9	No slowdown . . . . .	161
4.9.1	Red( $d, k, \delta$ ) . . . . .	161
4.9.2	RTQ( $d$ ) . . . . .	163
4.9.3	DR( $d, T$ ) . . . . .	166
4.10	Numerical method . . . . .	166
4.10.1	Computing the workload distribution . . . . .	167
4.10.2	Stability . . . . .	168
4.11	Validation . . . . .	169
4.11.1	Workload distribution . . . . .	169
4.11.2	Stability . . . . .	170
4.12	Future work . . . . .	170
<b>5</b>	<b>Mean waiting time in large-scale and critically loaded power of <math>d</math> load balancing systems</b>	<b>174</b>
5.1	Introduction . . . . .	176
5.2	Structure of this chapter . . . . .	179
5.3	Informal intuition . . . . .	180
5.4	General result . . . . .	182
5.4.1	The ordinary differential equation . . . . .	182
5.4.2	The recurrence relation . . . . .	183

5.4.3	Statement & application of the main result . . . . .	184
5.4.4	Computation of the limit . . . . .	189
5.4.5	Proof of Theorem 59 . . . . .	190
5.4.6	Proof of Theorem 60 . . . . .	193
5.5	LL( $d, K$ ) and SQ( $d, K$ ) . . . . .	196
5.6	LL( $d_1, \dots, d_n, p_1, \dots, p_n$ ) and SQ( $d_1, \dots, d_n, p_1, \dots, p_n$ ) . . . . .	214
5.6.1	The limit result . . . . .	216
5.6.2	The impact of $d_i$ and $p_i$ . . . . .	218
5.6.3	LL( $d, p$ ) . . . . .	220
5.7	Low load limit . . . . .	223
5.7.1	LL( $d$ ) . . . . .	223
5.7.2	LL( $d, K$ ) . . . . .	224
5.7.3	LL( $d_1, \dots, d_n, p_1, \dots, p_n$ ) . . . . .	225
5.7.4	Queue length dependent load balancing policies . . . . .	225
5.8	Numerical experiments . . . . .	225
5.9	Conclusions and extensions . . . . .	226
<b>6</b>	<b>Performance analysis of load balancing policies with memory</b>	<b>228</b>
6.1	Introduction . . . . .	229
6.2	Structure of the chapter . . . . .	231
6.3	Informal intuition . . . . .	231
6.4	Model description . . . . .	235
6.4.1	Classic SQ( $d$ ) . . . . .	235
6.4.2	Classic LL( $d$ ) . . . . .	236
6.5	Discovering idle servers . . . . .	236
6.5.1	Interrupted probing (IP) . . . . .	237
6.5.2	Continuous probing (CP) . . . . .	238
6.5.3	Bounded continuous probing (BCP) . . . . .	239
6.5.4	Other probing schemes . . . . .	240
6.5.5	Idle server messaging (ISM) . . . . .	240
6.6	Description of the queue at the cavity . . . . .	241
6.7	Analysis of the queue at the cavity . . . . .	243
6.7.1	SQ( $d$ ) . . . . .	243
6.7.2	LL( $d$ ) . . . . .	253
6.8	Finite system accuracy . . . . .	257
6.9	Numerical example . . . . .	257
6.10	Mean field limit for a critically loaded system . . . . .	259

6.11	Low load limit . . . . .	263
6.12	Conclusions and future work . . . . .	265
<b>7</b>	<b>Improved load balancing in large scale systems using attained service time reporting</b>	<b>267</b>
7.1	Introduction . . . . .	268
7.2	Structure of the chapter . . . . .	270
7.3	Informal intuition . . . . .	270
7.4	Model description . . . . .	274
7.5	Load balancing policies . . . . .	274
7.5.1	SQ( $d$ ) with Runtime based Tie Breaking (SQ( $d$ )-RTB)	275
7.5.2	SQ( $d$ ) with Runtime Exclusion (SQ( $d$ )-RE( $T$ ))	275
7.5.3	SQ( $d$ )-RTB with Runtime Exclusion (SQ( $d$ )-RTB-RE( $T$ ))	276
7.5.4	Least Attained Service (LAS( $d$ ))	276
7.5.5	LAS( $d$ ) with Queue length based Tie Breaking (LAS( $d$ )-QTB)	277
7.5.6	Runtime Exclusion (RE( $d, T$ ))	277
7.5.7	Least Expected Workload (LEW( $d$ ))	277
7.6	Model analysis . . . . .	278
7.6.1	Description of the cavity map . . . . .	278
7.6.2	Obtaining the steady state . . . . .	280
7.6.3	Determining the arrival rate . . . . .	283
7.6.4	Iterative procedure . . . . .	284
7.6.5	Obtaining performance measures . . . . .	285
7.6.6	Job size distribution . . . . .	286
7.7	Finite system accuracy . . . . .	288
7.8	Numerical experiments . . . . .	288
7.8.1	Impact of the number of phases $k$ . . . . .	291
7.8.2	Impact of the number of chosen servers $d$ . . . . .	292
7.8.3	Impact of the Squared Coefficient of Variation (SCV) .	294
7.8.4	Impact of the load from small jobs $f$ . . . . .	295
7.8.5	Tail of the waiting time distribution . . . . .	296
7.8.6	The granularity $\Delta$ . . . . .	297
7.8.7	Choice of the threshold $T$ . . . . .	298
7.9	Conclusions and future work . . . . .	299

<b>8 Open problems</b>	<b>303</b>
8.1 Refined mean field approximation . . . . .	303
8.2 Binary search to compute the load . . . . .	304
8.3 Obtaining the stability region . . . . .	305
8.3.1 General results . . . . .	306
8.3.2 Stability . . . . .	308
8.4 More results on critically loaded systems . . . . .	308
8.4.1 BSQ( $d, K$ ) . . . . .	308
8.4.2 The polynomial $p(x) = x^{K+1} - (1 + \frac{d}{K}x^K + \frac{d}{K})$ . . . . .	314
8.4.3 Erlang job sizes . . . . .	315
8.4.4 Deterministic job sizes . . . . .	316
8.4.5 The LL( $d$ ) policy with Erlang/Deterministic job sizes .	317
8.4.6 Convex combinations . . . . .	317
8.5 Bounds on the mean waiting time . . . . .	318
8.6 Alternative scheduling policies . . . . .	320
8.7 Memory dependent load balancing continued . . . . .	321
8.7.1 SQ( $d$ ) . . . . .	321
8.8 Link between queue length and workload dependent load bal- ancing . . . . .	323
<b>Appendices</b>	<b>324</b>
<b>A The M/M/1 queue</b>	<b>325</b>
A.1 Level crossing - queue length . . . . .	326
A.2 Stationary distribution of a continuous time Markov chain .	327
A.3 Observing the System at completion times . . . . .	329
A.4 Level crossing - workload . . . . .	330

# Abstract

Markov processes have found widespread use in the analysis of computer systems and beyond. Over time the size of the systems under consideration has grown considerably, e.g. Google has hundreds of thousands of servers located in its various data centers. This growth in the system size has made conventional methods to analyse these Markov processes infeasible.

As such, deterministic approximations, also known as mean field or fluid models, have been introduced to analyse such large scale systems. Interestingly, these deterministic models have been shown to correspond to the limit of a sequence of appropriately scaled Markov processes showing that the systems behaviour becomes deterministic as the system size tends to infinity. These Markov processes typically have a countable state space and the limiting system is described by a set of ordinary differential equations.

However, in order to analyse large scale computer systems with general job size distributions, one needs to keep track of the age or residual service time of each job. This makes the state space uncountable and the natural candidate for the limiting system becomes a set of partial differential equations (PDEs).

In this thesis, we initiated the analysis of workload dependent load balancing policies. For these policies, the aforementioned PDEs can be reduced to a single integro differential equation or to ordinary differential equations. Therefore, these policies are relatively easy to study for general job sizes. We found that many existing policies fall into the category of workload dependent policies. Besides obtaining numerical methods to analyse these load balancing policies, we additionally prove many analytical results for these type of models.

In addition, we recognized that studying the more *classic* queue length dependent load balancing policies for general job sizes is indeed difficult. However, if one restricts to phase type job sizes (which are dense in the set

of all probability distributions) the analysis simplifies significantly. Furthermore, we introduce and analyse a set of policies which take both the queue length and the age of the job currently receiving service into account.

# Abbreviations

**FCFS** First Come First Served

**PS** Processor Sharing

**DHR** Decreasing Hazard Rate

**iid** Independent and Identically Distributed

**pdf** Probability Density Function

**cdf** Cumulative Distribution Function

**ccdf** Complemented Cumulative Distribution Function

**FPE** Fixed Point Equation

**FPI** Fixed Point Iteration

**FDE** Functional Differential Equation

**ODE** Ordinary Differential Equation

**DDE** Delayed Differential Equation

**DIDE** Delayed Integro Differential Equation

**PIDE** Partial Integro Differential Equation

**DTMC** Discrete Time Markov Chain

**CTMC** Continuous Time Markov Chain

**SCV** Squared Coefficient of Variation

**W** Waiting time distribution

**Q** Queue length distribution

**R** Response time distribution

**L/U** Workload distribution

**LL( $d$ )** Least Loaded  $d$

**LL( $d, K$ )** Assign  $K$  jobs to the  $K$  least loaded queues amongst  $d$  randomly selected servers.

**SQ( $d$ )** Shortest Queue  $d$

**SQ( $d, K$ )** Assign  $K$  jobs to the  $K$  shortest queues amongst  $d$  randomly selected servers.

**Red(d)** Redundancy  $d$

# Notation

$\lambda$  The arrival rate.

$\lambda_{\max}$  The system is stable for all  $\lambda < \lambda_{\max}$  but unstable for  $\lambda \geq \lambda_{\max}$ .

$N$  The number of servers.

$d$  The number of selected servers at each arrival instant.

$K$  The number of incoming jobs in a batch arrival.

For a random variable  $X$  we denote:

$f_X$  The pdf of  $X$ .

$F_X$  The cdf of  $X$ .

$\bar{F}_X$  The ccdf of  $X$ .

$\mathbb{E}[X]$  The mean of  $X$ .

$\mathbb{E}[F_X]$  The mean of  $X$ .

$\text{Var}(X)$  The variance of  $X$ .

Sometimes (especially for the cavity queue's distribution) we leave out the index  $X$ .

$g, G, \bar{G}$  the pdf, cdf and ccdf of the job size distribution.

$f, F, \bar{F}$  the pdf, cdf and ccdf of the workload distribution of the cavity queue.

$u_k$  The probability that the cavity queue has  $k$  or more jobs.

$x_k$  The probability that the cavity queue has exactly  $k$  jobs.

$\lim_{x \rightarrow y^+} f(x)$  The limit of  $f(x)$  as  $x > y$  tends to  $y$ .

$\lim_{x \rightarrow y^-} f(x)$  The limit of  $f(x)$  as  $x < y$  tends to  $y$ .

${}_2F_1(a, b; c; z)$  The integral representation for the analytic continuation of the hypergeometric function.

$B(x, y)$  The Beta function.

$(q)_n$  The Pochhammer symbol (or falling fraction).

$(f_1 * f_2)(w)$  The convolution product, equal to  $\int_0^w f_1(u)f_2(w-u) du$ .

$Y_{(k)}$  The  $k$ 'th order statistic of  $Y_1, \dots, Y_n$  with  $n \geq k$ .

$I_A(x)$  Indicator function, equal to one if  $x \in A$ , zero otherwise.

$\lfloor x \rfloor$  The largest integer smaller than  $x$ .

$\lceil x \rceil$  The smallest integer larger than  $x$ .

$I_A(x)$  Indicator function, equal to one if  $x \in A$ , zero otherwise.

$\stackrel{d}{=}$  Equality in distribution

# Chapter 1

## Introduction

You can't feel the heat until you  
hold your hand over the flame  
You have to cross the line just  
to remember where it lays

---

Rise Against

### 1.1 Motivation

Discrete and continuous time Markov processes have been used in a wide range of scientific disciplines to analyse system behaviour [76, 72, 44]. While it is often not difficult to come up with a system description that is Markovian, a common problem exists in determining/computing its steady state in an efficient manner (provided that it exists). This problem is known as the state space explosion: in order to obtain a Markovian descriptor, the stochastic process has to keep track of various quantities which makes the state space large, both in the number of states and/or dimensions. This makes a conventional (numerical) analysis of the Markov process in many cases infeasible due to the associated memory and time complexity.

While there is in general no easy way to circumvent the state space explosion problem, it is possible to obtain accurate deterministic approximations for certain classes of Markov processes [54, 15, 31]. In such case, the deterministic approximation exists in studying the limit of an appropriate sequence of rescaled Markov processes and in showing that the sample paths of these

Markov chains converge (weakly, in probability or almost surely) to the trajectories of a deterministic system of equations on a suitably defined space. The evolution of the deterministic system is typically captured by a finite or infinite set of ordinary differential equations (ODEs). This set of ODEs is subsequently used to study the system behaviour and to gain engineering insights.

One example of a popular class of discrete time Markov processes for which such a result can be established is the following. Consider a set of  $N$  particles such that each particle is in some state part of a finite set  $S$  at any time  $t$ . Let  $X(N, n, t)$  be the state of particle  $n$  at time  $t$  in a system with  $N$  particles and denote  $M(N, s, t)$  as the fraction of the  $N$  particles in state  $s$  at time  $t$ , that is,

$$M(N, s, t) = \frac{1}{N} \sum_{n=1}^N I_{\{X(N, n, t)=s\}}.$$

Further assume that  $X(N, n, t + 1)$  is completely determined by  $X(N, n, t)$  and the vector

$$M(N, t) = (M(N, 1, t), \dots, M(N, |S|, t)),$$

meaning the transition probabilities are fully specified by:

$$\mathbb{P}\{X(N, n, t + 1) = s' \mid X(N, n, t) = s, M(N, t) = m\},$$

where  $m \in \Delta^{(N)} = \{(m_1, \dots, m_{|S|}) \mid \sum_i m_i = 1, m_i \in \{1, 2, \dots, N\}\}$  and  $s, s' \in S$ . In other words, the manner in which the particles interact is solely via the mean occupancy vector  $M(N, t)$ . Now define  $f(N, m)$  as the mean *drift vector* of the size  $N$  system at time  $t$  given that the occupancy vector equals  $m$ , being

$$f(N, m) = \mathbb{E}[M(N, t + 1) - M(N, t) \mid M(N, t) = m],$$

and let  $f(m) = \lim_{N \rightarrow \infty} f(N, m)/\varepsilon(N)$ , where  $\varepsilon(N)$  is typically  $1/N$ . Further define the rescaled process  $\tilde{M}(N, t)$  such that

$$\tilde{M}(N, t) = M(N, \lfloor tN \rfloor), \text{ for } t \geq 0.$$

Then under suitable conditions (which include the existence of the above limit  $f(m)$ ) one can establish (see [15]) the following result:

given that  $\tilde{M}(N, 0)$  converges to  $m$  in probability as  $N$  tends to infinity, we have for any finite  $T > 0$ :

$$\sup_{0 \leq t \leq T} \|\tilde{M}(N, t) - \mu(t)\| \rightarrow 0$$

in probability, where  $\mu(t)$  is the unique solution of the set of ODEs given by

$$\frac{d\mu(t)}{dt} = f(\mu(t)),$$

with initial condition  $\mu(0) = m$ . In other words, we can approximate  $M(N, t)$  for  $N$  sufficiently large by  $\mu(t/N)$ .

If in addition, this set of  $|S|$  ODEs has a unique fixed point  $\mu$  that is a global attractor on the space

$$\Delta = \{(x_1, \dots, x_{|S|}) \mid 0 \leq x_i \leq 1, \sum_i x_i = 1\},$$

then the stationary measures of the  $N$  dimensional Markov chains converge weakly to the Dirac measure  $\delta_\mu$ . Hence, we can gain insight on the long term system behaviour by studying the fixed point  $\mu$  of the set of ODEs.

This and similar classes of Markov chains have been used to analyse the performance of garbage collection algorithms used in flash-based solid state drives [88, 89], bin packing problems in cloud computing [99], work stealing and sharing algorithms in large distributed networks [41, 62], load balancing algorithms [95, 66], optical packet switches [91], etc. An example that illustrates the typical accuracy of a mean field model (taken from [42] is shown in figure 1.1).

In all of the above-mentioned examples the limiting system was described by a set of ODEs (or differential inclusions) and this was mostly due to the fact that certain components, such as the job duration in the load balancing, cloud computing and work stealing/sharing cases, are assumed to follow an exponential distribution. In practice, assuming exponential job durations is somewhat restrictive as in some cases service times tend to be more accurately modelled using a non-exponential distribution, such as a log-normal, gamma or shifted exponential distribution [3]. Mean field models for systems with general service times are very scarce and only started appearing very recently. The load balancing model of [95, 66] with exponential job durations was recently generalized to general service time distributions in [21],

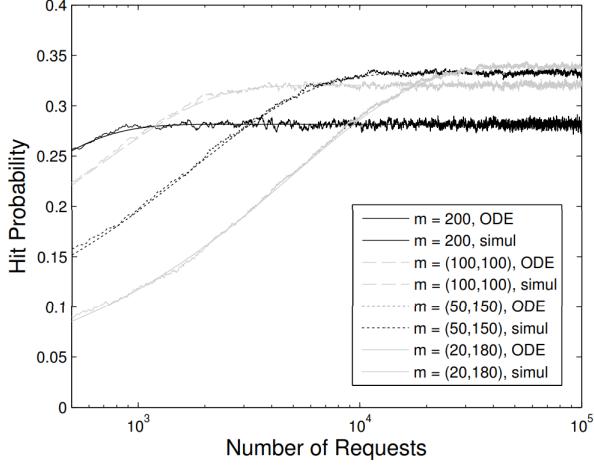


Figure 1.1: Comparison of the hit probability of a caching algorithm obtained by the mean field model (ODE) and by simulating the  $N = 1000$  dimensional Markov chain (averaged over 5 runs) for a cache of size  $m = 200$ .

for distributions with a decreasing hazard rate, and in [3], using a mean field model the limit of which is described by a set of partial differential equations (PDEs).

Another interesting model which was considered in [21] is the Least Loaded  $d$  policy. For this policy, the workload rather than the queue length is used to choose the queue to which an incoming job is assigned. From this we find in this thesis that it is possible to analyse this load balancing policy using a simple integro differential equation without placing any restrictions on the job size distribution. Moreover, if the job size distribution is exponential, we are able to find a closed form solution for this integro differential equation. Furthermore, we find that many other load balancing policies which are used in practice fall into this category of workload dependent load balancing policies. Besides obtaining a numerical method to analyse these load balancing policies, we additionally prove many analytical results for these type of models.

In addition, we recognized that studying the *classical* queue length dependent load balancing policies for general job sizes is indeed difficult. However, if one restricts to Phase Type job sizes (which are dense in the set of all probability distributions), the analysis simplifies significantly. Furthermore,

we introduce a set of policies which take both the queue length and the age of the job currently receiving service into account.

## 1.2 Perks of reading this thesis

While the main body of this thesis is a collection of articles we wrote in my period as pre-doctoral student, there are a few good reasons why reading this thesis rather than the articles which are presented in it might be beneficial.

1. We have tried to make the text as self-contained as possible, we only require the reader to have a basic knowledge of probabilities (including Markov chain theory) and calculus (integration, differential equations, ...). For readers who have little previous experience with queueing theory, we have added Appendix A in which we lay out all the queueing theory background needed to be able to understand the text at hand.
2. The method referred to as the *queue at the cavity* is explained clearly such that anyone should be able to both grasp its meaning and use it for their own applications. In this way, we would like people to see this thesis as a tutorial on the queue at the cavity method. In order to convey this message as clear as possible, we added extensive intuitive explanations of its application to some models and made videos explaining how these models can be analysed. In addition, we added an *Informal Intuition* section to each chapter, giving a feel for what the methods used to derive the results in the chapter are without getting into the technical details. Associated to these intuitive sections, we made a video for each chapter in which we explain the same basic notions as in the intuitive sections. This way we hope that also less experienced readers are able to quickly grasp what this thesis is actually about. These videos can be found at <https://github.com/THellemans/thesis>.
3. Many of our results yield a numerical method which allows to study load balancing policies analytically. Implementing these numerical methods may be more challenging than one would expect, especially when the load is extremely high the implementation requires some thought. Therefore we collected the code used to generate the plots in this thesis at our Github page <https://github.com/THellemans/thesis>. At this location, one can also find the simulation code which was used to verify all results.

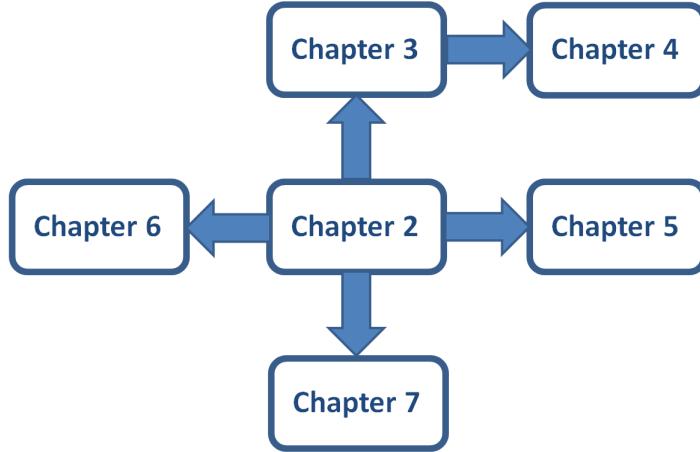


Figure 1.2: This scheme reflects the dependencies of the chapters in this work.

4. As we have been working on pretty much the same subject for four years, many insights were gained at a later stage on a subject which was already published. We could not adapt the original papers, but we made additions and corrections throughout the papers and therefore the chapters in this thesis may be viewed as *improved versions* of the original papers.
5. We end with a chapter consisting of a list of problems which we were only partly able to solve. These subjects may be worthwhile to investigate further. Reading this thesis in addition to those problems might provide the reader with inspiration to further elaborate on these subjects, which could potentially turn these subjects into papers. Of course, the more experienced reader may prefer to only read Chapter 8.

While every chapter can be read separately, we suggest (especially for the less experienced reader) to respect the order outlined in Figure 1.2

### 1.3 Overview of results

Throughout this text, we always assume that there are  $N$  homogeneous servers which all work on jobs at a constant rate equal to one. We often

assume (in particular for our numerical experiments) the mean job size is equal to one. This is a mere technicality, mainly used to compare different plots in function of the arrival rate and to better understand the stability regions for different policies.

All scheduling policies we consider are non-idling (in the sense that, when there is work at a server, the server executes the work). We mainly focus on the First Come First Served (FCFS) scheduling policy, although some results (such as the results on the workload distribution for the Least Loaded  $d$  policy) are independent of the scheduling policy. There are always one or more dispatcher(s) which assign incoming jobs to servers with the objective of minimizing the response time while not causing too much overhead. To this end, we employ several algorithms which exploit the power-of- $d$ -choices: at each arrival instant the dispatcher selects  $d$  servers at random, it then applies some rule to distribute the incoming job on these  $d$  servers. Pioneering works in this direction are [63] and [95]. Our models include (but are not restricted to):

- (a) **LL( $d$ )** : Least Loaded  $d$ , this model assumes each job joins the server which has the least work left amongst  $d$  randomly selected servers. This is the server at which the job can start receiving service first. We study this policy in detail in Chapter 2.
- (b) **Red( $d$ )** : Redundancy  $d$ , for this model  $d$  servers are selected and a replica of the incoming job is assigned to each of the  $d$  servers. As soon as one of the replicas completes service, all  $d - 1$  other replicas are cancelled. A detailed study for this policy can be found in Chapter 3.
- (c) **DR( $d, T$ )** Delayed Replication  $d, T$ , this model assigns the incoming job arbitrarily and assigns replicas to  $d - 1$  other servers if it has not finished service after time  $T$ . All replicas are deleted as soon as one of the  $d$  copies finishes service. This policy and other, more exotic, policies are seen to fit a general framework in Chapter 4.
- (d) **SQ( $d, K$ )** : Shortest Queue  $d, K$ , here we assume jobs arrive in batches of size  $K$ . For each batch we probe  $d$  queues and assign one incoming job to each of the  $K$  shortest queues (out of the  $d \geq K$  which were selected). In particular we show for this policy that (with  $W_\lambda$  the mean

waiting time for traffic intensity  $\lambda$ )

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1 - \lambda)} = \frac{1}{\log\left(\frac{d}{K} - 1\right)}.$$

This proof is quite involved and we present it in Chapter 5.

- (e) **Memory Based Policies** : The dispatcher has a (finite) memory of idle servers and it only probes to find a good server when it has no idle servers in its memory, otherwise the incoming job is simply assigned to an idle queue in memory. These type of policies are the subject of Chapter 6.
- (f) **SQ( $d$ )-RTB** : Shortest Queue  $d$  with Runtime based Tie Breaking, for each arrival we probe  $d$  servers and the incoming job joins the server which has been executing the job at the head of the queue for the shortest period of time. This policy, and other policies which combine runtime and queue length information are the subject of Chapter 7.

**Remark 1.** *There are two technicalities which we need to address:*

- *We distinguish between selecting  $d$  servers with or without replacement. For the mean field limit (that is as the number of servers  $N$  tends to infinity), these two models are equivalent. For the simulation experiments, we always assume that servers are chosen with replacement.*
- *When selected servers have the same amount of work or the same number of jobs in their queue, we assign the job arbitrarily between these servers.*

From a mathematical point of view, the load balancing policies we consider can be split into two main categories. The first category exists of queue length dependent load balancing policies where the dispatcher collects some information on the number of jobs in some servers and assigns an incoming job using this information. A well studied example of this policy type is the SQ( $d$ ) policy, where an incoming job is assigned to the shortest among  $d$  randomly selected servers (see e.g. [63, 95] and Section 1.5). The analysis of this type of models can be seen as extensions of the methods for the analysis of the M/M/1 queue which we present in Appendix A.1, A.2 and A.3.

The second category consists of workload dependent load balancing policies, for these policies the dispatcher balances the load on the servers by

employing information on the amount of work that is left on some of the servers (see also Chapter 4). This can be done explicitly if we assume the amount of work on servers is known or implicitly by employing some form of redundancy such as e.g. cancellation on start or late binding (see also [73]). The methods used for this type of models can be seen as extensions of the analysis for the M/M/1 queue which we present in Appendix A.4.

Another way to see this distinction is as follows: If one were to consider the  $N$ -dimensional stochastic process consisting of the queue lengths resp. the workload at all of the different servers, then this stochastic process forms a Markov chain for queue length resp. workload dependent load balancing policies. We use SQ( $d$ ) resp. LL( $d$ ) as the *standard examples* for the queue length resp. workload dependent load balancing policies as they are the most simple to understand and analyse. Policies which depend both on the queue length and workload information of the queue are much more difficult to analyse and we do not consider them in this work. However, we do consider policies which depend both on the queue length and the runtime of the job at the head of the queue in Chapter 7. The state space of these models is mix of a continuous and a discrete state space, we are able to analyse them by discretizing the continuous component, this policy type can be found in Chapter 7.

As the analysis of the aforementioned  $N$  dimensional Markov chain is intractable for large  $N$ , we require an alternative analytical approach. Therefore we analyse the *cavity queue* associated to each model, this method was introduced as a method to analyse power-of- $d$ -choices models in [20]. This method is explained in more detail in Section 3.5, and applications of the theory can be found throughout this work. In short: we assume all servers behave as i.i.d. random variables, therefore we only need to analyse one (cavity) queue. The whole system is then defined through the behaviour of this single queue.

While the analysis of workload and queue length dependent load balancing policies are fundamentally different, there are also strong similarities! Let us illustrate one similarity here : If we denote by  $u_k$  the probability that the cavity queue has  $k$  or more jobs and  $\bar{F}(w)$  the probability that the cavity queue has  $w$  or more work. We have found that for many queue length dependent policies (assuming exponential job sizes), the value of  $u_k$  can be found from some recursive formula:

$$u_{k+1} = T(u_k),$$

while for the workload dependent counterpart,  $\bar{F}(w)$  is the solution of the ODE:

$$\bar{F}'(w) = T(\bar{F}(w)) - \bar{F}(w),$$

where  $T$  is the same function in both equations (see Chapter 5 for more details). This observation then allows to see the similarity in proofs for both model types.

We give a short overview of our main innovations.

- In case of exponential job sizes, we find elegant closed form solutions for the performance characteristics of the Least Loaded  $d$  (in short: LL( $d$ )) policy and the LL( $d$ )/SQ( $d$ ) (Shortest Queue  $d$ ) policies with a dispatcher which remembers ids of idle servers. These closed form solutions can be used to show many interesting results for these policies. These results can be found in Chapters 2 and 6.
- Most results revolving the cavity queue approach consider queue length dependent load balancing policies with either exponential or phase type job sizes. In contrast, we mainly study workload dependent load balancing policies in large scale systems. This is done by using the workload rather than the queue length to describe the state of the queue at a particular time. This way we are able to characterize the complemented cumulative distribution function of the workload distribution as the fixed point of an FDE which can be solved either numerically or explicitly. This is the essence of Chapters 2, 3, 4 and 6.
- Letting  $W_\lambda$  denote the waiting time for a load balancing model with traffic intensity  $\lambda$ , we find that the limit  $\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)}$  can be expressed explicitly using a simple formula for many load balancing policies. This analysis is carried out in Chapter 5. We believe this analysis can be extended far beyond the results presented in Chapter 5, therefore we have dedicated a substantial portion of the open problems to this subject.
- In Chapter 7, we improve the classical SQ( $d$ ) policy by using the run-time information of the job which is currently receiving service.

## 1.4 Job size distribution and notation

Throughout, we denote by  $X$  a generic random variable which represents the size of an arbitrary job. In general we assume  $X$  is any type of non-zero job size distribution which can be decomposed into a continuous and a discrete part, i.e. we have  $0 \leq X = X_1 + X_2$  where  $0 \leq X_1$  is a continuous random variable and  $0 \leq X_2$  is a discrete random variable. We exclude the case where  $X$  may have a singular component or an atom at 0.

The cumulative distribution function (cdf) of  $X$  is denoted  $F_X(\cdot)$ , the complemented cdf (ccdf) is denoted by  $\bar{F}_X(\cdot)$  and the probability density function (pdf) of its continuous part is denoted by  $f_X(\cdot)$ . If there is only one job type and no slowdown, we typically use the notation  $G(\cdot) = F_X(\cdot)$ ,  $\bar{G}(\cdot) = \bar{F}_X(\cdot)$  and  $g(\cdot) = f_X(\cdot)$ . We assume the job size distribution has no atom in zero, thus  $G(0) = 0$ . We can decompose  $G$  in the continuous and discrete part by stating  $G = G_1 + G_2$ , where  $G_1$  has a density function  $g_1 : [0, \infty) \rightarrow [0, \infty)$ ,  $G_2$  has a discrete density given by  $g_2(\cdot) = \sum_{n=0}^{\infty} p_n \delta_{a_n}(\cdot)$  (assuming  $0 < a_0 < a_1 < \dots$ ) and we have

$$\int_0^{\infty} g_1(u) du + \sum_{n=0}^{\infty} p_n = 1.$$

Here we use the notation  $\delta_x$  for the Dirac measure. For simplicity we denote  $g(w) dw = g_1(w) dw + g_2(w)$ , thus for a subset  $A \subseteq [0, \infty)$  we have

$$\int_A g(w) dw = \int_A g_1(w) dw + \sum_n p_n \delta_{a_n}(A),$$

where  $\delta_a(A)$  equals one if  $a \in A$  and zero otherwise. By abuse of notation, for  $w \in (0, \infty)$  we write

$$g(w) = g_1(w) + \sum_{n=0}^{\infty} p_n \delta\{w = a_n\}$$

(where  $\delta\{x = y\}$  equals one if  $x = y$  and zero otherwise). We generally employ the following notation: a capital letter for cdf, a capital letter with an overline for ccdf, a lowercase letter for pdf and  $\mathbb{E}$  for an expectation.

We often assume (in particular for our numerical experiments) the mean job size, denoted by  $\mathbb{E}[X]$  or  $\mathbb{E}[G]$  is equal to one. This is a mere technicality, mainly used to compare different plots in function of the arrival rate  $\lambda$  and

to better understand the stability regions for different policies. We define the incoming amount of work by  $\rho = \lambda\mathbb{E}[X]$ . In general, for policies which only assign one replica of each incoming job, one finds that the policy is stable as long as  $\rho < 1$ .

We often restrict our interest to exponential job sizes with parameter 1, for this job size distribution, we are able to obtain the most elegant results. In particular, to obtain closed form expressions, we require to restrict ourselves to exponential job sizes.

A second type which enables us to obtain improved numerical schemes is the case of PH Distributed jobs. PH distributions are distributions with a modulating finite state background Markov chain [55] and any general positive-valued distribution can be approximated arbitrary closely with a PH distribution. Further, various fitting tools are available online for PH distributions (e.g., [75, 52]). A PH distribution with  $G(0) = 0$  is fully characterized by a stochastic vector  $\alpha = (\alpha_i)_{i=1}^n$  and a subgenerator matrix  $A = (a_{i,j})_{i,j=1}^n$  such that  $\tilde{G}(w) = \alpha e^{Aw}\mathbf{1}$ , where  $\mathbf{1}$  is a column vector of ones. Furthermore, one defines  $\mu = -A\mathbf{1}$  and finds that the pdf is given by  $g(w) = \alpha e^{Aw}\mu$ . We now explain how to interpret a PH distribution. We have the following transition rate matrix:

$$Q = \begin{pmatrix} A & \mu \\ 0 & 0 \end{pmatrix},$$

with its initial state chosen according to the vector  $(\alpha, 0)$ . The PH distribution with parameters  $(\alpha, A)$  is now defined as the absorption time of the Continuous Time Markov Chain (CTMC) with transition rate matrix  $Q$  and initial distribution  $(\alpha, 0)$ .

In words: at any time, the job can be in  $n$  possible phases (denoted by phase  $1, \dots, n$ ). When the job starts receiving service it will start in phase  $i$  with probability  $\alpha_i$ . Its phase then changes according to the subgenerator matrix  $A$ , i.e. when it is in phase  $i$ , its phase transitions to state  $j$  at rate  $A_{i,j}$ . Finally, when it is in state  $i$ , the job completes service at rate  $\mu_i$ .

For our numerical experiments, we are often interested in the behaviour of policies w.r.t. the Squared Coefficient of Variation (SCV), which is defined as  $\frac{\text{Var}(X)}{\mathbb{E}[X]}$ . The SCV of an exponential random variable is exactly equal to one. In real systems a significant part of the total workload is often offered by a small fraction of long jobs, while the remaining workload consists mostly of (very) short jobs [74]. We have 2 go-to PH distribution to represent job sizes with a higher resp. lower SCV. To obtain a higher SCV, we model the

job sizes as a hyperexponential distribution (with 2 phases) such that we can vary the job size variability in a systematic manner. More precisely, with probability  $p$  a job is a type-1 job and has an exponential length with parameter  $\mu_1 > 1$  and with the remaining probability  $1 - p$  a job is a type-2 job and has exponential length with parameter  $\mu_2 < 1$ . Hence, the type-2 jobs are longer on average and we therefore sometimes refer to the type-2 jobs as the *long* jobs. The parameters  $p, \mu_1$  and  $\mu_2$  are set such that the following three values are matched:

- (i) mean job length  $\mathbb{E}[X]$ ,
- (ii) the SCV and
- (iii) a shape parameter  $f$ ,

to this end we use the following equations:

$$\begin{aligned}\mu_1 &= \frac{SCV + (4f - 1) + \sqrt{(SCV - 1)(SCV - 1 + 8f\bar{f})}}{2\mathbb{E}[X]f(SCV + 1)}, \\ \mu_2 &= \frac{SCV + (4\bar{f} - 1) - \sqrt{(SCV - 1)(SCV - 1 + 8f\bar{f})}}{2\mathbb{E}[X]\bar{f}(SCV + 1)},\end{aligned}$$

with  $\bar{f} = 1 - f$  and  $p = \mathbb{E}[X]\mu_1 f$ . The shape parameter  $f \in (0, 1)$  represents the fraction of the workload that is offered by the type-1 jobs. We note that this describes a PH distribution with parameters:

$$\alpha = (p \quad 1 - p) \text{ and } A = \begin{pmatrix} -\mu_1 & 0 \\ 0 & -\mu_2 \end{pmatrix}.$$

When modelling jobs with a low SCV, we employ Erlang( $k, k$ ) jobs (or in short: Erlang- $k$ ), which are defined as a sum of  $k$  independent exponential job sizes which all have a mean equal to  $\frac{1}{k}$ . One may compute that the SCV of an Erlang( $k, k$ ) distribution is equal to  $\frac{1}{k}$ . One can define this distribution using a PH distribution, we let :

$$\alpha = (1 \quad 0 \quad \dots \quad 0) \text{ and } A = \begin{pmatrix} -k & k & & & \\ & -k & k & & \\ & & \ddots & \ddots & \\ & & & -k & k \\ & & & & -k \end{pmatrix} \in \mathbb{R}^{k \times k}.$$

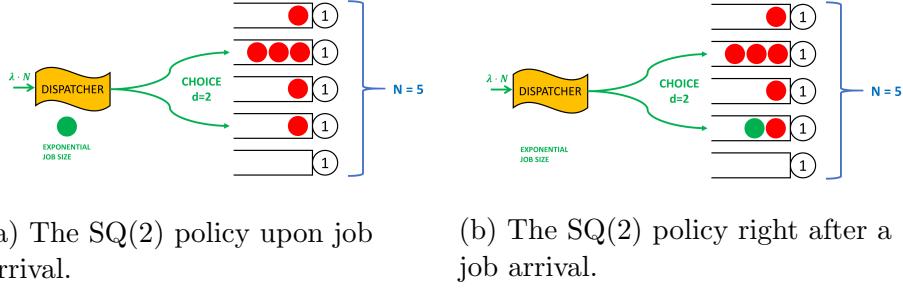


Figure 1.3: Graphical depiction of the  $\text{SQ}(2)$  policy with  $N = 5$  and arrival rate  $\lambda N$ .

## 1.5 Informal intuition : cavity queue for $\text{SQ}(d)$

### 1.5.1 Exponential job sizes

#### Level crossing argument

In this section we give an informal description which less experienced readers to understand how the analysis of the  $\text{SQ}(d)$  policy can be done without going into any details of the proofs. The analysis we carry out here is similar to the analysis in Section A.1.

We assume that job sizes are exponential with mean one, it follows that any working server finishes (on average) one job per time unit irrespective of the time the job has been in service (by virtue of the memoryless property of the exponential distribution). Therefore, all one needs to keep track of is the number of jobs in each queue (that is: the queue lengths). Jobs arrive as a Poisson process with rate  $\lambda \cdot N$ , which means that the time between 2 arrivals is (on average)  $\frac{1}{\lambda N}$ , and the time you have already been waiting for the next arrival does not impact the time until the next arrival occurs. Whenever a job arrives,  $d$  servers are chosen at random and the job is assigned to the server which has the least number of jobs in its queue.

One would like to answer the question, when you are a job which arrives to this system, *what's the probability that there will be  $k$  or more jobs in front of you?* In what follows, we show that this probability can simply be computed as  $u_k^d$  with  $u_k$  the probability that the *queue at the cavity* has  $k$  or more jobs, moreover we find that  $u_k = \lambda^{\frac{dk-1}{d-1}}$ . From this, it is not hard to compute performance measures such as the waiting time distribution and

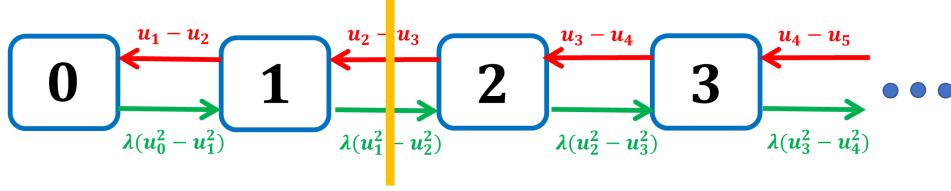


Figure 1.4: Scheme which illustrates the transitions for the  $\text{SQ}(d)$  policy with exponential job sizes with mean one.

the mean waiting time.

In Figure 1.3 we observe what happens at an arrival instant for the  $\text{SQ}(2)$  policy. In order to numerically analyse this model, we would set an upper bound on the number of jobs that can be present in a queue (let us take 3 as this upper bound). We would then describe the system in Figure 1.3a as  $(1, 3, 1, 1, 0)$ , and the arrival is simply a transition to the state  $(1, 3, 1, 2, 0)$ . One could attempt to compute the stationary distribution of this Markov chain, but the number of states is equal to  $3^N$ , which becomes prohibitively large as the value of  $N$  grows large. For example, when  $N = 500$ , the state space has  $3^{500} \approx 10^{238}$  states which renders this approach intractable.

When  $N = d = 2$ , it is obvious that whenever the servers are not well balanced, all subsequent arrivals will be routed to the server with the shortest queue, balancing out the loads. This incurs that in general the queue lengths will have a strong positive correlation. However, as  $N$  grows large, the probability of selecting 2 specific servers becomes negligible which makes the queue lengths independent. Moreover, as all queues are treated in the same way, one finds that the queue length distribution is identical for all servers. Therefore we can hide the whole system except for one cavity where we can still view exactly one queue, namely the *queue at the cavity*. We observe that queue and assume that all other queues are independent of this queue and have the same queue length distribution. For this cavity queue, we may denote by  $u_k$  the probability that it has  $k$  or more jobs. In order to make this analysis rigorous, one needs to prove the independence property, for  $\text{SQ}(d)$  this was done in [21] for job sizes which have a Decreasing hazard Rate (DHR). Further on, we set  $d = 2$  for simplicity.

In order to compute  $u_k$ , we use a level crossing argument, that is: *what goes up, must come down!* We note that the event *the cavity queue's load decreases from  $k$  to  $k-1$*  should occur the same number of times as the event

the cavity queue's load increases from  $k - 1$  to  $k$ .

To this end, we could represent the transitions of the cavity queue as in Figure 1.4. We draw a square for each possible state the cavity queue can be in, the number represents the number of jobs in the cavity queue. We note that the cavity queue has a probability of  $u_k - u_{k+1}$  to have a queue length equal to  $k$ . The green arrows represent the arrival rates while the red arrows indicate the departure rates. We can draw a vertical line anywhere in the diagram and the level-crossing argument can be formulated as stating that for any of these lines the rate at which we cross the line from left to right must equal the rate at which we cross the line from right to left. For the orange line we drew, this translates to the equality  $(u_2 - u_3) = \lambda(u_1^2 - u_2^2)$ . We now explain in detail how to obtain the transition rates given in Figure 1.4.

As the cavity queue finishes jobs at a rate equal to 1 job per time period, it follows for any  $k \geq 1$  that when the queue at the cavity has  $k$  jobs, the rate at which its queue length decreases from  $k$  to  $k - 1$  is equal to one. To obtain the total rate at which this transition occurs we must multiply by the probability that the queue at the cavity has queue length  $k$ , that is, at an arbitrary point in time the *down-crossing* rate from  $k$  to  $k - 1$  is given by:

$$\mathbb{P}\{\text{Queue at the cavity has } k \text{ jobs}\} \cdot 1 = u_k - u_{k+1} \quad (1.1)$$

To have an up-crossing from  $k - 1$  to  $k$ , the queue at the cavity must have  $k - 1$  jobs in its queue and it must experience an arrival. Note that the arrival rate is  $\lambda \cdot N$  and the probability that the arrival selects the cavity queue is equal to  $2/N$ , therefore the *potential arrival rate* is equal to

$$\lambda N \cdot 2/N = 2\lambda.$$

When the queue at the cavity is selected for a potential arrival, we know that some second queue (which we cannot see) is selected. We assume that the other queue has the same queue length distribution as the queue at the cavity, therefore it has  $\ell$  or more jobs with probability  $u_\ell$ . When the cavity queue has  $k - 1$  jobs and a potential arrival occurs, we find that there are 3 distinct possibilities:

1. With probability  $u_k$  the second queue has a higher load than the queue at the cavity and the arrival is routed to the queue at the cavity. In this case we do have an up-crossing.

2. With probability  $1 - u_{k-1}$  the second queue has a lower load than the queue at the cavity and the arrival is routed to the other queue. In this case we do not have an arrival to the queue at the cavity.
3. With probability  $u_{k-1} - u_k$  the second queue has the same load as the queue at the cavity, in this case, the job is routed to the queue at the cavity with probability  $\frac{1}{2}$ .

Summarizing all of the above we find that the up-crossing rate from  $k - 1$  to  $k$  is given by:

$$\begin{aligned}
 & \underbrace{(u_{k-1} - u_k)}_{\substack{\text{Queue at the cavity} \\ \text{has } k-1 \text{ jobs}}} \cdot \underbrace{2 \cdot \lambda}_{\substack{\text{Potential arrival at} \\ \text{the cavity queue.}}} \cdot \underbrace{\left( u_k + \frac{1}{2}(u_{k-1} - u_k) \right)}_{\substack{\text{The queue at the cavity} \\ \text{receives the job.}}} \\
 &= 2 \cdot \lambda(u_{k-1} - u_k) \cdot \left( \frac{1}{2}u_{k-1} + \frac{1}{2}u_k \right) \\
 &= \lambda(u_{k-1}^2 - u_k^2).
 \end{aligned}$$

Setting the up-crossing rate equal to the down-crossing rate, we find that the  $u_k$  values satisfy the following relation for all  $k \geq 1$  :

$$u_k - u_{k+1} = \lambda(u_{k-1}^2 - u_k^2).$$

This equality for  $k = 2$  is represented by the vertical orange line in Figure 1.4. Taking the sum for all values larger than  $k$  on both sides of this equality, we find that:

$$u_k = \sum_{\ell \geq k} (u_\ell - u_{\ell+1}) = \lambda \sum_{\ell \geq k} (u_{\ell-1}^2 - u_\ell^2) = \lambda u_{k-1}^2.$$

As we obviously have  $u_0 = 1$ , it follows that:

$$u_0 = 1, \quad u_1 = \lambda, \quad u_2 = \lambda \cdot u_1^2 = \lambda^{1+2}, \quad u_3 = \lambda u_2^2 = \lambda \lambda^{2 \cdot (1+2)} = \lambda^{1+2+2^2}, \dots$$

we have thus shown that  $u_k = \lambda^{\sum_{\ell=0}^{k-1} 2^\ell} = \lambda^{2^k - 1}$  for  $k \geq 1$ . We can answer the original question: there are  $k$  or more jobs in front of me when I arrive to this queueing system if both servers the dispatcher picks have  $k$  or more jobs, which occurs with probability  $u_k^2 = \lambda^{2^{k+1}-2}$ . We can go a bit further computing related performance measures.

In order to obtain the average waiting time a job experiences, let us first compute the average queue length  $\mathbb{E}[Q]$ :

$$\begin{aligned}\mathbb{E}[Q] &= \sum_{k=1}^{\infty} k \cdot \mathbb{P}\{\text{Queue length equal to } k\} \\ &= \sum_{k=1}^{\infty} k \cdot (u_k - u_{k+1}) = \sum_{k=1}^{\infty} u_k.\end{aligned}$$

Applying Little's Law allows us to conclude that the average waiting time a job experiences is given by  $\mathbb{E}[W] = \mathbb{E}[Q]/\lambda - 1$ . Moreover, the waiting time distribution can be computed as:

$$\begin{aligned}\bar{F}_W(w) &= \mathbb{P}\{\text{The waiting time exceeds } w\} \\ &= \sum_{k=1}^{\infty} (u_k^2 - u_{k+1}^2) \cdot \mathbb{P}\left\{\sum_{i=1}^k E_i > w\right\},\end{aligned}$$

where  $(E_i)_i$  is an i.i.d. sequence of exponential random variables with mean one. The sum  $\sum_{i=1}^k E_i$  is an Erlang( $k, 1$ ) distribution for which we have:

$$\mathbb{P}\left\{\sum_{i=1}^k E_i > w\right\} = \sum_{n=0}^{k-1} \frac{1}{n!} e^{-x} x^n.$$

Throughout this text, we are often faced with Erlang distributions. It is not hard to see that these observations generalize to the case  $d \geq 2$  for which we have  $u_k = \lambda^{\frac{d^k - 1}{d-1}}$ .

### Fixed point iteration

We introduce another method which can be used to analyse the SQ( $d$ ) policy with exponential job sizes. While the method might seem more involved, its main advantage is the fact that it generalizes well to other job size distributions and other models for which a simple recursion might not suffice to obtain the stationary queue length distribution. This method is a Fixed Point Iteration (FPI) where we *guess* some values for  $u_k$ , as a starting point. One could start by *guessing* that all queues are empty, that is:  $u_0 = 1$  and  $u_k = 0$  for all  $k \geq 1$ . We define some procedure which maps  $u = (u_k)_k$  to a new object  $T(u)$ . The solution  $u$  we are looking for is given by the fixed

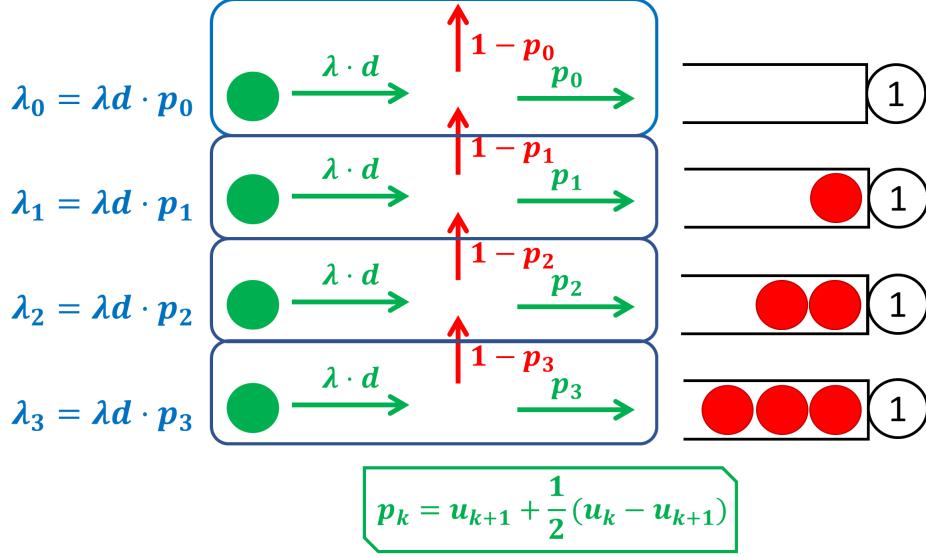


Figure 1.5: Scheme which illustrates the actual arrival rates at the queue at the cavity for the  $SQ(d)$  policy.

point of this map, that is the value  $u$  for which  $T(u) = u$ . The numerical method to obtain the queue length distribution is defined by iteratively setting  $u^{(n+1)} = T(u^{(n)})$  until convergence occurs.

Let us assume that we have some guess  $u = (u_k)_k$  for the queue length distribution. For each possible queue length  $k$  we know that the potential arrival rate to the cavity queue is given by  $\lambda \cdot d$ . Moreover, when the cavity queue has  $k$  jobs in its queue, and a potential arrival occurs, we can easily compute the probability that the potential arrival actually joins the queue at the cavity, it is given by

$$p_k = u_{k+1} + \frac{1}{2}(u_k - u_{k+1}) = \frac{1}{2}(u_k + u_{k+1})$$

(where the values of  $u_k$  are given by our guess). This observation is represented in Figure 1.5. From this, one can easily construct the transition rate matrix associated to the cavity queue  $Q$ , indeed its non-zero elements are given by:

- $Q_{i,i+1} = \lambda d p_i$ , for  $i \geq 0$  the rate at which a potential arrival occurs multiplied by the probability that it becomes an actual arrival.

- $Q_{i,i-1} = 1$ , for  $i \geq 1$  the rate at which the cavity queue finishes jobs in its queue.
- $Q_{i,i} = -\sum_{j \neq i} Q_{i,j}$ , the rate at which the cavity queue leaves state  $i$ .

For this transition matrix  $Q$ , one can compute the stationary distribution  $(\tilde{x}_k)_k$  where  $\tilde{x}_k$  represents the probability that the cavity queue has length  $k$  assuming the actual arrival rate  $(\lambda dp_k)_k$ . Indeed,  $\tilde{x}$  is given as the solution of  $\tilde{x}Q = 0$ . If  $\tilde{x}Q = 0$  is hard to solve, one can consider the embedded Discrete Time Markov Chain (DTMC) with transition matrix:

$$P = \frac{Q}{\max_i \{-Q_{i,i}\}} + I.$$

One finds that  $\tilde{x}$  satisfies the equality  $\tilde{x}P = \tilde{x}$ . To compute  $\tilde{x}$  we note that all columns of the matrix  $\bar{P} = \lim_{n \rightarrow \infty} P^n$  are equal and  $\tilde{x}$  is given as any row of  $\bar{P}$ . In order to quickly compute  $\bar{P}$ , one can use the recursion  $P_n = P_{n-1}^2$ . From the value  $\tilde{x}$  which satisfies  $\tilde{x}P = \tilde{x}$  we then compute  $\tilde{u}_k = \sum_{\ell \geq k} \tilde{x}_\ell$  and set  $T(u) = \tilde{u}$ . Note that if our guess was correct, we would have  $u = \tilde{u}$ . This yields the following numerical scheme:

1. Initiate some initial guess for the queue length distribution  $u^{(0)}$ , set  $n = 0$  and pick some tolerance  $tol$ .
2. Use the above method to compute  $u^{(n+1)} = T(u^{(n)})$ .
3. Increment  $n$  by one and return to step 2 if  $\sum_{k \geq 0} |u_k^{(n+1)} - u_k^{(n)}| > tol$ .

In the next section, we show how to apply this method to find the stationary distribution of SQ( $d$ ) with PH job sizes.

### 1.5.2 Phase Type Job Sizes

Let us consider the case where job sizes have some PH distribution rather than an exponential distribution. For a more elaborate review on PH distributions we refer the reader to Section 1.4. For this section we restrict ourselves to 2 dimensional PH distributions and it suffices to know that for this distribution any job is in phase one or phase two. When a job starts receiving service, it starts service in phase  $i$  with probability  $\alpha_i$ , while it receives service it transitions from phase  $i$  to phase  $j$  at rate  $A_{i,j}$  and it leaves

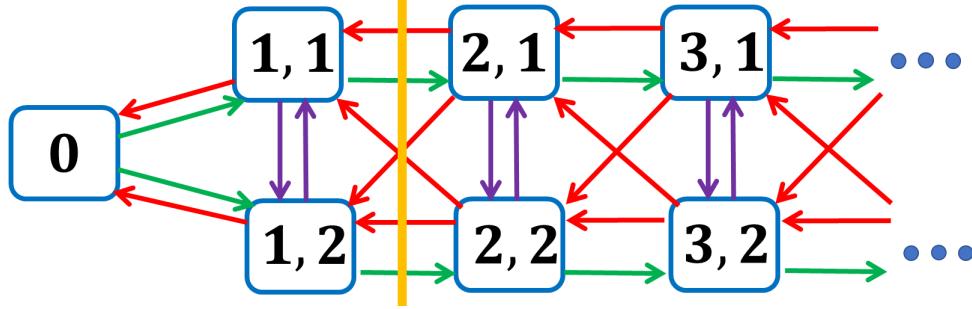


Figure 1.6: Scheme which illustrates the transitions for the  $\text{SQ}(d)$  policy with exponential job sizes with mean one.

the system with rate  $\mu_i$  when it is in phase  $i$ . Further we set  $A_{1,1} = -A_{1,2} - \mu_1$  and  $A_{2,2} = -A_{2,1} - \mu_2$  which denotes the rate at which the job leaves state 1 and 2.

For this model it is clear that it does not suffice to only keep track of the number of jobs present in each queue. We should also remember which state the job currently receiving service has. Therefore the state space is given by

$$S = \{0\} \cup \{(k, j) \mid k \geq 1, j \in \{1, 2\}\}.$$

We can again consider the associated diagram, which is shown in Figure 1.6. In this diagram one can start drawing orange lines and note that the rates crossing these lines in both ways should be equal. This would again lead to a series of equations. Unfortunately those equations (in most cases) do not result in a recursive formula. One could try to solve those equations using a FPI or some other numerical method which solves systems of nonlinear equations. However, we can set up the iterative scheme we introduced for exponential job sizes (this method was also used in [93] to analyse  $\text{SQ}(d)$  with PH job sizes). This appears to work better and generalizes well to other load balancing policies as we will see throughout the thesis (in particular the method used in Chapter 7 is based on this idea).

We explain the method in detail. First, we introduce some additional notation, we denote by  $x_{k,j}$  the probability that the cavity queue has  $k$  jobs and the job currently receiving service is in phase  $j$ . Further denote  $x_k = \sum_j x_{k,j}$  and  $u_k = \sum_{\ell \geq k} x_\ell$ . We note that the same diagram shown in Figure 1.5 still holds true for any job size distribution, that is, we have an actual arrival rate  $\lambda_k = \lambda d p_k$  to the cavity queue when it has length  $k$  (with  $p_k =$

$\frac{1}{2}(u_k - u_{k+1})$ ). We now assume that we have some initial guess for  $x$  and describe how to obtain a new (improved) guess  $\tilde{x}$ . Using our initial guess we can define the transition matrix of the cavity queue, for ease of notation we use a double index for the matrix elements. For example  $Q_{(k,1),(k,2)}$  denotes the transition rate of the job at the head of the queue from state 1 to state 2 when the cavity queue with length  $k$ . We only make an exception for the state 0, as there is no job at the head of the queue when the queue is empty. We have the following non-zero elements of the rate matrix  $Q$ :

- $Q_{0,(1,j)} = \lambda_0 \cdot \alpha_j$ , we have an arrival to an empty cavity queue with rate  $\lambda_0$  and the arriving job starts in phase  $j$  with probability  $\alpha_j$ .
- $Q_{(1,j),0} = \mu_j$ , the only job in the cavity queue completes service.
- $Q_{(k,j),(k+1,j)} = \lambda_k$  for  $k \geq 1$ , when an arrival occurs to the cavity queue with length  $k$ , it increases to length  $k + 1$  and the phase of the job at the head of the queue remains unchanged.
- $Q_{(k,j),(k,j')} = A_{j,j'}$  for  $k \geq 1$  and  $j \neq j'$ , the job at the head of the queue changes phase.
- $Q_{(k,j),(k-1,j')} = \mu_j \cdot \alpha_{j'}$  for  $k \geq 2$ . The job at the head of the queue finishes service and the new job starts in phase  $j'$ .
- $Q_{(k,j),(k,j)} = -\sum_{(k',j') \neq (k,j)} Q_{(k,j),(k',j')}$  for  $k \geq 1$ , the rate at which we leave state  $(k,j)$ .
- $Q_{0,0} = -\lambda_0$ , the rate at which the cavity queue stops being idle.

For this transition rate matrix we can define the embedded Markov chain represented by  $P = \frac{Q}{\max_i \{-Q_{i,i}\}} + I$ , we can use the same numerical method which we described in detail when we considered exponential job sizes. In the following section we formally define the cavity queue.

**Remark 2.** *We added a similar intuitive explanation for the LL( $d$ ) policy in Section 2.3 and also one for the Red( $d$ ) policy with identical replicas in Section 3.3.*

## 1.6 Formal definition of the cavity queue

In this section we give the formal definition of the cavity queue in the generality which is required throughout this thesis. On a first read, one may choose to skip this section as it may feel fabricated without having seen its applications.

For policies such as  $SQ(d)$  and  $LL(d)$ , where the job is simply assigned to one queue, we obtain an actual arrival rate which depends on either the queue length ( $k$ ) or workload ( $w$ ) at the time of the potential arrival (this value is denoted by  $\lambda_k$  resp.  $\lambda(w)$ ). For these type of policies, when a potential arrival is an actual arrival, the whole job will be assigned to the selected server. However there are also policies (e.g.  $Red(d)$ ) where only some part of the job is executed on the server. In general, we denote by  $U_1, \dots, U_d$  the random variables which represent the number of jobs resp. workload at the  $d$  randomly chosen servers and  $V_1, \dots, V_r$  random variables which are initiated upon a job arrival. Associated to these variables we define

$$\mathcal{Q}(U_1, \dots, U_d, V_1, \dots, V_r),$$

which denotes the amount of work (or queue length) at the queue with queue length/workload  $U_1$  after the potential arrival occurred (note that this is also a random variable). When  $U_2, \dots, U_d$  and  $V_1, \dots, V_r$  are clear from the context, we may leave them out of the notation.

**Example 1.** For  $SQ(d)$  with exponential job sizes with mean one, we do not require any additional variables  $V_i$  and set  $r = 0$ . The value  $U_i$  denotes the queue length on queue  $i$  and we have  $\mathcal{Q}(U_1, \dots, U_d) = U_1 + 1$  if  $U_1 < \min_{i=2}^d U_i$ , and  $\mathcal{Q}(U_1, \dots, U_d) = U_1 + 1$  with probability  $\frac{1}{j+1}$  if  $U_1 = \min_{i=2}^d U_i$  and there are exactly  $j$  values of  $i > 1$  for which  $U_i = U_1$ . In all other cases we have  $\mathcal{Q}(U_1, \dots, U_d) = U_1$ .

**Example 2.** For  $LL(d)$ , the value  $U_i$  denotes the amount of work left on queue  $i$ . We set  $r = 1$  and  $V_1 = X$  denotes the size of a single job. We have  $\mathcal{Q}(U_1, \dots, U_d, X) = U_1 + X$  if  $U_1 < \min_{i=2}^d U_i$ , and  $\mathcal{Q}(U_1, \dots, U_d, X) = U_1 + X$  with probability  $\frac{1}{j+1}$  if  $U_1 = 0$  and there are exactly  $j$  values of  $i > 1$  for which  $U_i = 0$ . In all other cases we have  $\mathcal{Q}(U_1, \dots, U_d, X) = U_1$ .

We are now in a position to give the general definition of the queue at the cavity for workload dependent load balancing policies:

**Definition 3** (Cavity Process of workload dependent policies). *Let  $\mathcal{H}(t)$ ,  $t \geq 0$ , be a set of probability measures on  $[0, \infty)$  called the environment process. The cavity process  $U^{\mathcal{H}(\cdot)}(t)$ ,  $t \geq 0$ , takes values in  $[0, \infty)$  and is defined as follows. Potential arrivals occur according to a Poisson process with rate  $\lambda d$ . When a potential arrival occurs at time  $t$ , the cavity process  $U^{\mathcal{H}(\cdot)}(t)$  becomes  $\mathcal{Q}(U^{\mathcal{H}(\cdot)}(t), U_2, \dots, U_d, V_1, \dots, V_r)$ . Here  $U_2, \dots, U_d$  are  $d-1$  independent random variables with law  $\mathcal{H}(t)$  and  $V_1, \dots, V_r$  are  $r$  random variables. The cavity process decreases at rate one during periods without arrivals and is lower bounded by zero.*

**Remark 3.** For queue length dependent processes, one replaces the state space  $[0, \infty)$  by  $\mathbb{N}^n$  (for some  $n \geq 1$ ). Moreover, instead of having the process decrease continuously at a rate equal to one, we have rates at which the process jumps. For exponential job sizes one typically has  $n = 1$  and the only jumps are those associated to job departures (from  $k$  to  $k-1$ ) and job arrivals (from  $k$  to  $k+1$ ).

Let us define the cavity process associated to the equilibrium environment process, which is such that the cavity process has distribution  $\mathcal{H}(t)$  at time  $t$ :

**Definition 4** (Equilibrium Environment). *When a cavity process  $U^{\mathcal{H}(\cdot)}(\cdot)$  has distribution  $\mathcal{H}(t)$  for all  $t \geq 0$ , we say that  $\mathcal{H}(\cdot)$  is an equilibrium environment process. Further, a probability measure  $\mathcal{H}$  is called an equilibrium environment if  $\mathcal{H}(t) = \mathcal{H}$  for all  $t$  and  $U^{\mathcal{H}(\cdot)}(t)$  has distribution  $\mathcal{H}$  for all  $t$ .*

**Remark 4.** Throughout this work, we always clearly specify the queue at the cavity which is used for our analysis.

The modularized program for analyzing load balancing systems presented in [20] when applied to our policies involves the following steps (assuming stability for large  $N$ ):

- a. **Asymptotic Independence.** Demonstrate  $\Pi^N \rightarrow \Pi$  as  $N \rightarrow \infty$ , where  $\Pi^N$  is the stationary distribution for the studied policy with  $N$  queues and  $\Pi$  is a stationary and ergodic distribution on  $[0, \infty)^\infty$ . Show that the limit  $\Pi$  is unique, depending only on the service time distribution. Show that, for every  $k$ :

$$\Pi^{(k)} = \bigotimes_{i=1}^k \Pi^{(1)},$$

where  $\Pi^{(k)}$  is  $\Pi$  restricted to its first  $k$  coordinates and  $\bigotimes_{i=1}^k \Pi^{(1)}$  denotes the product space.

- b. **The queue at the cavity.** Let  $\mathcal{B}_w^N$  resp.  $\mathcal{B}_k^N$  denote the arrival size distribution (which may be zero with a non-zero probability) in case of a potential arrival when the queue at the cavity has workload  $w$  resp. queue length  $k$ . Show that the arrival process of a queue in the system of size  $N$  converges to a Poisson process with rate  $\lambda d$  and a job size distribution  $\mathcal{B}_w$  resp.  $\mathcal{B}_k$  that depends on the workload  $w$  resp. queue length  $k$  at arrival time. Denote  $\mathcal{B} = \{\mathcal{B}_w, w \geq 0\}$ , resp.  $\mathcal{B} = \{\mathcal{B}_k, k \in \mathbb{N}\}$ .
- c. **Calculations.** Given  $\mathcal{B}$ , the arrival size distributions, analyse the queue at the cavity in the large  $N$  limit using queueing techniques to express  $\Pi^{(1)}$  as a function of  $\mathcal{B}$ :

$$\Pi^{(1)} = H_1(\mathcal{B}).$$

The arrival size distribution is determined by the workload distribution  $\Pi^{(1)}$  (as explained above) we thus have:

$$\mathcal{B} = H_2(\Pi^{(1)}).$$

We then must solve these two Fixed Point Equations (FPEs) to obtain the equilibrium environment  $\Pi^{(1)} = \mathcal{H}$ .

In this thesis, we mainly focus on **c**, the computational step of the program. We present numerical methods but also closed form solutions for the Equilibrium Environment  $\mathcal{H}$  corresponding to a variety of load balancing policies. We validate our results using simulation to show that the obtained solutions indeed correspond to the system under study (as  $N \rightarrow \infty$ ). We frequently make use of the following Conjecture :

**Conjecture 1.** *Consider a load balancing system operating under some power-of- $d$ -choices policy on  $N$  servers, assume  $\lambda, d$  and  $V_i$  are such that this system is uniformly stable for sufficiently large  $N$  and the local service is First Come First Served (FCFS). It follows that in the large  $N$  limit, there is a unique equilibrium distribution. Under this distribution, any finite number of queues are independent. Moreover, this equilibrium can be found as the unique fixed point in step **c**.*

**Remark 5.** Three notable exceptions for which this conjecture was proven are:

- Any convex combination of  $LL(d_j, K_j)$  policies, see [83].
- The  $Red(d)$  policy with exponential job sizes and i.i.d. replicas, see [83].
- The  $SQ(d)$  policy with DHR job sizes (see [21]).

For queue length dependent load balancing methods with exponential job sizes, there exists a *standard method* to show Conjecture 1, one takes the following steps :

- (1) Find a set of Ordinary Differential Equations (ODEs) which describes the transient behaviour of the cavity queue. In particular, defining  $u_k(t)$  as the probability that the server has  $k$  or more jobs at time  $t$ , one finds that  $(u_k(t))_k$  satisfies some set of ODEs:

$$\frac{du_k(t)}{dt} = T_{k,\lambda,d}((u_\ell(t))_\ell), k = 0, 1, \dots$$

- (2) Show that this set of ODEs has a unique fixed point. That is, there is a unique, decreasing sequence  $1 = u_0 \geq u_1 \geq u_2 \searrow 0$  for which:

$$0 = T_{k,\lambda,d}((u_\ell)_\ell), \ell = 0, 1, \dots$$

- (3) Prove the convergence of the  $N$  dimensional stochastic process over finite time scales, that is, for any fixed  $\mathcal{T} \in (0, \infty)$  one shows that (as  $N$  tends to infinity) the stochastic processes on  $[0, \mathcal{T}]$  converge towards the solution of the system of ODEs. For this step, one often relies on Kurtz's theorem (see [31, 67]) or the convergence of transition semigroups of Markov processes [1, 95].
- (4) Show that the  $N$  dimensional processes all have a stationary measure and that this sequence of stationary measures has a limit point, which follows from the tightness of the stationary measures.
- (5) Finally, one shows that the fixed point is a global attractor and that the limit point of the stationary measures must be the Dirac measure associated to the fixed point.

These steps are quite standard within the field of mean field theory, however showing that the fixed point is a global attractor often proves to be difficult to check. In [90], this methodology is explained and generalized to the case of hyperexponential job sizes. For this method, one relies on the fact that the fixed point is discrete, that is, it is a sequence defined on  $\mathbb{N}$ . Moreover, the proof simplifies significantly when one artificially bounds the queues, that way the stationary distribution becomes a distribution on some finite set.

## 1.7 Supplementary material

All algorithms derived in this thesis have also been implemented in Matlab, one can find them on the github page <https://github.com/THellemans/thesis>. With this code, it is possible to generate all figures and tables which are included in this thesis. Here, one can also find the videos we made for a better understanding of the text.

## Chapter 2

# The power of $d$ choices using least loaded server selection

In order to solve this differential equation you look at it until a solution occurs to you.

---

George Pólya

### Abstract

This chapter is based on our work on the LL( $d$ ) policy which can be found at [47].

Motivated by distributed schedulers that combine the power-of-d-choices with late binding (see also [73]) and systems that use replication with cancellation on start, we study the performance of the LL( $d$ ) policy which assigns a job to a server that currently has the least workload among  $d$  randomly selected servers in large-scale homogeneous clusters.

We consider a general job size distributions and propose a partial integro differential equation to describe the evolution of the system. This equation relies on Conjecture 1, which was proven in [21] (and later also in [83]) for the LL( $d$ ) policy. Based on this equation we propose a FPI for the limiting workload distribution and study its convergence.

For exponential job sizes we present a simple closed form expression for the limiting workload distribution that is valid for any work-conserving service discipline as well as for the limiting response time distribution in case

of FCFS scheduling. We further show that for PH distributed job sizes the limiting workload and response time distribution can be expressed via the unique solution of a simple set of ODEs. Numerical and analytical results that compare response time of the classic power-of- $d$ -choices algorithm and the LL( $d$ ) policy are also presented and the accuracy of the limiting response time distribution for finite systems is illustrated using simulations.

The LL( $d$ ) policy can be seen as the most simple example of a workload dependent load balancing policy, therefore understanding its analysis is paramount to having a good understanding of the remaining of the text. In Chapter 4 we generalize the analysis of the LL( $d$ ) policy to many other (more involved) policies. In Chapter 5 we analyse the LL( $d$ ) policy when the system tends towards instability, Chapter 6 considers the LL( $d$ ) policy with memory at the dispatcher.

## 2.1 Introduction

Load balancing plays a crucial role in achieving low latency in large-scale clusters. A simple randomized approach, denoted as SQ( $d$ ), exists in assigning incoming jobs to a server that currently holds the fewest number of jobs among a set of  $d$  randomly selected servers, the so-called *power-of- $d$ -choices* algorithm, see also Section 1.5 and [95, 67, 71, 4]. While this approach yields short queues with high probability in case of first-come-first-served (FCFS) scheduling even for general job size distributions provided that  $d$  is chosen sufficiently large [20, 23], short queues do not guarantee low latency as the queue length is only a coarse indicator of the waiting time in the presence of high job size variability. The main issue is that under the FCFS discipline short jobs can get stuck behind a single long job which significantly increases the short job's latency. In addition, when multiple dispatchers are used to distribute the jobs, *race* conditions may occur where multiple schedulers concurrently place jobs on a server that appears lightly loaded [64].

To avoid these issues the notion of *late binding* was recently introduced in [74]. With late binding the dispatcher still probes  $d$  servers at random, but the servers do not immediately reply by sending their queue length information. Instead they place a reservation at the end of a local work queue and when the reservation reaches the front of the queue, the server requests the job associated to the reservation from the dispatcher. In this manner the job is assigned to the server that is able to launch the job the soonest

among the  $d$  randomly selected servers. The downside of late binding is that the server always experiences some idle time in between the execution of two jobs, which implies some efficiency loss. However, whenever the network latencies are much smaller than the shortest job runtimes (and the system load is not extremely high), experiments on a 110-machine cluster show that a scheduler that relies on late binding performs close to an ideal scheduler [74].

Note that late binding as described above is equivalent to assigning the job to the server that has the least workload among  $d$  randomly selected servers, which is known as the LL( $d$ ) policy [21], provided that the network latencies are negligible<sup>1</sup>.

The main objective of this chapter is to study the large-scale limit of the server workload and response time distribution of the LL( $d$ ) policy when employed on a homogeneous cluster subject to Poisson job arrivals with general service times. For this purpose we introduce a Partial Integro Differential Equation (PIDE) that captures the evolution of the so-called *cavity process* and study its equilibrium. The key observation, established in [21] (and more recently also in [83]), is that under the LL( $d$ ) policy with general service time distributions, the workload distribution of any finite set of servers becomes asymptotically independent as the number of servers tends to infinity (provided that all the servers employ the same local non-idling service discipline, e.g., FCFS, Processor Sharing (PS), etc.). Moreover, the limit of the marginal workload distribution of a server corresponds to the unique equilibrium environment.

It is worth noting that the LL( $d$ ) policy is equivalent to the following system that uses replication with cancellation on start to reduce waiting times. Arriving jobs are replicated  $d$  times and are randomly assigned to  $d$  servers (that all operate in FCFS order). As soon as a single replica starts execution on a server, the remaining  $d - 1$  replicas are killed (with the additional assumption that if multiple replicas start at exactly the same time, only one is executed). Prior work on replication was mainly done in the context of systems that experience server slowdown and therefore focused on replication with cancellation on job completion [40, 39], which is considerably different from LL( $d$ ) as jobs are often (partially) executed on multiple servers in such

---

<sup>1</sup>When the network latencies are not negligible compared to the job runtimes, we can regard them as part of the workload of a job such that the job execution consists of two parts: fetching the job and executing it, see Section 2.9.3.

case.

Another reason for studying the large-scale limit of the LL( $d$ ) policy exists in understanding how much benefit precise workload information gives in comparison to the coarser queue length information used by SQ( $d$ ).

The main contributions of this chapter are as follows:

1. A PIDE to describe the transient evolution of the limiting workload of a server under the LL( $d$ ) policy is derived.
2. An FPE for the limiting stationary workload distribution is presented together with an FPI to compute its solution. Convergence of the FPI is proven for system loads below  $e^{-1/e} \approx 0.6922$ . Moreover, we also introduce a DIDE which is equivalent to the suggested FPE.
3. A simple explicit solution for the limiting workload and response time distribution is presented in case of exponential job sizes. For PH distributed job sizes we prove that the limiting workload distribution can be computed easily by solving a simple set of ODEs.
4. We present both analytical and numerical results that compare the response time of the LL( $d$ ) policy with the classic SQ( $d$ ) policy. These results illustrate that late binding offers a significant reduction in the response time under a wide range of loads even when taking the idleness caused by late binding into account.

## 2.2 Structure of the chapter

The chapter is structured as follows. An informal, intuitive summary of the chapter is given in Section 2.3. The model considered in this chapter is formally described in Section 2.4. The PIDE that captures the transient evolution of the workload is introduced in Section 2.5, while the integral equation for the limiting stationary workload and its associated FPE are presented in Section 2.6. Sections 2.7 and 2.8 discuss the special cases of exponential and PH distributed job sizes, respectively. Section 2.9 compares the performance of the LL( $d$ ) and SQ( $d$ ) policies, while Section 2.10 briefly studies the accuracy of the limiting distributions for systems of finite size. Conclusions are drawn and future work is suggested in Section 2.11.

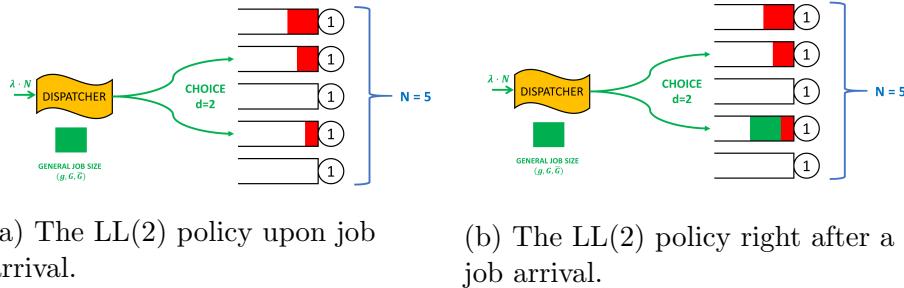


Figure 2.1: Graphical depiction of the LL(2) policy with  $N = 5$ , exponential job sizes with mean one and arrival rate  $\lambda N$ .

## 2.3 Informal intuition

### 2.3.1 General job sizes

In this section we give an informal description which allows anyone to be able to understand how the analysis of the LL( $d$ ) policy can be done without going into much detail of the proofs. We also included a video explaining the contents of this section at our github page. For the LL( $d$ ) policy, upon a job arrival, the dispatcher selects  $d$  random servers and assigns the incoming job to the server which has the least amount of work left in its queue. In the same way as Section 1.5 went along the same lines as Appendix A.1, this section goes along the same lines as Appendix A.4.

While the SQ( $d$ ) policy typically assumes exponential job sizes, we do not put any restrictions on the job size distribution for the LL( $d$ ) policy (except, for simplicity, that its mean is one). All one needs to keep track of for every queue is the time it takes until it has completed all of its present work. Jobs arrive as a Poisson process with rate  $\lambda \cdot N$ , which means that the time between 2 arrivals is (on average)  $\frac{1}{\lambda N}$ , and the time you have already been waiting for the next arrival does not impact the time until the next arrival occurs. Whenever a job arrives,  $d$  servers are chosen at random and the job is assigned to the server which has the least amount of work left in its queue. One would then like to answer the question, when you are a job which arrives to this system, *what's the probability that you will have to wait for  $w$  or more time until you start receiving service?* In what follows, we show that this probability can simply be computed as  $\bar{F}(w)^d$  with  $\bar{F}(w)$  the probability that the *queue at the cavity* has  $w$  or more work, moreover we

find that  $\bar{F}(w)$  can be computed from the DIDE:

$$\bar{F}'(w) = -\lambda \cdot \left[ \bar{G}(w) - \bar{F}(w)^d + \int_0^w \bar{F}(u)^d g(w-u) du \right].$$

with  $g$  and  $\bar{G}$  the pdf and ccdf of the job size distribution. From this, it is not hard to compute the waiting time distribution and the mean waiting time.

Thinking about the model, it is clear that all we really need to know is the amount of work present at each queue. In Figure 2.1 we show what happens at an arrival instant for the LL(2) policy. In order to numerically analyse this model, we would discretize the time it takes for the queue to finish all present work. We would then describe the system in Figure 1.3a as e.g.  $(2.5, 1.8, 0, 0.6, 0)$ , and the arrival is then simply a transition to the state  $(2.5, 1.8, 0, 4.5, 0)$ . One could attempt to compute the stationary distribution of this Markov chain, but seeing that the number of states is equal to  $M^N$  (with  $M$  the number of points used to discretize the queue's workload), this becomes prohibitively large as the value of  $N$  grows large. Even for  $N = 2$  this model is only tractable for small values of  $\lambda$ .

When  $N = d = 2$ , it is obvious that when the servers are not well balanced, all subsequent arrivals will be routed to the one with the least work left, balancing out the loads. This incurs that in general the workloads will have a strong positive correlation. However, as  $N$  grows large, the probability of selecting 2 specific servers becomes negligible which makes the workloads independent. Moreover, as all queues are treated in the same way, one finds that the workload distribution is identical for all servers. Therefore we can hide the whole system except for one cavity where we can still view exactly one queue, namely the *queue at the cavity*. We observe that queue and assume that all other queues are independent of this queue and have the same workload distribution. For this cavity queue, we may denote by  $f$ ,  $F$  and  $\bar{F}$ , its pdf, cdf and ccdf. We therefore have that  $f(w)$  represents the density that the workload is exactly  $w$ ,  $F(w)$  the probability that its workload does not exceed  $w$  and  $\bar{F}(w)$  the probability that its workload does exceed  $w$ . We compute the distribution through the computation of  $\bar{F}(w)$ . Further on, we set  $d = 2$  for simplicity.

In order to compute  $\bar{F}(w)$ , we use a level crossing argument, that is: *what goes up, must come down!* We note that the event *the cavity queue's load decreases below  $w$*  should occur the same number of times as the event *the*

cavity queue's load increases above  $w$ . As all servers work at a constant rate equal to one, the down-crossing rate through  $w$  is given by  $f(w) = -\bar{F}'(w)$ .

For the up-crossing rate, we note that the arrival rate is  $\lambda \cdot N$  and the probability that the arrival occurs at the cavity queue is equal to  $2/N$ , therefore the *potential arrival rate* is equal to  $\lambda N \cdot 2/N = \lambda 2$ . When the queue at the cavity is selected for a potential arrival, we know that some second queue (which we cannot see) is selected, we assume that the other queue has the same workload distribution as the queue at the cavity, therefore it has  $u$  or more work with probability  $\bar{F}(u)$ . We consider 2 separate cases, namely the case where the cavity queue is empty and the one where the cavity queue has some workload  $u \in (0, w)$ . We find:

1. When the cavity queue is empty, the second (hidden) queue is either non-empty (which happens with probability  $\bar{F}(0)$ ), in this case the cavity queue is sure to receive the incoming job. The cavity queue crosses the boundary  $w$  in case the job has a size greater than  $w$ , which happens with probability  $\bar{G}(w)$ . When the other queue is empty (with probability  $F(0)$ ), the cavity queue receives the job with probability  $\frac{1}{2}$ . The cavity queue then crosses the boundary  $w$  with probability  $\bar{G}(w)$ .
2. When the cavity queue is non-empty, but has some load  $u \in (0, w)$ , the cavity queue will only receive the incoming job when the other selected queue has more than  $u$  work, which happens with probability  $\bar{F}(u)$ . In this case, the cavity queue breaches the load  $w$  if the job size exceeds  $w - u$ .

In total, we find that the probability of having a crossover when a potential arrival occurs is given by:

$$F(0) \cdot \bar{F}(0)\bar{G}(w) + \frac{1}{2}F(0)^2\bar{G}(w) + \int_0^w f(u)\bar{F}(u)\bar{G}(w-u) du.$$

We know that the up-crossing rate should be equal to the down-crossing rate. As the potential arrival rate is given by  $\lambda 2$ , and the down-crossing rate is  $-\bar{F}'(0)$ , we find that  $\bar{F}(0)$  satisfies the DIDE:

$$\bar{F}'(w) = -\lambda 2 \cdot \left[ F(0) \cdot \bar{F}(0)\bar{G}(w) + \frac{1}{2}F(0)^2\bar{G}(w) + \int_0^w f(u)\bar{F}(u)\bar{G}(w-u) du \right].$$

The main issue with obtaining  $\bar{F}(w)$  from this equation is the fact that we have occurrences of  $-\bar{F}'(w) = f(w)$  in the right hand side making it numerically challenging to obtain  $\bar{F}(w)$ . We should therefore try to get rid

of  $f(w)$ , this can easily be done by applying integration by parts (differentiating  $\bar{G}(w - u)$  and integrating  $f(u)\bar{F}(u) du$ ). Some elementary calculus and applying the suggested integration by parts on the integral allows one to conclude that the DIDE reduces to:

$$\bar{F}'(w) = -\lambda \cdot \left[ \bar{G}(w) - \bar{F}(w)^2 + \int_0^w \bar{F}(u)^2 g(w-u) du \right]. \quad (2.1)$$

This equation can easily be solved numerically by iteratively setting :

$$\bar{F}(w + \Delta) = \bar{F}(w) - \lambda \Delta \cdot \left[ \bar{G}(w) - \bar{F}(w)^2 + \int_0^w \bar{F}(u)^2 g(w-u) du \right].$$

The only thing we still require is a boundary condition for  $\bar{F}(0)$ . The amount of work coming into the system is  $\lambda \cdot N$ , while the amount of work being completed per time unit is given by:

$$\mathbb{P}\{\text{Queue working}\} \cdot N \cdot 1 = \bar{F}(0) \cdot N.$$

When the system is stable, the amount of incoming work should be equal to the amount of outgoing work, therefore we have:

$$\bar{F}(0)N = \lambda N \Rightarrow \bar{F}(0) = \lambda.$$

We find that (2.1) easily generalizes to:

$$\bar{F}'(w) = -\lambda \cdot \left[ \bar{G}(w) - \bar{F}(w)^d + \int_0^w \bar{F}(u)^d g(w-u) du \right]. \quad (2.2)$$

After solving the above equation using a forward Euler scheme, we can answer the original question, we have to wait for  $w$  or more time until we start receiving service if all  $d$  servers we selected have  $w$  or more work, which happens with probability  $\bar{F}(w)^d$ .

### 2.3.2 Exponential job sizes

It turns out that when job sizes are exponential (with mean one), the DIDE (2.2) can be simplified to the ODE  $\bar{F}'(w) = \lambda \bar{F}(w)^d - \bar{F}(w)$ , which can be solved explicitly. From this, one obtains the simple expression for the expected waiting time  $-\frac{\log(1-\lambda^2)}{\lambda}$  when  $d = 2$ , for arbitrary  $d \geq 2$  we find  $\mathbb{E}[W] = \sum_{n=1}^{\infty} \frac{\lambda^{dn}}{1+n(d-1)}$ .

### 2.3.3 Phase type job sizes

When the job size distribution is Phase Type (see also Section 1.4), one finds that there exist: an  $n \times n$  matrix  $A$ , a  $1 \times n$  matrix  $\alpha$  and some  $n \times 1$  vector  $\mu$  such that  $g(w) = \alpha e^{Aw} \mu$ . Therefore, (2.2) can be written as:

$$\bar{F}'(w) = -\lambda \cdot \left[ \bar{G}(w) - \bar{F}(w)^d + \alpha \int_0^w \bar{F}(u)^d e^{A(w-u)} \mu du \right].$$

One may then define  $\xi(w) = \int_0^w \bar{F}(u)^d e^{A(w-u)} \mu du$  and computes:

$$\xi'(w) = \bar{F}(w)^d + A\xi(w).$$

Therefore the above DIDE reduces to the system of ODEs:

$$\begin{aligned} \bar{F}'(w) &= -\lambda \cdot [\bar{G}(w) - \bar{F}(w)^d + \alpha \xi(w)], \\ \xi'(w) &= \bar{F}(w)^d + A\xi(w). \end{aligned}$$

This system of differential equations can be solved significantly faster than the equivalent DIDE. A similarly simple Delayed Differential Equation (DDE) is available when the job size distribution is a mix of a discrete and a PH random variable.

## 2.4 Model description

We consider a system consisting of  $N$  single server queues each having an infinite waiting room. Arrivals occur into the system as a Poisson process with rate  $\lambda N$ . For each incoming job,  $d$  queues are selected uniformly at random and the job joins the queue that currently holds the least workload with ties being broken uniformly at random. The service discipline is such that the workload at any queue reduces at rate 1 when positive, that is, we do not put any restriction on the service discipline apart from the fact that it is non-idling and identical in each server (unless stated otherwise). The workload offered by a job has a general distribution (as discussed in Section 1.4) with cdf  $G(\cdot)$ , pdf  $g(\cdot)$ , mean  $\mathbb{E}[G]$  and is such that  $G(0) = 0$ . We define  $\rho = \lambda \mathbb{E}[G]$  and assume that  $\rho < 1$ .

The above model corresponds to the so-called least-loaded supermarket model, denoted as LL( $d$ ) in [20, 21]. Note that the corresponding Markov process that keeps track of the workloads of the  $N$  queues is positive Harris

recurrent and has a unique stationary probability measure  $\mathcal{E}^{(N)}$  whenever the queueing system is subcritical, that is, when  $\rho < 1$ , as noted at the end of Section 5 in [19]. In fact, this result is a special case of [33, Theorem 2.5].

## 2.5 Cavity process

We start by introducing the cavity process from [20] for the LL( $d$ ) supermarket model. The process is intended to capture the evolution of the workload of a single queue for the limiting system where the number of servers  $N$  tends to infinity.

**Definition 5** (LL( $d$ ) cavity process). *Let  $\mathcal{H}(t)$ ,  $t \geq 0$ , be a set of probability measures on  $[0, \infty)$  called the environment process. The cavity process  $U^{\mathcal{H}(\cdot)}(t)$ ,  $t \geq 0$ , takes values in  $[0, \infty)$  and is defined as follows. Potential arrivals occur according to a Poisson process with rate  $\lambda d$ . When a potential arrival occurs at time  $t$ , we compare the state  $U^{\mathcal{H}(\cdot)}(t-)$  just prior to time  $t$  with the states of  $d - 1$  independent random variables with law  $\mathcal{H}(t)$ . The potential incoming job is assigned to the state among these  $d$  states that has the lowest value, where ties are broken uniformly at random. If the job is assigned to state  $U^{\mathcal{H}(\cdot)}(t-)$ , we immediately add the job to the queue, that is,  $U^{\mathcal{H}(\cdot)}(t) = U^{\mathcal{H}(\cdot)}(t-) + x$  where  $x$  is the size of the incoming job. Otherwise, the job immediately leaves the system, i.e.,  $U^{\mathcal{H}(\cdot)}(t) = U^{\mathcal{H}(\cdot)}(t-)$ . Clearly, if  $U^{\mathcal{H}(\cdot)}(t-)$  has law  $\mathcal{H}(t)$  a potential arrival at time  $t$  joins the queue with probability  $1/d$ . Finally, the cavity process decreases at rate one during periods without arrivals and is lower bounded by zero.*

**Definition 6** (Equilibrium Environment). *When a cavity process  $U^{\mathcal{H}(\cdot)}(\cdot)$  has distribution  $\mathcal{H}(t)$  for all  $t \geq 0$ , we say that  $\mathcal{H}(\cdot)$  is an equilibrium environment process. Further, a probability measure  $\mathcal{H}$  is called an equilibrium environment if  $\mathcal{H}(t) = \mathcal{H}$  for all  $t$  and  $U^{\mathcal{H}(\cdot)}(t)$  has distribution  $\mathcal{H}$  for all  $t$ .*

**Theorem 7** (due to Theorem 2.2 of [21]). *Consider the LL( $d$ ) supermarket model with  $N$  queues, general service times (with mean  $\mathbb{E}[G]$ ), Poisson arrivals with rate  $\lambda N < N/\mathbb{E}[G]$  and an identical non-idling service discipline at each queue. Let  $\mathcal{E}^{(N,N')}$  be the projection of the stationary measure  $\mathcal{E}^{(N)}$  of the  $N$  workloads into the workloads of the first  $N'$  queues, then  $\mathcal{E}^{(N,N')}$  converges in total variation to the  $N'$ -fold convolution of  $\mathcal{E}^{(\infty,1)}$  (in an appropriate metric space) as  $N$  tends to infinity. Moreover,  $\mathcal{E}^{(\infty,1)}$  is the unique equilibrium environment of the LL( $d$ ) supermarket model.*

In other words the above theorem indicates that the workload distributions of any finite set of  $N'$  queues becomes asymptotically independent as  $N$  tends to infinity and the marginal workload distribution of any queue is given by the *unique* equilibrium environment  $\mathcal{H}$  of the LL( $d$ ) supermarket model. Thus, there is no need to interchange the limits when relying on Theorem 7 when studying the limit of the stationary distributions. We do note that the order of these limits can in fact be reversed (when starting from an empty system) as shown in [21, Section 8]. In particular this provides a proof for the result of Conjecture 1. More recently, it was also proven in [83] that Conjecture 1 indeed holds for the LL( $d$ ) policy.

We now characterize the evolution of the cavity process associated with the equilibrium environment process  $\mathcal{H}(\cdot)$  of the LL( $d$ ) supermarket model.

Let  $f(t, w)$  for  $w \in (0, \infty)$  describe the density of servers which, at time  $t$ , have workload  $w$ . Note that  $f(t, \cdot)$  is not an actual pdf as some of the servers may be idle, denote  $F(t, 0) = 1 - \int_0^\infty f(t, w)dw$ . In the following we refer to  $f(t, \cdot)$  as a density, and we define its cdf  $F(t, \cdot)$  as  $F(t, w) = F(t, 0) + \int_0^w f(t, u)du$ . Moreover, we define the ccdf as  $\bar{F}(t, w) = 1 - F(t, w)$ .

For any  $d \in \{2, 3, \dots\}$ , we define the function  $c_d(t, u)$  as the density at which a potential arrival at time  $t$  joins the cavity queue with workload  $u > 0$ . By definition of the cavity process associated to the equilibrium environment, this density is given by:

$$c_d(t, u) = f(t, u)(1 - F(t, u))^{d-1} = f(t, u)\bar{F}(t, u)^{d-1}, \quad (2.3)$$

We further denote the probability that a potential arrival at time  $t$  joins the cavity queue with workload at most  $u$  by  $C_d(t, u)$ . In this case we have, as ties are broken uniformly at random:

$$\begin{aligned} C_d(t, u) &= F(t, 0) \sum_{k=0}^{d-1} \binom{d-1}{k} \frac{F(t, 0)^k \bar{F}(t, 0)^{d-1-k}}{k+1} + \int_{v=0}^u c_d(t, v)dv \\ &= \frac{1 - \bar{F}(t, 0)^d}{d} + \int_{v=0}^u c_d(t, v)dv = \frac{1 - \bar{F}(t, u)^d}{d}. \end{aligned} \quad (2.4)$$

In particular,  $C_d(t, 0)$  is the probability that a potential arrival joins an empty cavity queue.

**Theorem 8.** *The evolution of the cavity process associated to the equilibrium environment process of the LL( $d$ ) supermarket model is captured by the*

following PIDE:

$$\frac{\partial f(t, w)}{\partial t} - \frac{\partial f(t, w)}{\partial w} = \lambda d \int_0^w c_d(t, u) g(w - u) du + \lambda d C_d(t, 0) g(w) - \lambda d c_d(t, w) \quad (2.5)$$

$$\frac{\partial F(t, 0)}{\partial t} = f(t, 0^+) - \lambda d C_d(t, 0), \quad (2.6)$$

for  $w > 0$ , where  $f(x, z^+) = \lim_{y \rightarrow z^+} f(x, y)$ .

*Proof.* Assume  $w > 0$  and let  $w > \Delta > 0$  be arbitrary. In order to have a workload of  $w$  at time  $t + \Delta$  we need to consider three possible cases: no arrivals in  $[t, t + \Delta]$ , an arrival occurs in  $[t, t + \Delta]$  when the workload is non-zero and an arrival occurs in an idle server in  $[t, t + \Delta]$ . Hence, we can write

$$f(t + \Delta, w) = Q_1 + Q_2 + Q_3 + o(\Delta). \quad (2.7)$$

The terms  $Q_i$ , for  $i = 1, 2$  and  $3$  are graphically illustrated in Figure 2.2 discussed next.

( $Q_1$ ) No arrivals in the interval  $[t, t + \Delta]$ : if the cavity queue at time  $t$  has a workload exactly equal to  $w + \Delta$  and experiences no arrivals in  $[t, t + \Delta]$ , it will have a workload equal to  $w$  at time  $t + \Delta$ . The density of having a workload  $w + \Delta$  at time  $t$  is given by  $f(t, w + \Delta)$  and the density at which an arrival occurs at the cavity queue at time  $t + u$ ,  $u \in [0, \Delta]$ , when it has workload  $w + \Delta - u$ , is equal to  $\lambda d c_d(t + u, w + \Delta - u)$ . Therefore we find:

$$Q_1 = f(t, w + \Delta) - \lambda d \int_{u=0}^{\Delta} c_d(t + u, w + \Delta - u) du.$$

( $Q_2$ ) A single arrival occurs when the cavity queue is not idle: in this case at some time  $t + v$ ,  $v \in [0, \Delta]$  an arrival of size  $w + \Delta - v$  occurs at the cavity queue which has workload  $u - v$  for some  $u \in [v, w + \Delta]$  occurs. We find:

$$Q_2 = \lambda d \int_{v=0}^{\Delta} \int_{u=v}^{w+\Delta} c_d(t + v, u - v) g(w - u + \Delta - v) du dv.$$

( $Q_3$ ) A single arrival occurs when the cavity queue is empty: in this case a job of size  $w + \Delta - u$  arrives at time  $t + u$  for some  $u \in [0, \Delta]$ . Hence,

$$Q_3 = \lambda d \int_{u=0}^{\Delta} C_d(t + u, 0) g(w + \Delta - u) du.$$

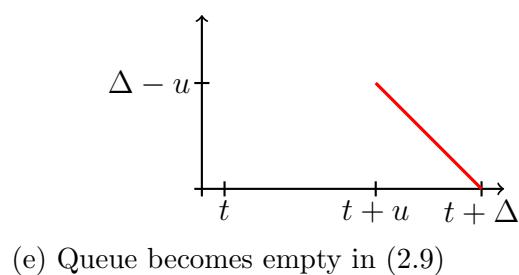
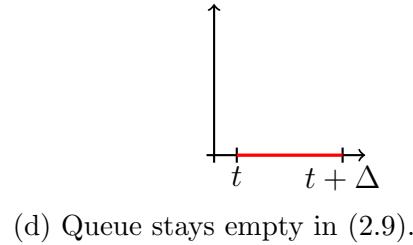
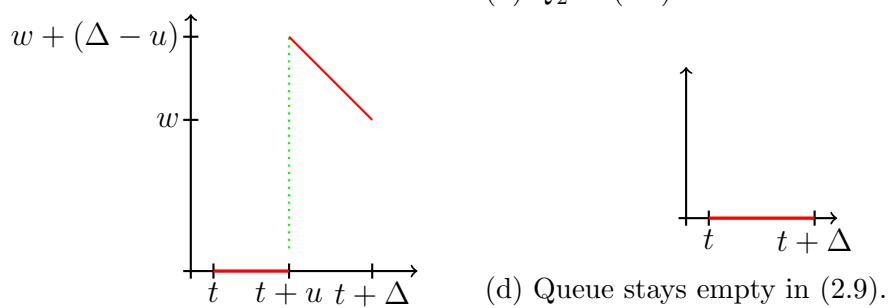
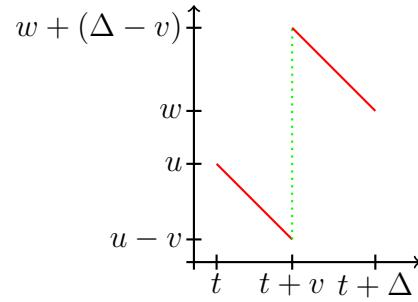
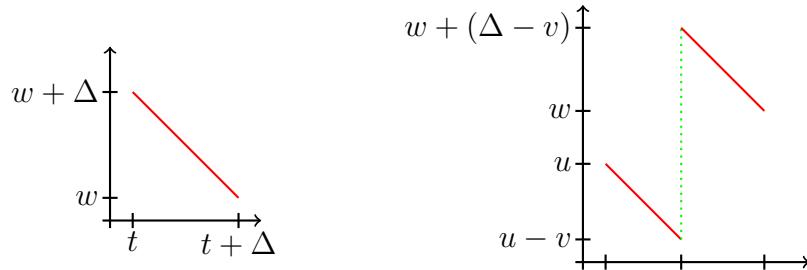


Figure 2.2: Graphical illustration for the proof of Theorem 8

By subtracting  $f(t, w + \Delta)$ , dividing by  $\Delta$  and letting  $\Delta$  decrease to zero, we find (2.5) from (2.7).

We still require a differential equation for  $F(t, 0)$ , a server may be idle at time  $t + \Delta$  by remaining idle in  $[t, t + \Delta]$  or having a workload equal to  $\Delta - u, u < \Delta$  at time  $t + u$ . We therefore find:

$$F(t + \Delta, 0) = F(t, 0) - \lambda d \int_{u=0}^{\Delta} C_d(t + u, 0) du \quad (2.8)$$

$$+ \int_{u=0}^{\Delta} f(t + u, \Delta - u) du + o(\Delta), \quad (2.9)$$

subtracting  $F(t, 0)$ , dividing by  $\Delta$  and letting  $\Delta$  tend to zero yields (2.6).  $\square$

**Remark 6.** *The PIDE given by (2.5-2.6) can be solved numerically using the following scheme:*

$$\begin{aligned} f(t + \Delta, 0^+) &= \lambda d C_d(t, 0), \\ f(t + \Delta, w) &= f(t, w + \Delta) + \lambda d \Delta \int_0^w c_d(t, u) g(w - u) du \\ &\quad + \lambda d \Delta C_d(t, 0) g(w) - \lambda d \Delta c_d(t, w), \end{aligned}$$

for  $w \geq \Delta$ . As a boundary condition, we may impose that we start with all servers being idle, i.e., for  $w > 0$  we set  $f(0, w) = 0$  and  $F(0, 0) = 1$ . The main objective of Theorem 8 is however to use it to characterize the unique equilibrium environment  $\mathcal{H}$  of the  $LL(d)$  policy.

**Remark 7.** A propagation of chaos result was established in [21, Section 7] (and more recently also in [83]) which states that the workload distributions of any set of  $N' \leq N$  queues at time  $t$  become asymptotically independent, provided that the workload distribution at time 0 in the  $N$  queues is i.i.d. and does not depend on  $N$ , e.g., if the system is empty at time zero. As such Theorem 8 characterizes the limiting transient behavior.

## 2.6 Limiting workload distribution

As indicated in the previous section, the limiting stationary workload distribution is given by the unique equilibrium environment. Let  $F(w)$  be the cdf of the workload distribution, that is,  $F(w)$  represents the probability that

the workload is at most  $w$ , let  $f(w)$  be its density for  $w > 0$  and denote its ccdf by  $\bar{F}(w) = 1 - F(w)$ . Furthermore, similar to (2.3) and (2.4), define:

$$c_d(u) = f(u)\bar{F}(u)^{d-1}, \quad (2.10)$$

and

$$C_d(u) = \frac{1 - \bar{F}(u)^d}{d}. \quad (2.11)$$

**Theorem 9.** *The stationary workload distribution is the unique distribution for which the ccdf obeys the following FPE:*

$$\bar{F}(w) = \rho - \lambda \cdot \left( \int_0^w (1 - \bar{F}(u)^d) \bar{G}(w-u) du \right) \quad (2.12)$$

*Proof.* By demanding that the derivatives with respect to  $t$  are zero in (2.5-2.6), we find

$$\frac{\partial f(w)}{\partial w} = \lambda d \left( c_d(w) - \int_0^w c_d(u) g(w-u) du - C_d(0) g(w) \right), \quad (2.13)$$

and

$$f(0^+) = \lambda d C_d(0). \quad (2.14)$$

Integrating (2.13) once (and relying on the assumption that  $G(0) = 0$ ) we find:

$$f(w) = K - \lambda d \cdot \left( \frac{1}{d} - C_d(w) + C_d(0)G(w) + \int_0^w c_d(u)G(w-u) du \right), \quad (2.15)$$

for an appropriate constant  $K$ . As we know from (2.14) that  $f(0^+) = \lambda d C_d(0)$ , we see that we should set  $K$  equal to  $\lambda$ . We may therefore conclude that

$$f(w) = \lambda d \cdot \left( C_d(w) - C_d(0)G(w) - \int_0^w c_d(u)G(w-u) du \right) \quad (2.16)$$

Integrating equation (2.16) once more and using the fact that  $F(0) = 1 - \rho$ , yields

$$F(w) = (1 - \rho) + \lambda d \cdot \left( \int_0^w C_d(u)(1 - G(w-u)) du \right),$$

from which (2.12) easily follows. The uniqueness follows from the fact that there exists a unique equilibrium environment for the LL( $d$ ) supermarket model as stated earlier.  $\square$

**Remark 8.** For  $d$  large, we have  $\bar{F}(u)^d \approx 0$  and therefore (2.12) yields

$$\bar{F}(w) \approx \rho - \lambda \int_0^w \bar{G}(u) du = \lambda \int_w^\infty \bar{G}(u) du,$$

meaning  $f(w) \approx \lambda \bar{G}(w)$ . Note that  $\lambda \int_0^w \bar{G}(u) du$  can be recognized as the equilibrium distribution of the forward recurrence time of a renewal process with inter-arrival time distribution  $G$ .

**Remark 9.** The cavity process evolves as the workload of an  $M/G/1$  queue with a workload dependent arrival rate, we can therefore also apply Theorem 2.1 in [14] to the LL( $d$ ) cavity process. In this manner we obtain that

$$f(w) = \lambda d \left( C_d(0) \bar{G}(w) + \int_0^w c_d(u) \bar{G}(w-u) du \right),$$

which can easily be shown to be equivalent to (2.16) by using the fact that  $c_d(u) = \frac{d}{du} C_d(u)$ . The interpretation of this equation is as follows. The left-hand side of the equation corresponds to the downcrossing rate through level  $w$ , while the right-hand side denotes the upcrossing rate through  $w$ . Another way to write this equality is :

$$\bar{F}'(w) = -\lambda \left[ \int_0^w \bar{F}(u)^d g(w-u) du + \bar{G}(w) - \bar{F}(w)^d \right]. \quad (2.17)$$

This equation is not all too important for LL( $d$ ) but in the upcoming chapters, it is this equation which we generalize to other workload dependent load balancing policies.

### 2.6.1 Fixed point iteration

We propose to use the following simple FPI to solve the integral equation (2.12):

$$\bar{F}_{n+1}(w) = \rho - \lambda \cdot \left( \int_0^w (1 - \bar{F}_n(u)^d) \bar{G}(w-u) du \right),$$

which we prove converges to the unique fixed point provided that  $\rho < d^{-1/d}$ . In Section 2.8 we further show that if the job sizes follow a PH distribution,

we can directly compute the limiting workload distribution  $\bar{F}(w)$  by solving a simple set of ODEs (for any  $\rho < 1$ ), meaning there is no need to make use of the above FPI. Furthermore, the DIDE given in (2.17) can alternatively be used to obtain  $\bar{F}(w)$ .

Define the function space  $\text{ccdf}_\rho \subseteq [0, \rho]^{[0, \infty)}$  to be the space of complemented cumulative distribution functions starting in  $\rho$ , i.e., the space of functions which satisfy:

- $\bar{F}(0) = \rho$ ,
- $\lim_{w \rightarrow \infty} \bar{F}(w) = 0$ ,
- for  $w, h > 0$  :  $\bar{F}(w + h) \leq \bar{F}(w)$ ,
- for  $w > 0$  :  $\lim_{h \rightarrow 0^+} \bar{F}(w + h) = \bar{F}(w)$ .

On this space we can define an operator  $T_d : \text{ccdf}_\rho \rightarrow \mathbb{R}^{[0, \infty)}$  defined by:

$$T_d \bar{F} : [0, \infty) \rightarrow \mathbb{R} : w \mapsto \rho - \lambda \cdot \left( \int_0^w (1 - \bar{F}(u)^d) \bar{G}(w - u) du \right).$$

**Lemma 10.** *For  $\bar{F} \in \text{ccdf}_\rho$ , we have  $T_d \bar{F} \in \text{ccdf}_\rho$ .*

*Proof.* The only non-trivial part is to show that  $\lim_{w \rightarrow \infty} T_d \bar{F}(w) = 0$ . We find:

$$\begin{aligned} \lim_{w \rightarrow \infty} \left| \int_0^w (1 - \bar{F}(u)^d) \cdot \bar{G}(w - u) du \right| &\leq \lim_{w \rightarrow \infty} \int_0^w \bar{G}(w - u) du \\ &= \mathbb{E}[G], \end{aligned}$$

which shows that  $\lim_{w \rightarrow \infty} T_d \bar{F}(w) \geq 0$ . To obtain the other inequality observe that for any  $\varepsilon > 0$ , we can find a  $U > 0$  for which:

$$\lim_{w \rightarrow \infty} \int_U^w \bar{G}(w - u) du > \sqrt{1 - \varepsilon} \mathbb{E}[G], \quad 1 - \bar{F}(u)^d \geq \sqrt{1 - \varepsilon},$$

for  $u > U$ . We thus find:

$$\begin{aligned} \lim_{w \rightarrow \infty} \int_0^w d(1 - \bar{F}(u)^d) \bar{G}(w - u) du &\geq \lim_{w \rightarrow \infty} \int_U^w d(1 - \bar{F}(u)^d) \bar{G}(w - u) du \\ &\geq (1 - \varepsilon) \mathbb{E}[G] \end{aligned}$$

this shows that  $\lim_{w \rightarrow \infty} T_d \bar{F}(w) \leq 0$   $\square$

**Remark 10.** Due to the above lemma we may write  $T_d : \text{ccdf}_\rho \rightarrow \text{ccdf}_\rho$ .

**Remark 11.** We can define an order on  $\text{ccdf}_\rho$  by stating that  $\bar{F}_1 \preceq \bar{F}_2 \Leftrightarrow \forall w \in [0, \infty) : \bar{F}_1(w) \geq \bar{F}_2(w)$ , then a simple application of the Knaster-Tarski theorem [85] guarantees the existence of a fixed point of  $T_d$ . Indeed note that we have  $\bar{F}_1 \preceq \bar{F}_2 \Rightarrow T_d \bar{F}_1 \preceq T_d \bar{F}_2$ .

**Theorem 11.** For any  $\bar{F}_1, \bar{F}_2 \in \text{ccdf}_\rho$  we have:

$$d_K(T_d \bar{F}_1, T_d \bar{F}_2) \leq d\rho^d \cdot d_K(\bar{F}_1, \bar{F}_2),$$

where  $d_K$  denotes the uniform (or Kolmogorov) metric, i.e.,  $d_K(\bar{F}_1, \bar{F}_2) = \sup_w |\bar{F}_1(w) - \bar{F}_2(w)|$ .

*Proof.* Let  $\varepsilon > 0$  be arbitrary and let  $w^*$  be such that:

$$\begin{aligned} & \sup_w \int_0^w |\bar{F}_1(u)^d - \bar{F}_2(u)^d| \bar{G}(w-u) du \\ & < \int_0^{w^*} |\bar{F}_1(u)^d - \bar{F}_2(u)^d| \bar{G}(w^*-u) du + \varepsilon. \end{aligned}$$

We therefore have that  $d_K(T_d \bar{F}_1, T_d \bar{F}_2)$  is bounded above by:

$$\lambda \int_0^{w^*} |\bar{F}_2(u)^d - \bar{F}_1(u)^d| \bar{G}(w^*-u) du + \varepsilon.$$

We now use the fact (which can be shown by applying the mean value theorem) that for any  $x, y \in [0, \rho]$  we have  $|x^d - y^d| \leq d\rho^{d-1} \cdot |x - y|$ . This shows by applying the above that we have:

$$\begin{aligned} d_K(T_d \bar{F}_1, T_d \bar{F}_2) & < \lambda \int_0^{w^*} d\rho^{d-1} |\bar{F}_1(u) - \bar{F}_2(u)| \bar{G}(w^*-u) du + \varepsilon \\ & \leq \lambda d\rho^{d-1} d_K(\bar{F}_1, \bar{F}_2) \int_0^{w^*} \bar{G}(w^*-u) du + \varepsilon \\ & \leq d\rho^d d_K(\bar{F}_1, \bar{F}_2) + \varepsilon, \end{aligned}$$

which completes the proof.  $\square$

**Remark 12.** In particular, for  $\rho < e^{-1/e} \approx 0.6922$  the above theorem shows by the Banach fixed-point theorem that  $T_d$  admits a unique fixed point which

can be found by our proposed FPI with speed of convergence  $d_K(\bar{F}^*, \bar{F}_n) \leq \frac{d^n \rho^{nd}}{1-d\rho^d} d_K(\bar{F}_1, \bar{F}_0)$ . This follows from the fact that  $d^{-1/d}$  attains a minimum in  $e$ . For higher values of  $\rho$ ,  $d$  must be such that  $d\rho^d < 1$  to guarantee convergence via Theorem 11. Numerical experiments using both light-tailed and heavy-tailed distributions suggest that the FPI converges quickly for any  $\rho < 1$ .

**Remark 13.** Alternatively, one can solve the DIDE given in (2.17). Both methods are fast and appear to work for values of  $\rho$  close to one. Moreover, we will see in Chapter 4 that for more complex models, solving (2.17) is more efficient whenever a forward Euler scheme can be used (this is the case for future independent models). When the DIDE can not be solved using a forward Euler scheme, solving the FPE using a FPI proves to be more efficient (as is the case for future dependent models).

Using this numerical method, we can also compute the waiting and response time distribution. Indeed, the ccdf of the waiting time is given by  $\bar{F}_W(w) = \bar{F}(w)^d$ . The ccdf of the response time is given by the convolution of  $g$  with the ccdf of the waiting time, i.e.  $\bar{F}_R(w) = \bar{G}(w) + (g * \bar{F}_W)(w)$ .

## 2.7 Exponential job sizes

In the previous section we established an integral equation for the limiting stationary workload distribution (for any non-idling service discipline). In this section we derive an explicit expression for this distribution in case of exponential job sizes with mean 1, that is, when  $\bar{G}(w) = e^{-w}$  and  $\rho = \lambda$ . In addition we also derive an explicit expression for the limiting response time distribution in case the service discipline is FCFS. Assuming  $\mathbb{E}[G] = 1$  is only for technical reasons, to obtain the results for the case where  $\mathbb{E}[G]$  is arbitrary, one mostly needs to replace occurrences of  $\lambda$  by  $\rho$ .

### 2.7.1 Limiting workload distribution

**Theorem 12.** *The ccdf of the limiting stationary workload distribution for the LL( $d$ ) policy for any non-idling service discipline with exponential job sizes with mean 1 is given by:*

$$\bar{F}(w) = (\lambda + (\lambda^{1-d} - \lambda)e^{(d-1)w})^{\frac{1}{1-d}}. \quad (2.18)$$

*Proof.* Using (2.12) with  $\bar{G}(w) = e^{-w}$  and  $\rho = \lambda$ , we have

$$\bar{F}(w) = \lambda - \lambda d \int_0^w C_d(u) e^{u-w} du, \quad (2.19)$$

Taking the derivative on both sides and using the Leibniz integral rule, we find the following simple ODE for  $\bar{F}(w)$ :

$$\begin{aligned} \bar{F}'(w) &= -\lambda(1 - \bar{F}(w)^d) + \lambda \int_0^w (1 - \bar{F}(u)^d) e^{u-w} du \\ &= -\lambda(1 - \bar{F}(w)^d) + (F(w) - (1 - \lambda)) \\ &= -\bar{F}(w) + \lambda \bar{F}(w)^d, \end{aligned} \quad (2.20)$$

with boundary condition  $\bar{F}(0) = \lambda$ . This ODE can be solved explicitly and one easily verifies that the solution  $\bar{F}(w)$  is given by:

$$\bar{F}(w) = (\lambda + (\lambda^{1-d} - \lambda)e^{(d-1)w})^{\frac{1}{1-d}}.$$

□

**Remark 14.** *There is a striking and unexpected similarity between the limiting workload distribution of the LL( $d$ ) policy and the response time distribution of the replication with cancellation on completion [39, Section 5] in case of exponential job sizes in the sense that the response time distribution of the latter system solves exactly the same ODE as in (2.20), except that it is subject to the boundary condition  $\bar{F}_R(0) = 1$ , with  $\bar{F}_R$  the response time distribution for replication, with cancellation on completion and i.i.d. replicas (see also (29)).*

**Remark 15.** *For  $d$  large,  $\bar{F}(w)$  can be approximated by  $\lambda e^{-w}$  as  $(\lambda^{1-d} - \lambda)^{1/(1-d)}$  is close to  $\lambda$  for large  $d$ . This result is expected as for large  $d$  we expect that a fraction  $\lambda$  of the servers contains exactly one job and the remaining workload of any such job is exponentially distributed due to the memoryless nature of the exponential distribution.*

In order to obtain an expression for the expected workload of a server, we first recall the following integral representation for the analytic continuation of the hypergeometric function  ${}_2F_1(a, b; c; z)$  [2, Chapter 15]

$${}_2F_1(a, b; c; z) = \frac{1}{B(b, c-b)} \int_0^1 x^{b-1} (1-x)^{c-b-1} (1-zx)^{-a} dx, \quad (2.21)$$

where  $B(x, y) = \int_0^1 t^{x-1}(1-t)^{y-1}dt$  is the Beta function. This integral expression is valid for any  $c > b > 0$  and  $z < 1$ . When  $|z| < 1$  this function can be represented as an infinite sum using the Pochhammer symbol (or falling factorial)  $(q)_n = \prod_{k=0}^{n-1} (q+k)$  when  $n > 0$  and  $(q)_0 = 1$ :

$${}_2F_1(a, b; c; z) = \sum_{n=0}^{\infty} \frac{(a)_n (b)_n}{(c)_n} \frac{z^n}{n!}. \quad (2.22)$$

**Theorem 13.** *The mean workload  $\mathbb{E}[L_d]$  under the LL( $d$ ) policy with exponential job sizes with mean 1 is given by:*

$$\mathbb{E}[L_d] = \sum_{n=0}^{\infty} \frac{\lambda^{dn+1}}{1 + n(d-1)}, \quad (2.23)$$

in particular we find:

$$\begin{aligned} \mathbb{E}[L_2] &= -\frac{\log(1-\lambda^2)}{\lambda}, \\ \mathbb{E}[L_3] &= -\frac{1}{\sqrt{\lambda}} \cdot \log\left(\frac{\sqrt{1-\lambda^3}}{\lambda^{3/2}+1}\right). \end{aligned}$$

*Proof.* We employ the notation  $b = \lambda^{1-d} - \lambda$ . We begin by computing (using  $y = e^{-w}$  and  $x = y^{d-1}$ ):

$$\begin{aligned} \mathbb{E}[L_d] &= \int_0^\infty \bar{F}(w) dw \\ &= \int_0^1 \frac{1}{(\lambda y^{d-1} + b)^{1/(d-1)}} dy \\ &= \frac{1}{b^{1/(d-1)}} \frac{1}{(d-1)} \int_0^1 \frac{x^{-(d-2)/(d-1)}}{(1 + \frac{\lambda}{b}x)^{1/(d-1)}} dx \end{aligned}$$

Hence, by (2.21) this last integral can be expressed via the hypergeometric function  ${}_2F_1$  as

$$\mathbb{E}[L_d] = \frac{1}{b^{1/(d-1)}} \cdot {}_2F_1\left(\frac{1}{d-1}, \frac{1}{d-1}; 1 + \frac{1}{d-1}; -\frac{\lambda}{b}\right).$$

Note that we cannot directly use the sum representation of  ${}_2F_1$  as  $\lambda/b$  may become greater than 1 (which happens when  $\lambda$  gets close to one). Therefore

we now employ the well-known linear transformation formulas:

$$\begin{aligned} {}_2F_1(a, b; c; z) &= (1 - z)^{c-a-b} \cdot {}_2F_1(c - a, c - b; c; z) \\ {}_2F_1(a, b; c; z) &= (1 - z)^{-a} \cdot {}_2F_1\left(a, c - b; c; \frac{z}{z - 1}\right). \end{aligned} \quad (2.24)$$

Using (2.24) indicates that

$$\begin{aligned} \mathbb{E}[L_d] &= \frac{1}{b^{1/(d-1)}} \left(1 + \frac{\lambda}{b}\right)^{-\frac{1}{d-1}} \cdot {}_2F_1\left(1, \frac{1}{d-1}; 1 + \frac{1}{d-1}; \lambda^d\right) \\ &= \lambda \cdot {}_2F_1\left(1, \frac{1}{d-1}; 1 + \frac{1}{d-1}; \lambda^d\right) \end{aligned}$$

As  $\lambda^d \in (0, 1)$ , we can use the sum representation given by (2.22) to find that

$$\mathbb{E}[L_d] = \sum_{n=0}^{\infty} \frac{\lambda^{nd+1}}{1 + n(d-1)},$$

as  $(1)_n = n!$  and  $(1/(d-1))_n / (1/(d-1) + 1)_n = 1/(1 + n(d-1))$ . The expressions for  $d = 2, 3$  can be either found directly by looking at the Taylor expansion of the logarithm or by solving the integral representation of  $\mathbb{E}[L_d]$ .

□

## 2.7.2 Limiting waiting and response time distribution

We now focus on the limiting waiting and response time distribution  $W$  and  $R$  in case the service discipline is FCFS. For the ccdf of the waiting time distribution, one easily finds that it is given by  $\bar{F}_W(w) = \bar{F}(w)^d$ . For the ccdf of the response time distribution, we find:

**Theorem 14.** *The ccdf of the limiting response time distribution of the LL( $d$ ) policy with FCFS service and exponential job sizes with mean 1 is given by:*

$$\bar{F}_R(w) = (\lambda^d + (1 - \lambda^d)e^{(d-1)w})^{\frac{1}{1-d}}. \quad (2.25)$$

*Proof.* Let  $E$  be an exponential random variable with mean 1 and let  $L_i, i = 1, \dots, d$  denote the  $d$  independent workloads of the  $d$  randomly selected

servers. We find:

$$\begin{aligned}\bar{F}_R(w) &= \mathbb{P} \left\{ E + \min_{i=1}^d L_i > w \right\} \\ &= e^{-w} + \int_0^w \bar{F}(w-t)^d e^{-t} dt.\end{aligned}$$

Due to (2.18) and using standard integration techniques, this integral can be simplified to:

$$\bar{F}_R(w) = e^{-w} \cdot \left( 1 + \frac{1}{\lambda b^{1/(d-1)}} \cdot \int_{(\frac{b}{\lambda})^{1/(d-1)}}^{e^w (\frac{b}{\lambda})^{1/(d-1)}} (1+x^{d-1})^{d/(1-d)} dx \right),$$

where  $b = \lambda^{1-d} - \lambda$  as before. This is an integral that can be solved exactly to prove the statement.  $\square$

**Remark 16.** It is easy to verify that the workload and response time distributions  $\bar{F}(w)$  and  $\bar{F}_R(w)$  have the same increasing failure rate  $r(w) = f(w)/\bar{F}(w) = f_R(w)/\bar{F}_R(w)$ .

**Remark 17.** For  $d$  large,  $\bar{F}_R(w) \approx e^{-w}$ , as expected.

**Theorem 15.** The mean of the limiting response time distribution for the LL( $d$ ) policy with FCFS service and exponential job sizes with mean 1 is given by:

$$\mathbb{E}[R] = \sum_{n=0}^{\infty} \frac{\lambda^{dn}}{1+n \cdot (d-1)}. \quad (2.26)$$

The mean waiting time is given by  $\mathbb{E}[W] = \mathbb{E}[R] - 1 = \sum_{n=1}^{\infty} \frac{\lambda^{dn}}{1+n \cdot (d-1)}$ .

*Proof.* Let  $L_i$  be i.i.d. copies of the workload distribution  $L$  and  $E$  an exponential distribution with mean one, we find:

$$\begin{aligned}\mathbb{E}[R] &= \mathbb{E}[E + \min\{L_1, \dots, L_d\}] \\ &= 1 + \int_0^\infty \bar{F}(w)^d ds.\end{aligned}$$

Using (2.18) and standard integration techniques (mainly substitution), we can reduce this expression to:

$$\mathbb{E}[R] = 1 + \frac{1}{\lambda^{d/(d-1)} \cdot (d-1)} \cdot \int_0^{\lambda/b} \frac{v^{1/(d-1)}}{(1+v)^{d/(d-1)}} dv.$$

Using the substitution  $y = \frac{v}{1+v}$ , one can show that the above integral reduces to

$$1 + \frac{\lambda^d}{d} \cdot {}_2F_1\left(\frac{d}{d-1}, 1; 1 + \frac{d}{d-1}; \lambda^d\right).$$

As  $\lambda^d \in (0, 1)$ , one can use (2.22) and the claimed equality follows as  $(1)_n = n!$  and  $(d/(d-1))_n / (1 + d/(d-1))_n = d / ((n+1)(d-1) + 1)$ .  $\square$

**Remark 18.** In the proof of Theorem 15 it is also possible to directly use (2.25) instead of relying on (2.18).

**Remark 19.** Note that by combining Theorems 13 and 15 we have shown that  $\mathbb{E}[L] = \lambda\mathbb{E}[R]$ . Due to Little's law this shows that the mean workload of a server under the LL( $d$ ) policy for exponential job sizes with mean 1 is equal to the mean number of jobs in such a server, that is  $\mathbb{E}[L] = \mathbb{E}[Q]$ . The relation  $\mathbb{E}[L] = \lambda\mathbb{E}[R]$  also yields simple formulas for  $\mathbb{E}[R]$  in case  $d = 2, 3$  due to Theorem 13. It is possible to derive similar expressions for larger  $d$  values, but these become more and more complex as  $d$  increases.

**Remark 20.** In [39] the mean of the limiting response time distribution in case of exponential job sizes for the replication with cancellation on completion policy (under the assumption of the independence ansatz) was argued to be equal to (with  $\mu = 1/\mathbb{E}[G]$ ):

$$\mathbb{E}[R^{Red_{iid}(d)}] = \frac{{}_2F_1(1, 1; 1 + \frac{d}{d-1}; \frac{-\rho}{1-\rho})}{\mu d(1-\rho)}.$$

This expression can be reduced to a simple sum formula as follows (using (2.24) and (2.22) as  $\rho \in (0, 1)$ )

$$\begin{aligned} {}_2F_1\left(1, 1; 1 + \frac{d}{d-1}; \frac{-\rho}{1-\rho}\right) &= (1-\rho) {}_2F_1\left(1, \frac{d}{d-1}; 1 + \frac{d}{d-1}; \rho\right) \\ &= (1-\rho) \sum_{n=0}^{\infty} \frac{(1)_n (\frac{d}{d-1})_n \rho^n}{(1 + \frac{d}{d-1})_n n!}, \end{aligned}$$

which allows us to conclude that

$$\mathbb{E}[R^{Red_{iid}(d)}] = \frac{1}{\mu} \sum_{n=0}^{\infty} \frac{\rho^n}{n(d-1) + d}.$$

Note that  $E[R^{Red_{iid}(d)}]$  converges to  $1/(d\mu)$  as  $\rho$  tends to zero due to the independent execution times of the replicas in [39].

## 2.8 Phase-type and deterministic job sizes

In Section 2.6.1 we proposed a FPI to compute the limiting workload distribution  $\bar{F}(w)$  under  $LL(d)$  for any job size distribution  $\bar{G}$ , that was proven to converge if  $d\rho^d < 1$ . We now show that  $\bar{F}(w)$  can also be directly obtained as the solution of a set of coupled ODEs for any  $\rho < 1$ , provided that the job lengths follow a PH distribution. PH distributions are distributions with a modulating finite state background Markov chain [55] and any general positive-valued distribution can be approximated arbitrary closely with a PH distributions. Further, various fitting tools are available online for PH distributions (e.g., [75, 52]). A PH distribution with  $G(0) = 0$  is fully characterized by a stochastic vector  $\alpha = (\alpha_i)_{i=1}^n$  and a subgenerator matrix  $A = (a_{i,j})_{i,j=1}^n$  such that  $\bar{G}(w) = \alpha e^{Aw} \mathbf{1}$ , where  $\mathbf{1}$  is a column vector of ones.

**Theorem 16.** *Suppose the job lengths have a PH distribution characterized by  $(\alpha, A)$ , then the ccdf of the limiting workload distribution under the  $LL(d)$  policy satisfies:*

$$\begin{aligned}\bar{F}'(w) &= -\lambda ((1 - \bar{F}(w)^d) + \alpha A \xi(w)), \\ \xi'(w) &= (1 - \bar{F}(w)^d) \mathbf{1} + A \xi(w),\end{aligned}$$

with  $\bar{F}(0) = \rho$ ,  $\xi(0) = 0$  and  $\xi(w) : \mathbb{R} \rightarrow \mathbb{R}^{n \times 1}$ .

*Proof.* For  $i \in \{1, \dots, n\}$  we define:

$$\xi_i(w) = \int_0^w (1 - \bar{F}(u)^d) e_i^T e^{(w-u)A} \mathbf{1} du,$$

where  $e_i^T$  is the  $i$ -th row of the identity matrix  $I_n$ . First note that  $\xi_i(0) = 0$ . We now derive a differential equation for  $\xi_i(w)$ . Using the equality  $I_n = \sum_{k=1}^n e_k e_k^T$  we find :

$$\begin{aligned}\xi'_i(w) &= (1 - \bar{F}(w)^d) + \int_0^w (1 - \bar{F}(u)^d) e_i^T A I_n e^{(w-u)A} \mathbf{1} du \\ &= (1 - \bar{F}(w)^d) + \sum_{k=1}^n \int_0^w (1 - \bar{F}(u)^d) e_i^T A e_k e_k^T e^{(w-u)A} \mathbf{1} du \\ &= (1 - \bar{F}(w)^d) + \sum_{k=1}^n a_{i,k} \xi_k(w).\end{aligned}$$

In matrix notation this yields:

$$\xi'(w) = (1 - \bar{F}(w)^d)\mathbf{1} + A\xi(w).$$

Due to (2.12) and  $\bar{G}(w-u) = \alpha e^{(w-u)A}\mathbf{1}$ , we have  $\bar{F}'(w) = -\lambda\alpha\xi'(w)$ , which yields the equation for  $\bar{F}'(w)$ .  $\square$

**Remark 21.** We could have alternatively derived the result of Theorem 16 directly from (2.17). Indeed, we have  $g(w) = \alpha e^{Aw}\mu$  with  $\mu = -A \cdot \mathbf{1}$ , using (2.17), we obtain:

$$\bar{F}'(w) = -\lambda \left[ (1 - \bar{F}(w)^d) - \alpha \int_0^w (1 - \bar{F}(u)^d)e^{(w-u)A}\mu du \right]. \quad (2.27)$$

Defining  $\xi(w) = \int_0^w (1 - \bar{F}(u)^d)e^{(w-u)A}\mu du$  as in the above proof, one may recover the result directly from (2.27). This method generalizes well to other workload dependent load balancing policies and is further explored in Section 4.8.

We now generalize this result to the case where the service times are the sum of a deterministic random variable and a PH distribution.

**Theorem 17.** Assume the service times are the sum of a deterministic random variable equal to  $\tau > 0$  and a phase-type distribution characterized by  $(\alpha, A)$ , i.e.,  $\bar{G}(w) = I_{\{w \leq \tau\}} + I_{\{w > \tau\}}\alpha e^{(w-\tau)A}\mathbf{1}$ , then the ccdf of the limiting workload distribution under the LL( $d$ ) policy satisfies:

$$\begin{aligned} \bar{F}'(w) &= -\lambda(1 - \bar{F}(w)^d), & w \leq \tau, \\ \bar{F}'(w) &= -\lambda((1 - \bar{F}(w)^d) + \alpha A\xi(w - \tau)), & w > \tau, \\ \xi'(w) &= (1 - \bar{F}(w)^d)\mathbf{1} + A\xi(w), \end{aligned}$$

with  $\xi(0) = 0$  and  $\bar{F}(0) = \rho = \lambda(\tau + \alpha(-A)^{-1}\mathbf{1})$ .

*Proof.* We distinguish two cases: first let  $w \in [0, \tau]$ , we find that  $\bar{F}(w) = \rho - \lambda \int_0^w 1 - \bar{F}(u)^d du$ , differentiating this equation once yields the first equation.

For the second note that we have (using the notation from the proof of Theorem 16):

$$\bar{F}(w) = \rho - \lambda\alpha\xi(w - \tau) - \lambda \int_{w-\tau}^w (1 - \bar{F}(u)^d)du.$$

Taking the derivative and using the expression for  $\xi'(w)$  found in Theorem 16 completes the proof.  $\square$

From the above result, we obtain a very simple DDE for  $\bar{F}(w)$  in case job sizes are deterministic.

**Theorem 18.** *If the job sizes are deterministic and equal to one, the ccdf  $\bar{F}(w)$  is determined by  $\bar{F}(0) = \lambda$ , and*

$$\begin{aligned}\bar{F}'(w) &= -\lambda(1 - \bar{F}(w)^d) & w \in [0, 1), \\ \bar{F}'(w) &= -\lambda(\bar{F}(w-1)^d - \bar{F}(w)^d) & w \geq 1.\end{aligned}$$

*Proof.* The proof is similar to the proof of Theorem 17.  $\square$

In general, when job sizes have a discrete distribution  $\sum_n p_n \delta_{x_n}$  and we assume without loss of generality that  $x_0 < x_1 < \dots$ , it is not hard to see that the result of Theorem 18 generalizes to:

$$\begin{aligned}\bar{F}'(w) &= -\lambda(1 - \bar{F}(w)^d) & w \in [0, x_0), \\ \bar{F}'(w) &= -\lambda(p_0 \bar{F}(w-x_0)^d - \bar{F}(w)^d) & w \in [x_0, x_1), \\ \bar{F}'(w) &= -\lambda(p_0 \bar{F}(w-x_0)^d + p_1 \bar{F}(w-x_1)^d - \bar{F}(w)^d) & w \in [x_1, x_2), \\ &\dots\end{aligned}$$

Combining this result with Theorem 16 allows one to reduce the computation of  $\bar{F}(w)$  in case the job size distribution is a mix of a discrete distribution and a PH distribution to solving a system of DDEs.

**Remark 22.** *We note that the ODEs and DDEs presented in this section have a unique solution: the existence follows from the fact that (2.12) solves the ODE/DDE, while the uniqueness follows from [29, Section 23, theorem A].*

**Remark 23.** *It is easy to compute the ccdf of the waiting and response time distribution  $\bar{F}_R(w)$  given  $\bar{F}(w)$  as the probability that a new arrival joins a queue with a workload exceeding  $w$  is given by  $\bar{F}_W(w) = \bar{F}(w)^d$  under the LL( $d$ ) policy. Furthermore, the response time is simply the convolution of the job size distribution and the waiting time distribution as these are independent random variables.*

## 2.9 LL( $d$ ) versus SQ( $d$ )

The aim of this section is to study the margin of improvement that can be achieved by using exact workload information as opposed to the coarser

queue length information used by  $SQ(d)$ . This margin of improvement is of interest to understand the possible response time improvements offered by schedulers that implement late binding (as discussed in the introduction). Furthermore, we also compare the  $SQ(d)$  policy with the  $LL(d)$  policy where the job sizes of the latter take the late binding overhead into account. We start by focusing on exponential job sizes, for which we can also establish some closed form results.

### 2.9.1 Exponential job sizes

In this subsection we compare the limiting response time of the  $LL(d)$  and  $SQ(d)$  policies for exponential job sizes with mean 1 and FCFS service. This comparison provides an answer on the reduction in the response times that can be obtained if the workloads at the different servers are known instead of the coarser queue length information. To distinguish between the response times of both policies we make use of the superscripts  $(LL(d))$  and  $(SQ(d))$ . Moreover, we use the index  $\lambda$  to denote the arrival rate. For the  $SQ(d)$  policy the mean of the limiting response time distribution is given by [67]

$$\mathbb{E} \left[ R_{\lambda}^{(SQ(d))} \right] = \frac{1}{\lambda} \sum_{k=1}^{\infty} \lambda^{\frac{d^k - 1}{d-1}}.$$

**Theorem 19.** *The mean of the limiting response time distribution for the  $LL(d)$  policy is smaller than the mean for the  $SQ(d)$  policy for exponential job sizes with mean 1, moreover*

$$\mathbb{E} \left[ R_{\lambda}^{(SQ(d))} \right] - \mathbb{E} \left[ R_{\lambda}^{(LL(d))} \right] = \frac{1}{\lambda} \sum_{k=1}^{\infty} A_k,$$

where for  $\lambda \in (0, 1)$ :

$$A_k = \lambda^{\frac{d^{k+1} - 1}{d-1}} - \sum_{n=1}^{d^k} \frac{\lambda^{nd+1 + \frac{d^{k+1} - d^2}{d-1}}}{1 + n(d-1) + (d^k - d)} > 0.$$

*Proof.* Due to (2.26), we need to show:

$$\sum_{n=1}^{\infty} \frac{\lambda^{dn+1}}{1 + n(d-1)} \leq \sum_{k=2}^{\infty} \lambda^{\frac{d^k - 1}{d-1}}.$$

To see this, we group the terms on the left hand side with  $n \in \{\sum_{s=0}^{k-1} d^{s-1}, \dots, \sum_{s=1}^k d^s\}$  together and compare their sum with the term  $\lambda^{\frac{d^{k+1}-1}{d-1}}$  on the right hand side for  $k \geq 1$ . We have

$$\begin{aligned} \sum_{n=1+\dots+d^{k-1}}^{d+\dots+d^k} \frac{\lambda^{nd+1}}{1+n(d-1)} &< \sum_{n=1+\dots+d^{k-1}}^{d+\dots+d^k} \frac{\lambda^{d(1+d+\dots+d^{k-1})+1}}{(1+\dots+d^{k-1})(d-1)} \\ &= \lambda^{1+d+\dots+d^k} = \lambda^{\frac{d^{k+1}-1}{d-1}}. \end{aligned}$$

Hence, the result follows.  $\square$

Often, one is interested in the behaviour of a load balancing policy as the system tends towards instability, that is, in the limit  $\lambda \rightarrow 1^-$ . It turns out that for most simple load balancing policies that we consider in this work (see Chapter 5) there is a computable number which can be used to represent how well a load balancing policy behaves in case  $\lambda \approx 1$ . In particular we have the following result

**Theorem 20.** *For the ratio of the mean of the limiting waiting/response time distribution of  $SQ(d)$  and  $LL(d)$  for exponential job sizes with mean 1 we have*

$$\lim_{\lambda \rightarrow 1^-} \mathbb{E} [W_\lambda^{(SQ(d))}] / \mathbb{E} [W_\lambda^{(LL(d))}] = \lim_{\lambda \rightarrow 1^-} \mathbb{E} [R_\lambda^{(SQ(d))}] / \mathbb{E} [R_\lambda^{(LL(d))}] = \frac{d-1}{\log(d)}.$$

*Proof.* In Chapter 5 we present two proofs for this result. The first can be found in Section 5.4.3, but requires the quite involved Theorem 59. The second proof is more straightforward and can be found in Section 5.6.3. The second proof relies on finding simple upper and lower bounds on the mean response time for the  $LL(d)$  policy.  $\square$

**Remark 24.** *As  $(d-1)/\log(d)$  tends to infinity as  $d$  becomes large, we note that for any  $c > 0$  there exists a  $\lambda$  and  $d$  such that:*

$$\mathbb{E} [R_\lambda^{(SQ(d))}] / \mathbb{E} [R_\lambda^{(LL(d))}] > c.$$

*In other words, for arbitrary  $\lambda$  and  $d$ , there is no bound on how much worse the  $SQ(d)$  policy performs than the  $LL(d)$  policy.*

As the mean response time tends to one as  $\lambda$  tends to zero for any reasonable load balancing policy, not much can be said on the low load limit variant of Theorem 20 w.r.t. response times. However, as the waiting time tends to zeros for both policies, the low load limit is interesting to study. It is not hard to see (c.f. Section 5.7) that we have:

$$\lim_{\lambda \rightarrow 0^+} \mathbb{E} \left[ W_\lambda^{(SQ(d))} \right] / \mathbb{E} \left[ W_\lambda^{(LL(d))} \right] = d.$$

In Figure 2.3a we plot the ratio  $\mathbb{E} \left[ R_\lambda^{(SQ(d))} \right] / \mathbb{E} \left[ R_\lambda^{(LL(d))} \right]$  as a function of  $\lambda$ . We note that this ratio increases with  $\lambda$  and approaches a constant as  $\lambda$  approaches one. Looking at this figure, the limit values for the ratio  $\mathbb{E} \left[ R_\lambda^{(SQ(d))} \right] / \mathbb{E} \left[ R_\lambda^{(LL(d))} \right]$  as  $\lambda$  tends to one may appear to be less than  $(d - 1) / \log(d)$  (as shown in Theorem 20), but this is simply due to the fact that this ratio still increases significantly between 0.999 and 1. From this figure we may conclude that the increase in the mean of the limiting response time distribution by using the coarser queue length information instead of the exact workload is below 50% when  $d = 2$  for exponential job sizes. For larger  $d$  we see a more significant increase under high load.

We further note that the curves for different  $d$  values cross one another. Intuitively this can be understood by noting that for  $\lambda$  small many jobs select an idle server and when an idle server is selected knowing the queue length is equally good as knowing the workload. When  $d$  increases it becomes more likely that an idle server is selected and thus we expect the mean response time ratio to decrease with increasing  $d$  when  $\lambda$  is small. For large  $\lambda$  it becomes unlikely that one of the selected queues is idle and  $SQ(d)$  has to rely on the coarser queue length information. When  $\lambda$  is large, we therefore see a larger loss of more information as  $d$  increases and thus the mean response time ratio now increases with increasing  $d$ .

In Figure 2.3b we observe the same figure but the ratio of the mean waiting time rather than the mean response times. We observe that the ratio now decreases rather than increases. This makes sense as we filter out all jobs which join an idle queue. Moreover, the convergence to the limiting value  $\frac{d-1}{\log(d)}$  can be seen much more clear in this plot.

**Remark 25.** *We will have much more to say on this subject in Chapter 4. In this chapter we present a general result which can be used to compute the (scaled) limiting the waiting time for many load balancing policies, includ-*

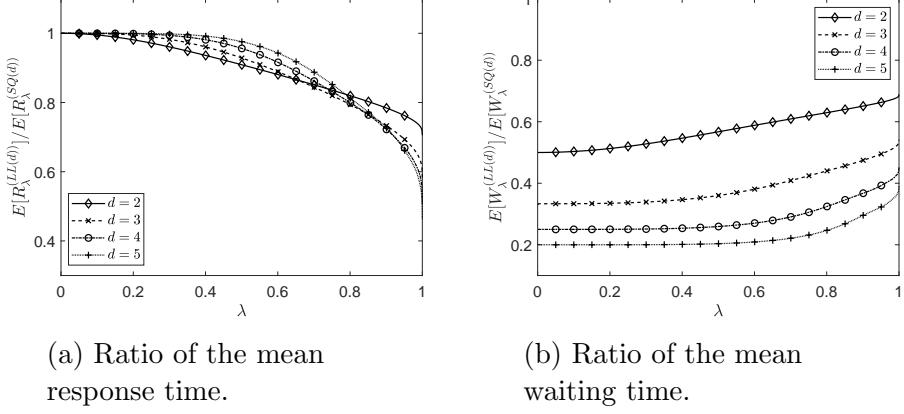


Figure 2.3: Ratio of the mean of the limiting response/waiting time distribution of  $SQ(d)$  and  $LL(d)$  for exponential job sizes with mean 1, FCFS service as a function of  $\lambda$ .

ing  $LL(d)$  and  $SQ(d)$ . This subject also still has quite a few open research directions (see 8.4).

Apart from comparing the mean response/waiting times, we can also easily compare the response time distribution of the  $LL(d)$  and  $SQ(d)$  policy. For the  $SQ(d)$  policy it is not hard to establish that the ccdf of the limiting response time distribution can be written as

$$\begin{aligned} \bar{F}_R^{(SQ(d))}(w) &= \sum_{k=1}^{\infty} \left( \lambda^{(d^{k-1}-1)d/(d-1)} - \lambda^{(d^k-1)d/(d-1)} \right) \sum_{n=0}^{k-1} \frac{w^n}{n!} e^{-w} \\ &= \sum_{n=0}^{\infty} \frac{w^n}{n!} e^{-w} \lambda^{(d^n-1)d/(d-1)}, \end{aligned} \quad (2.28)$$

by noting that a job which joins a queue of length  $k-1$  has an Erlang- $k$  distributed response time for exponential job sizes. Figure 2.4a depicts the response time distributions for  $\lambda = 0.95$  and  $d = 2, 3$  and 4. We note that  $\bar{F}_R(w)$  decreases as a function of  $d$  and  $\bar{F}_R^{(SQ(d))}(w)$  dominates  $\bar{F}_R^{(LL(d))}(w)$  for all  $w > 0$ . The next theorem proves an even stronger result: the ratio  $\bar{F}_R^{(SQ(d))}(w)/\bar{F}_R^{(LL(d))}(w)$  increases as a function of  $w$ , we also graphically show this fact in Figure 2.4b.

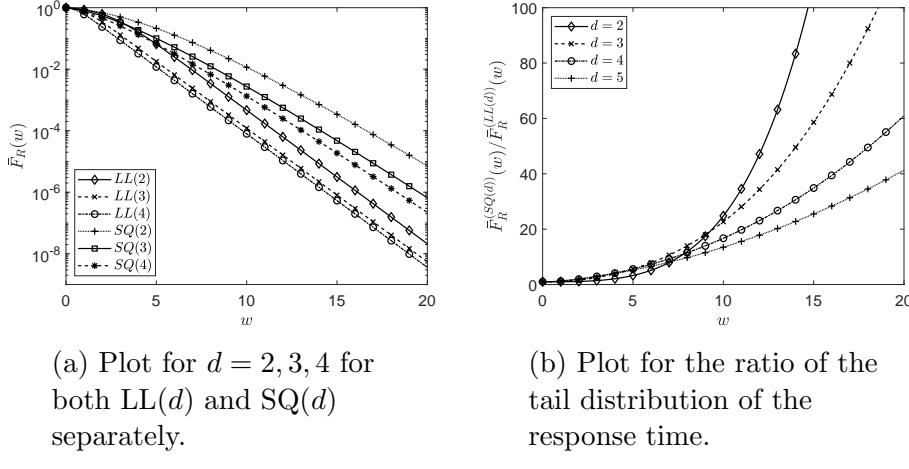


Figure 2.4: Plots associated to the limiting response time distribution of  $\text{SQ}(d)$  and  $\text{LL}(d)$  for exponential job sizes with mean 1, FCFS service and  $\lambda = 0.95$ .

**Theorem 21.** *The function  $f(w) = \bar{F}_R^{(\text{SQ},d)}(w)/\bar{F}_R^{(\text{LL},d)}(w)$  is non-decreasing on  $[0, \infty)$ , thus  $\bar{F}_R^{(\text{SQ},d)}(w) \geq \bar{F}_R^{(\text{LL},d)}(w)$  for all  $w$ .*

*Proof.* It suffices to show that  $f'(w) \geq 0$  for  $w > 0$  (as  $\bar{F}_R^{(\text{SQ},d)}(0) = \bar{F}_R^{(\text{LL},d)}(0) = 1$ ). Denote  $\mu = \lambda^d$ . Using (2.28) and (2.25), the condition  $f'(w) \geq 0$  can be restated as

$$\frac{\sum_{k=0}^{\infty} \mu^{\frac{d^k-1}{d-1}} (\mu^{d^k} - 1) \frac{w^k}{k!}}{\sum_{k=0}^{\infty} \mu^{\frac{d^k-1}{d-1}} \frac{w^k}{k!}} + \frac{(1-\mu)e^{(d-1)w}}{\mu + (1-\mu)e^{(d-1)w}} \geq 0.$$

By rearranging terms this is equivalent to showing:

$$e^{(d-1)w} \left( \sum_{k=0}^{\infty} \mu^{d^k} \mu^{\frac{d^k-1}{d-1}} \frac{w^k}{k!} \right) \geq \frac{\mu}{1-\mu} \sum_{k=0}^{\infty} \mu^{\frac{d^k-1}{d-1}} (1-\mu)^{d^k} \frac{w^k}{k!}.$$

For the left hand side we find, by using the Taylor expansion of  $e^{(d-1)w}$  and applying Merten's theorem (which states that if  $\sum_n a_n$  converges to  $A$  and  $\sum_n b_n$  converges to  $B$ , then the Cauchy product converges to  $AB$  if at least

one of the two sequences converges absolutely):

$$e^{(d-1)w} \left( \sum_{k=0}^{\infty} \mu^{d^k} \mu^{\frac{d^k-1}{d-1}} \frac{w^k}{k!} \right) = \sum_{n=0}^{\infty} \frac{w^n}{n!} \sum_{k=0}^n \binom{n}{k} (d-1)^{n-k} \mu^{d^k} \mu^{\frac{d^k-1}{d-1}}.$$

It therefore suffices to show that the inequality holds for all coefficients of  $\frac{w^n}{n!}$ , i.e. it remains to show that:

$$\frac{\mu}{1-\mu} \mu^{\frac{d^n-1}{d-1}} (1 - \mu^{d^n}) \leq \sum_{k=0}^n \binom{n}{k} (d-1)^{n-k} \mu^{d^k} \mu^{\frac{d^k-1}{d-1}}.$$

By noting that  $\frac{1-\mu^{d^n}}{1-\mu} \leq d^n$ , the result follows if the following holds

$$d^n \leq \sum_{k=0}^n \binom{n}{k} (d-1)^{n-k} \mu^{\frac{d^{k+1}-1}{d-1} - \frac{d^n-1}{d-1} - 1},$$

We clearly have an equality in  $\mu = 1$  (and for  $n = 0$ ). It therefore suffices to show that the right hand side decreases for  $\mu \in [0, 1]$  for  $n > 0$ . The first  $n$  terms are all convex decreasing, while the last term is convex increasing. The derivative of the sum of the first and last term in  $\mu = 1$  is  $(d^n - 1)(1 - (d-1)^{n-1}) \leq 0$ . Since the derivative of a convex function on  $[0, 1]$  is maximized in 1, the sum of the first and last term is decreasing and we may conclude that  $f'(w) \geq 0$ .  $\square$

### 2.9.2 Impact of job variability

In this subsection we study the impact of the job size variability on the ratio of the mean of the limiting response time distribution of  $SQ(d)$  and  $LL(d)$ . To this end, we make use of hyperexponential job sizes as described in Section 1.4.

The mean of the limiting response time distribution for the  $LL(d)$  policy can be computed in a fraction of a second for any  $\rho < 1$  by making use of Theorem 16. For the  $SQ(d)$  policy we use the FPI which is presented in Section 1.5 to determine the stationary queue length distribution of the cavity process associated to the equilibrium environment [20]. More specifically, we determine the queue length distribution of a sequence of M/G/1 FCFS queues with a queue length dependent arrival rate  $\lambda$ , where the queue length distribution determined during the  $n$ -th iteration determines the arrival rates

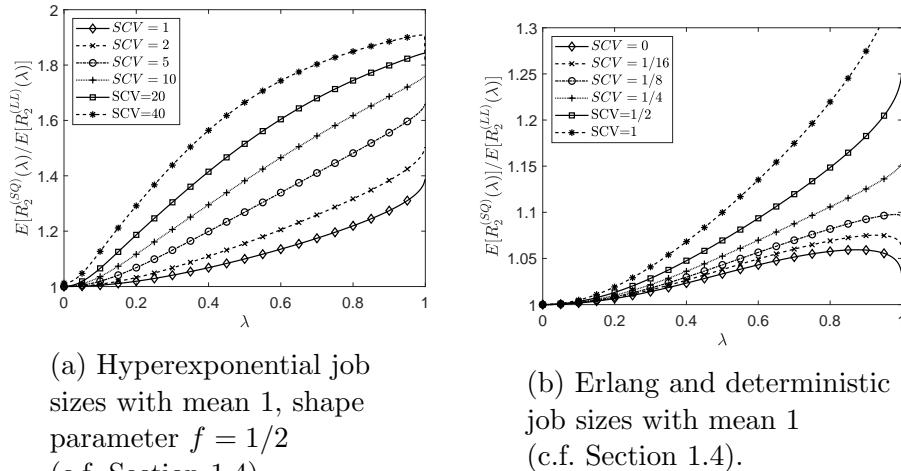


Figure 2.5: Ratio of the mean of the limiting response time distribution of SQ(2) and LL(2) with FCFS service as a function of the arrival rate  $\lambda$

of the  $n + 1$ -th iteration, until the queue length distribution converges (starting from the empty distribution). While the queue length distribution of such a queue can be computed in a very fast manner when the job sizes follow a PH distribution (or are deterministic), the number of iterations needed increases sharply as  $\rho$  approaches 1. This prevents us from studying what happens in the limit as  $\rho$  tends to one.

Figure 2.5a depicts the ratio of the mean of the limiting response time distribution of the SQ( $d$ ) and LL( $d$ ) policies when  $d = 2$  and  $f = 1/2$  (meaning half of the workload is offered by the *long* jobs). This ratio increases when the job sizes become more variable, which is expected as having precise workload information should be more valuable when jobs vary significantly in size. The results indicate that a mechanism like late binding can offer substantial gains even at fairly low loads if the job sizes vary significantly (and the round-trip time to fetch the job can be neglected). The results for  $f = 1/10$ , which implies that 90% of the workload is offered by the *long* jobs, are very similar (and therefore not depicted). For  $d > 2$  these ratios tend to increase under sufficiently high loads as in the exponential case.

For completeness we also present some results for job sizes with an SCV below 1 in Figure 2.5b. In this case we cannot make use of a hyperexponential

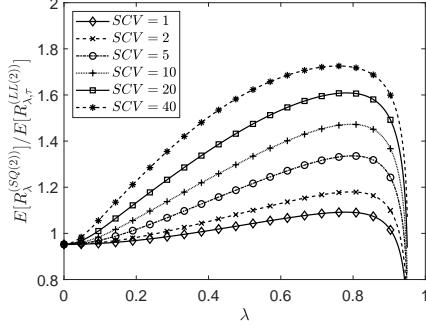
distribution and therefore consider Erlang- $k$  distributed and deterministic job sizes instead. This figure shows that as  $\lambda$  approaches 1 the ratio of the means of the limiting response time distribution starts to decrease for sufficiently small SCVs. In fact, studying this ratio for  $\lambda$  values closer to 1 as depicted in Figure 2.5b suggests that this ratio decreases to 1 for deterministic job sizes. This seems to make sense intuitively as for  $\lambda$  approaching one, the queue lengths become long and knowing the coarser queue length information is almost as good as knowing the exact workload. We further substantiate this claim in Section 8.4, where we compute the limiting scaled expected waiting time for the  $SQ(d)$  policy with deterministic job sizes and conjecture its value for  $LL(d)$ .

### 2.9.3 Late binding overhead

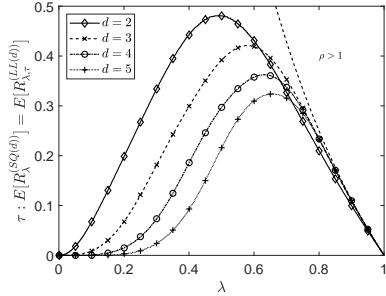
In the previous subsection we shed light on the margin of improvement that late binding can provide compared to the classic  $SQ(d)$  policy assuming that the jobs can be fetched from the dispatchers in negligible time. In this section we take the idleness caused by late binding into account. We do this by comparing the mean of the limiting response time distribution of the  $SQ(d)$  policy with the mean of the  $LL(d)$  policy, where the size of each job under the  $LL(d)$  policy is incremented by a deterministic quantity  $\tau$  that represents the overhead, that is, the time that the server remains idle under late binding while fetching the job. We denote the mean of the limiting response time in the latter case as  $\mathbb{E}[R_{d,\tau}^{(LL)}]$  and rely on Theorem 17 for its computation. We consider the same job size distributions (with average job size equal to one) as in the previous section.

In Figure 2.6a the ratio  $\mathbb{E}[R^{(SQ(d))}]/\mathbb{E}[R_{\tau}^{(LL(d))}]$  is shown as a function of  $\lambda$  for the case where  $\tau = 0.05$ , meaning each job induces an idle server period with a length equal to 5% of the mean job size. It indicates that for a wide range of arrival rates  $\lambda$ , late binding offers substantial gains over the  $SQ(d)$  policy even with an overhead of 5%. For systems with high job size variability, this range even includes arrival rates above 0.9. Note that the overhead of the scheduler implementation in [74] was estimated to be below 2%. We further note that late binding requires storing the jobs at the dispatcher(s) until a notification from one of the servers arrives, which may be regarded as a drawback compared to  $SQ(d)$  which allows immediate dispatching as soon as the queue length information is obtained.

In fact for medium loads much higher amounts of overhead can be toler-



(a) Ratio of the mean of the limiting response time distribution of  $SQ(2)$  and  $LL(2)$  with 5% overhead (i.e.,  $\tau = 0.05$ ) for hyperexponential job sizes with mean 1, shape parameter  $f = 1/2$  and FCFS service as a function of  $\lambda$ .



(b) Degree of delay that the  $LL(d)$  policy can tolerate without being outperformed by  $SQ(d)$  as a function of  $\lambda$  for hyperexponential job sizes with mean 1,  $SCV = 20$  and  $f = 1/2$ , i.e., the largest  $\tau$  such that  $E[R_\tau^{(LL(d))}] \leq E[R^{(SQ(d))}]$ .

Figure 2.6: Comparison mean response time for  $SQ(d)$  and  $LL(d)$  with late binding overhead.

ated by the  $LL(d)$  policy before it becomes inferior to  $SQ(d)$ . This is illustrated in Figure 2.6b, where we plot the largest  $\tau$  value for which  $E[R_{d,\tau}^{(LL(d))}] \leq E[R_d^{(SQ(d))}]$  when the  $SCV$  was set to 20. We observe that overheads of 25% and more can be tolerated for a system workload around 50%.

## 2.10 Finite system accuracy

In this section we briefly compare the tail of the limiting response time distribution  $\bar{F}_R(w)$  with simulation experiments where the number of servers  $N$  is finite. All simulation runs simulate the system up to time  $t = 10^7/N$  and use a warm-up period of 30%. Each simulation is the mean of 40 runs.

Figure 2.7a compares the expression for the limiting response time distribution given by (2.25) for exponential job sizes with simulation experiments.

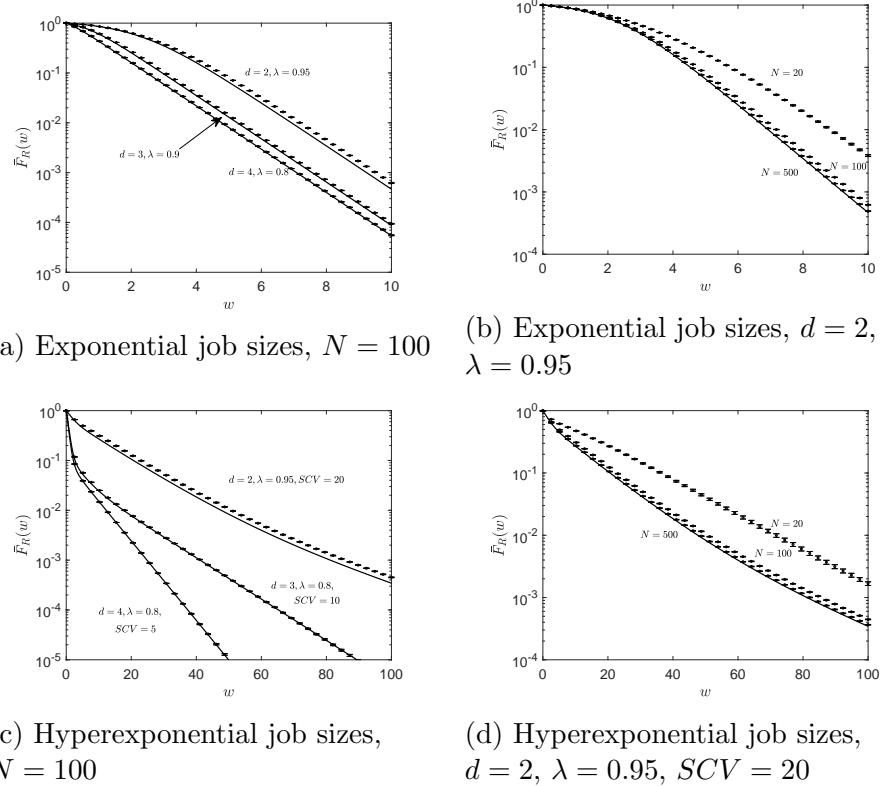
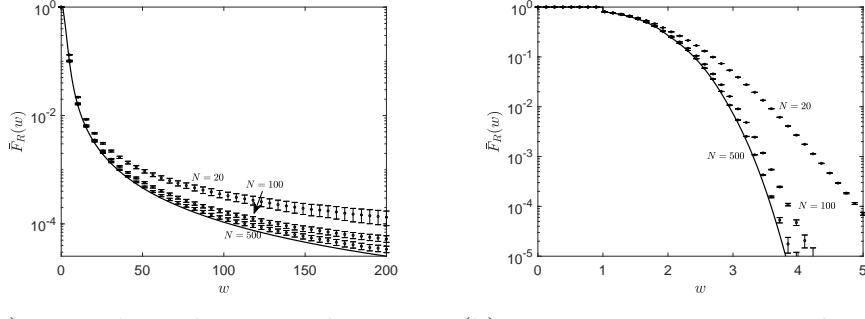


Figure 2.7: Limiting response time distribution vs. simulation for  $N$  servers with (hyper)exponential job sizes with mean 1. The full line represents the limiting response time distribution.

In the simulation the number of servers equals  $N = 100$  servers, the 95% confidence intervals are computed based on 10 runs that each start from an empty system. The agreement with simulation is very good (except for high loads combined with a small  $d$ ) considering that we are simulating a system with only 100 servers.

In Figure 2.7b we look at the impact of the number of simulated servers  $N$  under high loads when  $d = 2$ . We note that the limiting distribution is not necessarily a good match for the tail probabilities of the response time when  $N$  is small, e.g.,  $N = 20$ , but the accuracy quickly improves as the number of servers increases.



(a) Power law job sizes with  
 $\bar{G}(w) = w^{-2}$ ,  $d = 2$ ,  $\rho = 0.8$

(b) Deterministic size one jobs,  
 $d = 2$ ,  $\lambda = 0.9$

Figure 2.8: Limiting response time distribution vs. simulation for  $N$  servers with power law and deterministic job sizes with mean 1. The full line represents the limiting response time distribution.

In Figure 2.7c and 2.7d we look at a similar setting as in Figure 2.7a and 2.7b, but the job sizes now follow a hyperexponential distribution with  $f = 1/2$  (see Section 2.9.2 for details). In this case the 95% confidence intervals are computed based on 25 runs. We note that even though the job sizes are now substantially more variable, the accuracy seems quite similar to the exponential case. Thus, more variable job size distributions do not necessarily imply worse accuracy for a fixed  $N$ .

Figure 2.8 illustrates the accuracy of the limiting response time distribution in case of power law and deterministic job sizes (computed via the fixed point iteration in Section 2.6.1). More specifically, for the power law distribution we used  $\bar{G}(w) = w^{-\beta}$  with  $\beta = 2$ . This implies that the mean job size is finite and equal to 2, while the variance of the job size distribution is infinite. In the deterministic case the job size equals 1. The figure indicates that somewhat larger  $N$  values are needed to closely match the limiting response time distribution compared to the (hyper)exponential case.

In Section 8.1 we further discuss the accuracy of the mean field model introduced in this chapter.

## 2.11 Conclusions and future work

In this chapter we studied the limiting workload and response time distribution of the  $\text{LL}(d)$  policy which assigns an incoming job to a server with the least work left among  $d$  randomly selected servers. We introduced a FPI to determine the limiting workload distribution for general job size distributions and any non-idling service discipline and studied its convergence. We derived a closed form expression for both the workload and response time distribution (for FCFS service) in case of exponential job sizes and indicated that these distributions can be computed easily by solving a set of ODEs for PH distributed job sizes.

We provided insight into the gains that can be expected when exact workload information is used instead of the coarser queue length information by comparing the performance of the  $\text{LL}(d)$  policy with the classic  $\text{SQ}(d)$  policy. Such a comparison is relevant to understanding the performance gains offered by schedulers implementing *late binding*. In this regard we demonstrated that late binding offers significant gains over  $\text{SQ}(d)$  for a wide range of arrival rates, even when taking the late binding overhead into account.

There are many other workload dependent load balancing policies which can be analysed using the same methodology as the one we used throughout this chapter. We develop this analysis in Chapters 3, 4 and 6. In particular in Chapter 6 we show that when we add some form of memory (consisting of idle servers) to the dispatcher, we can still obtain closed form solutions for the workload distribution.

Another possible, but challenging, direction for future work on the  $\text{LL}(d)$  policy is to perform a worst-case analysis as was done for the  $\text{SQ}(d)$  policy in [13].

While our formula for  $\bar{F}(w)$  holds for any work conserving scheduling policy, the response time is harder to compute in general. An interesting avenue to explore is to compute the response time for other scheduling policies. One method which can be used to this end is to write a simulation of the cavity queue which makes use of the (known) workload dependent arrival process  $\lambda(w)$ . In [68], the  $\text{LL}(d)$  policy for alternative scheduling policies is considered, but the author uses a simulation of the finite system to obtain the response time distribution.

# Chapter 3

## Performance of redundancy( $d$ ) with identical/independent replicas

The further a society drifts from  
the truth the more it will hate  
those that speak it.

---

George Orwell

### Abstract

Queueing systems with redundancy have received considerable attention recently. The idea of redundancy is to reduce latency by replicating each incoming job a number of times and to assign these replicas to a set of randomly selected servers. As soon as one replica completes service the remaining replicas are cancelled. Most prior work on queueing systems with redundancy assumes that the job durations of the different replicas are i.i.d., which yields insights that can be misleading for computer system design.

In this chapter we develop a differential equation, using the cavity method, to assess the workload and response time distribution in a large homogeneous system with redundancy without the need to rely on this independence assumption. More specifically, we assume that the duration of each replica of a single job is identical across the servers and follows a general service time

distribution. In addition, we develop a numerical method to compute  $\lambda_{\max}$ , the maximal arrival rate  $\lambda$  for which the system is still stable.

Simulation results suggest that the differential equation yields exact results as the system size tends to infinity and can be used to study the stability of the system. We also compare our system to the one with i.i.d. replicas and show the similarity in the analysis used for independent resp. identical replicas.

### 3.1 Introduction

Redundancy is regarded as an effective technique to reduce latency in a variety of systems including large scale computer clusters [80, 6, 77, 78]. The idea of redundancy is to create a number of replicas of each incoming job and to assign these replicas to a set of random servers. When the first of these replicas is processed by a server, the remaining replicas get cancelled. An attractive feature of this scheme is that the replicas can be assigned immediately without the need to consult the server states or the need to maintain such information. Queueing models to study the effect of redundancy on the job response time have been introduced recently (e.g., [39, 11]). One of the key assumptions to enable their analysis often exists in assuming that the processing times of the replicas i.i.d. across servers. While this may be applicable in some contexts, this assumption may result in misleading insights in a computer systems setting. For instance this i.i.d. assumption suggests that mean response time reduces as a function of the number of replicas (for sufficiently variable job sizes), while without such an assumption the mean response time may increase sharply if too many replicas are used.

In this chapter we present a FPE, based on the cavity process, to assess the workload and response time distribution of a queueing model with redundancy when the processing times of the replicas are assumed to be *identical* across servers as opposed to assuming they are i.i.d.. Next, we rewrite this FPE as a DIDE for general job sizes which have no atom in zero, which reduces to a DDE in case the job sizes are discrete. For PH distributed job sizes, we are able to simplify this DIDE further to an ODE. We conjecture that (when the queueing system is stable) this DIDE has a unique solution that corresponds to the limit of the workload distribution as the number of servers tends to infinity. We propose a numerical scheme to solve the DIDE and illustrate that its accuracy improves with the system size for various job

size distributions (i.e., for bounded Pareto, (hyper)exponential and deterministic job sizes) using simulation. We also show how this DIDE leads to a method to accurately obtain the stability region for a given model. Furthermore, we use this technique to obtain the equilibrium workload and response time distribution to study redundancy with identical replicas and compare it to redundancy with independent replicas. For independent replicas we rely on the method suggested in [39] to obtain the equilibrium workload and response time distribution which also provides a DIDE, we introduce it in our setting along with redundancy with identical replicas in order to illustrate the similarities/differences in the approach taken to derive it. This DIDE can be reduced to a DDE for discrete job sizes. When job sizes are PH-distributed, we can again simplify the associated DIDE to a set of ODEs.

Our main findings in case of identical replicas can be summarized as follows: When identical replicas are used, the stability region shrinks severely as  $d$  increases and depends on the higher moments of the job size distribution. As such replicating too much can easily cause system instability. More variable job size distributions tend to result in a larger stability region, but still cause larger response times when the system load is low. The mean and the variance of the response time in a system with redundancy typically remains low and increases sharply as the system gets close to becoming unstable. This increase is considerably sharper than in a system without redundancy. The mean response time tends to increase linearly with the SCV of the job size distribution. For small SCVs increasing the SCV may reduce the mean response time. Finally, the tails of the response time distribution often decay much faster compared to a system without redundancy. We further show that these insights are considerably different from what is observed in a system with independent replicas, where the stability region often increases as more replicas are used, the mean response time tends to decrease as the SCV increases, etc.

## 3.2 Structure of this chapter

We start by giving an informal overview of the results in this chapter in Section 3.3. The models considered in this chapter (redundancy with independent resp. identical replicas) are introduced in Section 3.4. The cavity processes associated to these queueing systems is presented in Section 3.5. The DIDEs are derived in Section 3.6 where we also take a closer look at the

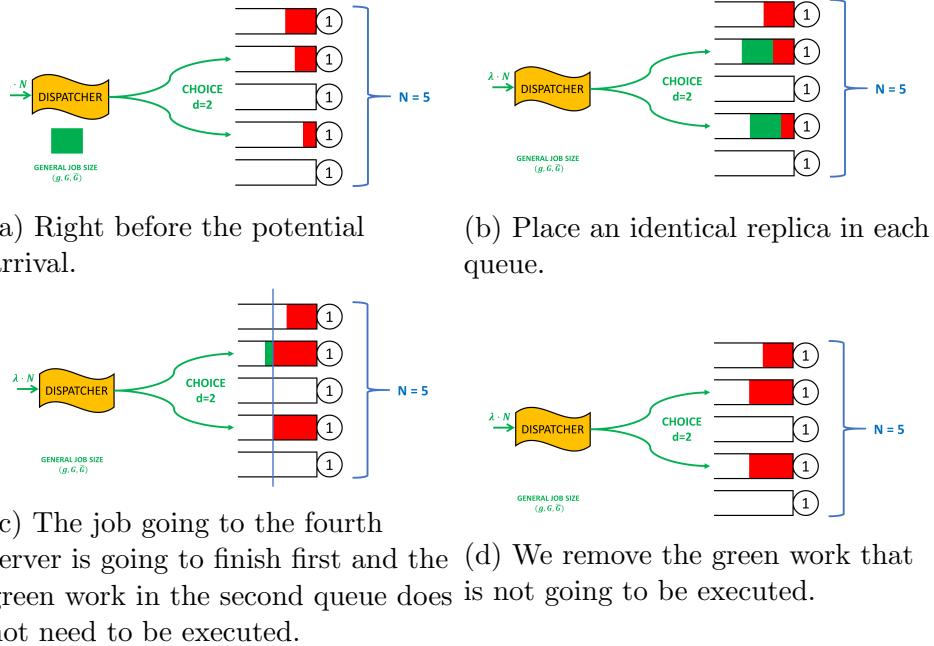


Figure 3.1: Graphical depiction of the Red(2) policy with  $N = 5$ , exponential job sizes with mean one and arrival rate  $\lambda N$ .

numerical method used to compute the equilibrium workload distribution. In Section 3.7 we show the accuracy of our suggested method by comparing with simulations of finite dimensional systems and validate our method to obtain the stability region by means of simulation. Numerical results on redundancy with identical replicas can be found in Section 3.8. In Section 3.9 we make a comparison between having independent and identical replicas. Section 3.10 discusses some future work.

### 3.3 Informal intuition

When a job arrives to the dispatcher for the Red( $d$ ) policy, we make  $d$  identical replicas of the incoming job and place one of the replicas on each of the selected servers. As soon as one server finishes the work on their replica of the job, all other replicas are cancelled and the job is considered *finished*. In Figure 3.1 we show what happens upon a job arrival for the Red(2) policy, we observe that one server simply gets the entire job added to its workload while

the other server works on the job until the least loaded server has finished the job, therefore we only add a part of the job size to its workload.

## A Word on Stability

When the amount of work coming into the system for each job is *known* as is the case for the LL( $d$ ) and SQ( $d$ ) policy, it is easy to compute the probability that a server is non-empty as we have :

$$\text{Arrival rate} \times \text{work from 1 job} = \text{Probability server is busy} \times \text{Server speed.}$$

However, for a policy such as Red( $d$ ), it is not clear how much work a single job brings in, therefore the probability that a server is busy can not easily be computed. One would expect that as the system tends to instability, that is, as  $\lambda \rightarrow 1^-$  for our system, the probability of creating *redundant work* becomes negligible, but as it turns out this is not at all true. We present a numerical method which enables us to compute the value of  $\lambda_{\max}$  which is the maximal arrival rate for which Red( $d$ ) is stable. We find that  $\lambda_{\max} \rightarrow 0^+$  as  $d \rightarrow \infty$ . Using another scheduling method such as Random Order Service (ROS) resolves this issue, but as of yet Red( $d$ ) with ROS has not been analysed (see [9] for more details on this issue).

## Analysis

For the analysis, we can again use a level-crossing argument (see also Section 2.3). Let us (for simplicity) assume  $d = 2$ , that is, we look at Red(2) (see also Figure 3.1). The down-crossing rate is given by  $f(w) = -\bar{F}'(w)$ , while for the up-crossing rate we again separately consider the case where the cavity queue is empty/busy upon a potential arrival.

1. When the cavity queue is empty (probability  $1 - \bar{F}(0)$ ), we find that irrespective of the work present in the other selected queue, we breach the workload level  $w$  if the job size exceeds  $w$ .
2. When the cavity queue has some workload  $u \in (0, w)$ , we consider three separate cases:
  - If the second queue's workload exceeds the workload  $u$  (probability  $\bar{F}(u)$ ), we simply require the incoming job to have a size which exceeds  $w - u$  (probability  $\bar{G}(w - u)$ ).

- If the second queue is idle (probability  $1 - \bar{F}(0)$ ) the job must have a size which exceeds  $w$ .
- If the second queue has some workload  $v \in (0, u)$ , we require the incoming job to have a size which exceeds  $w - v$ .

Summarizing all cases, we find from the level crossing argument that the following equality holds:

$$\begin{aligned} -\bar{F}'(w) &= \lambda \cdot d \cdot \left( (1 - \bar{F}(0))\bar{G}(w) + \int_0^w f(u)\bar{F}(u)\bar{G}(w-u) du \right. \\ &\quad \left. + \int_0^w f(u)(1 - \bar{F}(0))\bar{G}(w) du + \int_0^w f(u) \int_0^u f(v)\bar{G}(w-v) dv du \right) \end{aligned}$$

For the last integral we use Fubini followed by integration by parts to get rid of all occurrences of  $f(u)$  and lastly some simple algebra to simplify the result. This way we obtain that  $\bar{F}(w)$  satisfies the DIDE:

$$\bar{F}'(w) = -\lambda d \left[ \bar{G}(w)(1 - \bar{F}(w)) + \int_0^w (\bar{F}(w-u) - \bar{F}(w))\bar{F}(w-u)g(u) du \right].$$

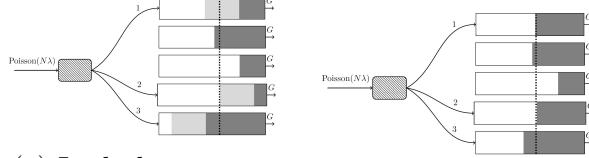
This generalizes to the general case of  $\text{Red}(d)$ , that is for  $\text{Red}(d)$  with identical replicas we have:

$$\bar{F}'(w) = -\lambda d \left[ \bar{G}(w)(1 - \bar{F}(w)) + \int_0^w (\bar{F}(w-u) - \bar{F}(w))\bar{F}(w-u)^{d-1}g(u) du \right].$$

For PH distributed job sizes one can get rid of the integral as we did for  $\text{LL}(d)$  in Section 2.3. However, we still have 2 major issues when trying to compute the stationary distribution :

- As we do not know the amount of work required to complete one job, we do not know the probability that a queue is non-empty, all we know is that if the system is stable,  $\bar{F}(0) \in (\lambda, \min\{1, \lambda \cdot d\})$ .
- For a certain choice of  $\lambda$  and  $d$  we do not know if the queueing system is stable.

For both these problems, we suggest to use a simple binary search, see Section 3.6.4 for more details.



(a) In dark gray, the workload right before the potential arrival occurs, in light gray the amount of work that arrives due to the potential arrival.

(b) In dark gray, the workload right after the potential arrival occurs, i.e. all light gray area right of the dotted line in Figure 3.2a.

Figure 3.2: Graphical representation of what happens at an arrival instant for  $\text{Red}_{\text{eq}}(d)$  with  $d = 3$  and  $N = 5$ .

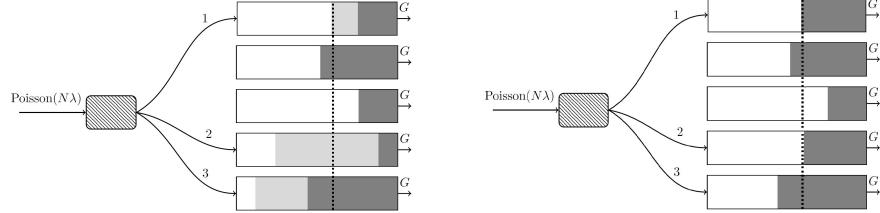
### 3.4 Model description

We consider a system with  $N$  identical servers (for large  $N$ ), each having an infinite waiting room. Arrivals occur according to a Poisson process with rate  $\lambda N$ . The service discipline at each server is assumed to be FCFS and jobs are processed at a constant rate 1. We use a general job size distribution as outlined in Section 1.4.

In this chapter, we consider 2 distinct policies:

**Redundancy-d with identical replicas ( $\text{Red}_{\text{eq}}(d)$ ) :** Each incoming job is replicated  $d$  times and each replica joins a random server (in total  $d$ , distinct, random servers receive an identical arrival). As soon as one replica finishes service, the remaining replicas are cancelled (whether in service or not). Cancellation is assumed to be immediate, although this assumption can be relaxed (see Section 3.6.2). It is important to emphasize that for  $\text{Red}_{\text{eq}}(d)$ , all  $d$  replicas of one job are assumed to be identical (i.e. equal in size).

**Redundancy-d with independent replicas ( $\text{Red}_{\text{iid}}(d)$ ) :** At each arrival instant, replicas are made and distributed in the same manner as for  $\text{Red}_{\text{eq}}(d)$ . The processing times of the  $d$  replicas of a job are however i.i.d. rather than identical, this model was studied in [39].



(a) In dark gray, the workload right before the potential arrival occurs, in light gray the amount of work that arrives due to the potential arrival.

(b) In dark gray, the workload right after the potential arrival occurs, i.e. all light gray area right of the dotted line in Figure 3.3a is now dark gray.

Figure 3.3: Graphical representation of what happens at an arrival instant for  $\text{Red}_{\text{iid}}(d)$  with  $d = 3$  and  $N = 5$ .

In Figure 3.2, we graphically show what happens at an arrival instant for  $N = 5$  and  $d = 3$  in the  $\text{Red}_{\text{eq}}(d)$  model. The dark gray area indicates actual workload while the light gray area indicates potential workload from the arrival. The same arbitrary amount of work is added to all (randomly) selected servers, this work is indicated in light gray in Figure 3.2a. All work that still needs to be done after the shortest queue finished serving the job may be discarded and the actual new workload of the queues is depicted in dark gray in Figure 3.2b.

In Figure 3.3, we show the events at an arrival instant for  $N = 5$  and  $d = 3$  for the  $\text{Red}_{\text{iid}}(d)$  model. An independent arbitrary amount of work is added to each selected queue. The workload at each chosen queue is then increased to match the workload at the server that finishes the new job first (which is the first queue in this case).

As in Chapter 2 the corresponding Markov processes only need to keep track of the workload at each of the  $N$  queues. We provide an analysis for  $\text{Red}_{\text{eq}}(d)$ , the policy  $\text{Red}_{\text{iid}}(d)$  has been studied in [39]. We restate their result for general job sizes using our notation in Proposition 29.  $\text{Red}_{\text{eq}}(d)$  is stable if  $\lambda \mathbb{E}[G] < 1/d$  and unstable for  $\lambda \mathbb{E}[G] \geq 1$ , its stability is unclear for  $\lambda \mathbb{E}[G] \in (1/d, 1)$ . It was shown in [39] that  $\text{Red}_{\text{iid}}(d)$  with exponential job sizes is stable iff  $\lambda \mathbb{E}[G] < 1$ , one would expect that the stability region grows as a function of the job size variability (note that, for deterministic job sizes, these policies are equivalent).

### 3.5 Cavity process

The cavity process methodology (see also Section 1.5) is used to analyse both systems. The cavity process intends to capture the evolution of the workload of one queue for the limiting system when the number of servers  $N$  tends to infinity.

- For  $\text{Red}_{\text{eq}}(d)$  we find that when a potential arrival of size  $X$  occurs to servers with workloads  $U_1, \dots, U_d$ , the workload  $U_1$  changes to:

$$\mathcal{Q}(U_1, \dots, U_d, X) = \max\{U_1, \min_{j=1}^d \{U_j\} + X\},$$

after the (possibly redundant) work of the arrival has been added.

- For  $\text{Red}_{\text{iid}}(d)$  we find that when a potential arrival of i.i.d. sizes  $X_1, \dots, X_d$  occurs to servers with workloads  $U_1, \dots, U_d$ , then the workload  $U_1$  becomes:

$$\mathcal{Q}(U_1, \dots, U_d, X_1, \dots, X_d) = \max\{U_1, \min_{j=1}^d \{U_j + X_j\}\}.$$

While Conjecture 1 was proven for Redundancy- $d$  with i.i.d. replicas and exponential job sizes in [83], it is not clear how to extend these results to more general job sizes nor to Redundancy- $d$  with identical replicas. One of the main issues for this extension is that it is not even clear for which arrival rates  $\lambda$  these policies are stable. Therefore, we assume Conjecture 1 holds throughout this chapter and show how to analyse both models under this assumption. Furthermore, we justify this assumption by means of simulation in Section 3.7.

We now characterize the evolution of the cavity process associated with the equilibrium environment process. Let  $f(t, w), t \in [0, \infty), w \in (0, \infty)$  describe the density at which the cavity queue, at time  $t$ , has workload  $w > 0$ . Note that  $f(t, \cdot)$  is not an actual pdf as the probability that the server is empty is non-zero. Let  $F(t, w) = F(t, 0) + \int_0^w f(t, u) du$  denote the cdf of the workload of a random server, here  $F(t, 0) = 1 - \int_0^\infty f(t, u) du$  is the probability that a random server is idle.

We define  $c_d(t, w, r)$  as the double density that, if a potential arrival occurs at time  $t$ , the queue at the cavity has workload  $w > 0$  and its workload is increased to  $r > w$  by the potential arrival. Lastly we let  $C_d(t, r)$  denote

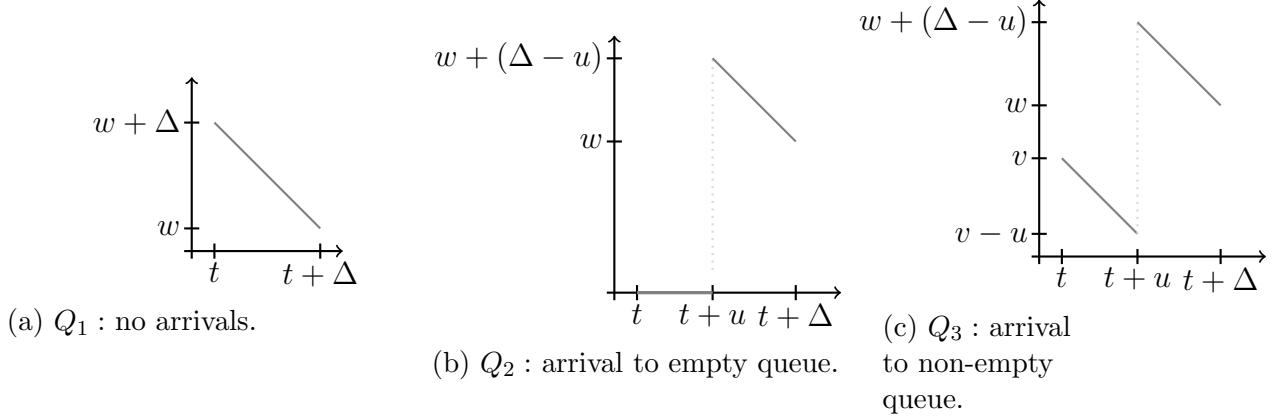


Figure 3.4: Graphical representation to illustrate (3.3), all ways one can have workload  $w$  at time  $t + \Delta$  (which are not  $o(\Delta)$ ).

the density at which, if a potential arrival occurs at time  $t$ , the queue at the cavity has workload 0 and its workload is increased to  $r > 0$ .

We now obtain a partial DIDE (PDIDE) which describes the transient evolution of the cavity queue as a function of  $c_d, C_d$ . The proof is similar to the proof of Theorem 8.

**Theorem 22.** *The evolution of the cavity process associated to the equilibrium environment process of the  $\text{Red}_{\text{eq}}(d), \text{Red}_{\text{id}}(d)$  policy is captured by the following set of equations:*

$$\frac{\partial f(t, w)}{\partial t} - \frac{\partial f(t, w)}{\partial w} = \lambda d \cdot \left( - \int_w^\infty c_d(t, w, r) dr + C_d(t, w) + \int_0^w c_d(t, u, w) du \right) \quad (3.1)$$

$$\frac{\partial F(t, 0)}{\partial t} = -\lambda d F(t, 0) + f(t, 0^+), \quad (3.2)$$

for  $w > 0$ , where  $f(x, z^+) = \lim_{y \rightarrow z^+} f(x, y)$ .

*Proof.* We first let  $t, w > 0$  and  $0 < \Delta < w$  be arbitrary. We now describe the possible evolution of the workload of the queue at the cavity in the interval  $[t, t + \Delta]$  s.t. it has exactly workload  $w$  at time  $t + \Delta$ . We write:

$$f(t + \Delta, w) = Q_1 + Q_2 + Q_3 + o(\Delta), \quad (3.3)$$

and describe how to obtain these  $Q_i$ .

- (Q<sub>1</sub>) First, we consider the case where the queue at the cavity has  $w + \Delta$  work at time  $t$  and no potential arrivals in  $[t, t + \Delta]$  make its workload increase. For this case we find:

$$Q_1 = f(t, w + \Delta) - \lambda d \int_0^\Delta \int_{w+\Delta-u}^\infty c_d(t+u, w + \Delta - u, r) dr du.$$

- (Q<sub>2</sub>) Second, we consider the case in which the queue at the cavity is empty at time  $t + u, u \in [0, \Delta]$  and its workload is increased to  $w + (\Delta - u)$  by a potential arrival. This happens with density:

$$Q_2 = \lambda d \int_0^\Delta C_d(t+u, w + (\Delta - u)) du.$$

- (Q<sub>3</sub>) Lastly, the queue at the cavity may be non-empty at time  $t + u, u \in [0, \Delta]$  and its workload increases to  $w + (\Delta - u)$  by a potential arrival. This case has density:

$$Q_3 = \lambda d \int_0^\Delta \int_u^{w+\Delta} c_d(t+u, v - u, w + (\Delta - u)) dv du.$$

We graphically show the three options,  $Q_1, Q_2$  and  $Q_3$  in Figure 3.4. Note that any other event involves having at least 2 arrivals which yields terms that are  $o(\Delta)$ . Subtracting  $f(t, w + \Delta)$ , dividing by  $\Delta$  and taking the limit  $\Delta \rightarrow 0$  on both sides of (3.3), we find that (3.1) indeed holds.

We have not yet considered the case  $w = 0$ , for this we need to consider which events on  $[t, t + \Delta]$  result in the workload of the queue at the cavity to be 0 at time  $t + \Delta$ . To this end we consider the following scenarios:

- The queue at the cavity is empty at time  $t$  and no actual arrivals occur in the interval  $[t, t + \Delta]$ . As each potential arrival is an actual arrival for empty queues, we find that this event occurs with probability  $F(t, 0)(1 - \lambda d\Delta)$ .
- The queue at the cavity is non empty at some time  $t + u, u \in [0, \Delta]$ , and decreases to zero by time  $t + \Delta$ . We find that this event occurs with probability  $\int_0^\Delta \int_0^{\Delta-u} f(t+u, v) dv du$ .

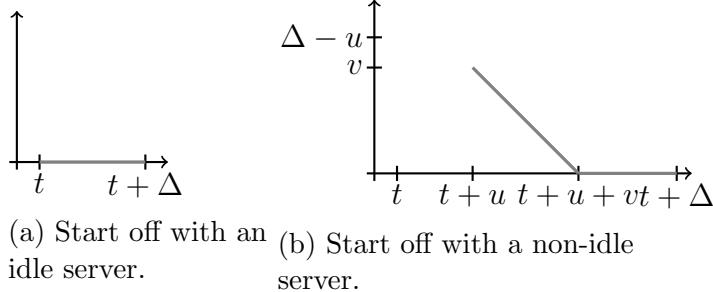


Figure 3.5: Graphical representation to illustrate all ways one can end up with an empty queue at time  $t + \Delta$  (which are not  $o(\Delta)$ ).

Putting these together, we find that the following equality holds for  $F(t + \Delta, 0)$ :

$$F(t + \Delta, 0) = F(t, 0)(1 - \lambda d\Delta) + \int_0^\Delta \int_0^{\Delta-u} f(t + v, u) dv du + o(\Delta).$$

Subtracting  $F(t, 0)$ , dividing by  $\Delta$  and taking the limit  $\Delta \rightarrow 0$  on both sides results in (3.2).  $\square$

**Remark 26.** *The PDIDE found in Theorem 22 could alternatively have been derived using the generalized Master Equation given by (7.25-7.26) in [79].*

We still require an exact expression for  $c_d$  and  $C_d$ . Moreover, we need an efficient method to compute the quantities  $\int_w^\infty c_d(t, w, r) dr$  and  $\int_0^w c_d(t, u, w) du$ . Therefore, in the next Proposition, we describe how to determine  $c_d$  and  $C_d$ .

**Proposition 23.** *For the Red<sub>eq</sub>(d) policy we have  $c_d(t, w, r) = c_{d,1}(t, w, r) +$*

$c_{d,2}(t, w, r) + c_{d,3}(t, w, r)$  such that:

$$\int_w^\infty c_{d,1}(t, w, r) dr = \bar{G}(w) f(t, w) (1 - \bar{F}(t, 0)^{d-1}) \quad (3.4)$$

$$\int_w^\infty c_{d,2}(t, w, r) dr = f(t, w) \bar{F}(t, w)^{d-1} \quad (3.5)$$

$$\int_w^\infty c_{d,3}(t, w, r) dr = (d-1) f(t, w) (\bar{F}(t, \cdot)^{d-2} f(t, \cdot) * \bar{G}(\cdot))(w) \quad (3.6)$$

$$\int_0^w c_{d,1}(t, u, w) du = g(w) \cdot (F(t, w) - F(t, 0)) (1 - \bar{F}(t, 0)^{d-1})$$

$$\int_0^w c_{d,2}(t, u, w) du = (g(\cdot) * f(t, \cdot) \bar{F}(t, \cdot)^{d-1})(w)$$

$$\begin{aligned} \int_0^w c_{d,3}(t, u, w) du &= (d-1) F(t, w) \cdot (g(\cdot) * f(t, \cdot) \cdot \bar{F}(t, \cdot)^{d-2})(w) \\ &\quad - (d-1) (g(\cdot) * F(t, \cdot) f(t, \cdot) \bar{F}(t, \cdot)^{d-2})(w), \end{aligned}$$

where  $(f_1 * f_2)(w) = \int_0^w f_1(u) f_2(w-u) du$  denotes the convolution product. These quantities can all be computed quickly which simplifies solving (3.1-3.2) significantly. Lastly, we have  $C_d(t, w) = F(t, 0) \cdot g(w)$ .

*Proof.* First we define  $c_{d,1}$ ,  $c_{d,2}$  and  $c_{d,3}$  as follows:

- At least one of the  $d-1$  independent random variables with law  $\mathcal{H}(t)$  is zero and the incoming job has size  $r$ . We find (for  $w < r$ ):

$$c_{d,1}(t, w, r) dr ds = g(r) f(t, w) (1 - \bar{F}(t, 0)^{d-1}) dr ds.$$

- The queue at the cavity is the queue with the minimal workload (i.e.  $w$ ) and the size of the arrival is exactly  $r-w$ :

$$c_{d,2}(t, w, r) dr dw = g(r-w) f(t, w) \bar{F}(t, w)^{d-1} dr dw.$$

- The queue with minimal workload has  $0 < u < w$  workload, where  $w$  is the workload of the queue at the cavity, and the arrival size is  $r-u$ :

$$c_{d,3}(t, w, r) dr ds = (d-1) f(t, w) \int_0^w g(r-u) \bar{F}(t, u)^{d-2} f(t, u) du dr dw.$$

now the claimed equalities all follow from direct computation and applying Fubini (which is allowed as all integrands are positive functions). It is trivial to derive the expression for  $C_d$ .  $\square$

The PDIDE (3.1-3.2) can now be solved using an (improved) Euler scheme. This result is also of interest to obtain a FPE for the equilibrium environment, i.e., workload distribution. In the subsequent section, we provide an efficient method to compute the equilibrium workload (and thus also response time) distribution.

## 3.6 Equilibrium regime

For the equilibrium we use the same notations as in the transient case, but we leave out the time dependence, i.e., we write  $f(w)$  instead of  $f(t, w)$  and set  $\frac{\partial f(w)}{\partial t} = 0$ . From the PDIDE describing the transient behaviour (3.1-3.2) we now derive a method to compute the equilibrium workload distribution.

**Proposition 24.** *The equilibrium workload distribution associated to the equilibrium environment of  $\text{Red}_{\text{eq}}(d)$  or  $\text{Red}_{\text{iid}}(d)$  satisfies the following equation:*

$$\bar{F}'(w) = -f(w) = -\lambda d \left( F(0) \bar{G}(w) + \int_0^w \int_w^\infty c_d(u, v) dv du \right) \quad (3.7)$$

*Proof.* Integrating (3.1) w.r.t.  $w$  and using (3.2) as a boundary condition ( $f(0^+) = \lambda d F(0)$ ), we obtain:

$$f(w) = \lambda d \left( F(0) - \int_0^w C_d(u) du + \int_0^w \int_u^\infty c_d(u, r) dr du - \int_0^w \int_0^r c_d(u, r) du dr \right). \quad (3.8)$$

We can further simplify (3.8) by applying Fubini to the term  $\int_0^w \int_0^r c_d(u, r) du dr$  to obtain (3.7).  $\square$

### 3.6.1 $\text{Red}_{\text{eq}}(d)$

From Proposition 24 we obtain a simple DIDE which can be solved numerically:

**Theorem 25.** *The stationary workload distribution associated to the equilibrium environment satisfies the following DIDE:*

$$\bar{F}'(w) = -\lambda d \left[ \bar{G}(w)(1 - \bar{F}(w)) + \int_0^w g(u)\bar{F}(w-u)^{d-1}(\bar{F}(w-u) - \bar{F}(w)) du \right]. \quad (3.9)$$

*Proof.* Using (3.4-3.6) we can simplify (3.7) applied to  $\text{Red}_{\text{eq}}(d)$  to obtain:

$$\begin{aligned} \bar{F}'(w) &= -\lambda d \left[ \bar{G}(w) ((1 - \bar{F}(0)^d) - \bar{F}(w)(1 - \bar{F}(0)^{d-1})) \right. \\ &\quad \left. + d(\bar{G} * f\bar{F}^{d-1})(w) - (d-1)\bar{F}(w)(\bar{G} * f\bar{F}^{d-2})(w) \right] \end{aligned} \quad (3.10)$$

We decompose the ccdf  $\bar{G} = \bar{G}_1 + \bar{G}_2$ , where  $\bar{G}_1$  corresponds to the continuous part and  $\bar{G}_2$  the discrete part. For the continuous part we apply integration by parts to obtain:

$$(\bar{G}_1 * f\bar{F}^n)(w) = \frac{1}{n+1} \left( \bar{G}_1(w)\bar{F}^{n+1}(0) - \bar{G}_1(0)\bar{F}^{n+1}(w) + (g_1 * \bar{F}^{n+1})(w) \right). \quad (3.11)$$

For the discrete part we first define  $\iota(w) = \sup\{n \mid a_n \leq w\}$ , where  $a_n$  for  $n \in \{1, 2, \dots\}$  are the atoms of the job size distribution. We find that the following equality holds:

$$(\bar{G}_2 * f\bar{F}^n)(w) = \frac{1}{n+1} \left( \bar{G}_2(w)\bar{F}^{n+1}(0) - \bar{G}_2(0)\bar{F}^{n+1}(w) + \sum_{n=0}^{\iota(w)} p_n \bar{F}^{n+1}(w - a_n) \right). \quad (3.12)$$

This equation follows by splitting the integral over the intervals  $[0, a_0]$ ,  $[a_0, a_1]$ ,  $\dots$ ,  $[a_{\iota(w)-1}, a_{\iota(w)}]$ ,  $[a_{\iota(w)}, w]$ . Putting integrands together and using the definition of  $g$ , we find that (3.11-3.12) simplifies to (3.9).  $\square$

The ccdf of the waiting time is given by  $\bar{F}_W(w) = \bar{F}(w)^d$ . The ccdf of the response time is given by the convolution of  $g$  with the ccdf of the waiting time, i.e.  $\bar{F}_R(w) = \bar{G}(w) + (g * \bar{F}_W)(w)$ .

The DIDE found in (3.9) can be simplified to a set of ODEs in case job sizes have a PH distribution.

**Corollary 26.** If job sizes have a PH distribution with parameters  $(\alpha, A)$  then (3.9) simplifies to the following ODE:

$$\begin{aligned}\bar{F}'(w) &= -\lambda d\alpha [e^{wA} \mathbf{1}(1 - \bar{F}(w)) + \xi_1(w) - \xi_2(w)\bar{F}(w)] \\ \xi'_1(w) &= A\xi_1(w) + \mu\bar{F}^d(w) \\ \xi'_2(w) &= A\xi_2(w) + \mu\bar{F}^{d-1}(w),\end{aligned}$$

with  $\mu = -A\mathbf{1}$  and boundary condition  $\xi_1(0) = \xi_2(0) = 0$ .

*Proof.* For PH distributed job sizes we find  $\bar{G}(w) = \alpha e^{wA} \mathbf{1}$  and  $g(w) = \alpha e^{wA} \mu$ . Applying this to (3.9) and splitting terms, we find:

$$\begin{aligned}\bar{F}'(w) &= -\lambda d\alpha \left[ e^{wA} \mathbf{1}(1 - \bar{F}(w)) + \int_0^w e^{uA} \mu \bar{F}(w-u)^d du \right. \\ &\quad \left. - \int_0^w e^{uA} \mu \bar{F}(w-u)^{d-1} du \bar{F}(w) \right].\end{aligned}\tag{3.13}$$

Letting  $\xi_1(w) = \int_0^w e^{uA} \mu \bar{F}^d(w-u) du = \int_0^w e^{(w-u)A} \mu \bar{F}^d(u) du$  we find:

$$\xi'_1(w) = A\xi_1(w) + \mu\bar{F}^d(w).$$

Analogously for  $\xi_2(w) = \int_0^w e^{uA} \mu \bar{F}^{d-1}(w-u) du$  we find  $\xi'_2(w) = A\xi_2(w) + \mu\bar{F}^{d-1}(w)$ .  $\square$

**Remark 27.** We can generalize the result in Corollary 26 to conditioned PH distributions. I.e. let  $X$  have PH distribution with parameters  $(\alpha, A)$  and  $a < b$  two positive numbers. If job sizes have distribution  $(X \mid a < X < b)$  we find:

$$\bar{G}(w) = \begin{cases} 1 & w < a \\ p\alpha(e^{Aw} - e^{Ab}) \mathbf{1} & a \leq w < b \\ 0 & b \leq w \end{cases}, \quad g(w) = \begin{cases} 0 & w < a \\ p\alpha e^{Aw} \mu & a \leq w < b \\ 0 & b \leq w \end{cases}$$

with  $p = \frac{1}{\alpha(e^{Aa} - e^{Ab})\mathbf{1}}$ . This allows us to find the following DDE for the equilibrium workload distribution:

$$\begin{aligned}\bar{F}'(w) &= -\lambda d(1 - \bar{F}(w)) & w \leq a \\ \bar{F}'(w) &= -\lambda d [p\alpha(e^{Aw} - e^{Ab}) \mathbf{1}(1 - \bar{F}(w)) + p\alpha(\xi_1(w) - \xi_2(w)\bar{F}(w))] & a < w \leq b \\ \bar{F}'(w) &= -\lambda dp\alpha(\xi_1(w) - \xi_2(w)\bar{F}(w)) & b < w,\end{aligned}$$

where  $\xi_1, \xi_2$  satisfy  $\xi_1(a) = \xi_2(a) = 0$  and

$$\begin{aligned}\xi'_1(w) &= A\xi_1(w) + e^{Aa}\mu\bar{F}(w-a)^d & a < w \leq b \\ \xi'_1(w) &= A\xi_1(w) + (e^{Ab}\bar{F}(w-b)^d - e^{Aa}\bar{F}(w-a)^d)\mu & b < w \\ \xi'_2(w) &= A\xi_2(w) + e^{Aa}\mu\bar{F}(w-a)^{d-1} & a < w \leq b \\ \xi'_2(w) &= A\xi_2(w) + (e^{Ab}\bar{F}(w-b)^{d-1} - e^{Aa}\bar{F}(w-a)^{d-1})\mu & b < w.\end{aligned}$$

**Remark 28.** It is not hard to see that when the job size distribution is some combination (product, sum, mixture, . . . ) of discrete and PH-distributed random variables, one can still obtain a DDE by generalizing Corollary 26.

### 3.6.2 Red<sub>eq</sub>(d) with delayed cancellation

We now assume there is some delay in the cancellation of jobs, i.e. after the first server finishes a job, the other  $d-1$  servers continue working on the job for some time  $\delta > 0$ . We find the following result:

**Proposition 27.** The stationary workload distribution associated to the equilibrium environment for Red<sub>eq</sub>(d) with a cancellation delay equal to  $\delta > 0$  satisfies the following DIDE:

$$\bar{F}'(w) = -\lambda d \left( \bar{G}(w) + \int_0^w g(w-u)\bar{F}(u) du - \bar{F}(w) \right) \quad w \leq \delta \quad (3.14)$$

$$\begin{aligned}\bar{F}'(w) &= -\lambda d \left( \bar{G}(w) - \bar{F}(w)\bar{G}(w-\delta) + \int_0^\delta \bar{F}(u)g(w-u) du \right. \\ &\quad \left. + \int_0^{w-\delta} \bar{F}(w-u-\delta)^{d-1}(\bar{F}(w-u) - \bar{F}(u))g(u) du \right) \quad w > \delta.\end{aligned} \quad (3.15)$$

*Proof.* It is not hard to see (analogue to Proposition 23) that in this case we have:

$$\begin{aligned}c_d(w, r) dr ds &= g(r-\delta)f(w)(1-\bar{F}(0)^{d-1}) dr dw \\ &\quad + g(r-w)f(w)\bar{F}(w)^{d-1} dr ds \\ &\quad + (d-1)f(w) \int_0^w g(r-u-\delta)\bar{F}(u)^{d-2}f(u) du dr dw.\end{aligned}$$

The result then follows using arguments similar to the proof of Theorem 25.  $\square$

**Remark 29.** As for the  $\text{Red}_{\text{eq}}(d)$  model, the ccdf of the waiting time is given by  $\bar{F}_W(w) = \bar{F}(w)^d$  and the response time by  $\bar{F}_R(w) = \bar{G}(w) + (g * \bar{F}_W)(w)$ .

The above DIDE simplifies to a DDE in case  $X$  has a PH distribution:

**Corollary 28.** If job sizes have a PH distribution with parameters  $(\alpha, A)$ , then the DIDE given by (3.14-3.15) simplifies to the following DDE:

$$\begin{aligned}\bar{F}'(w) &= -\lambda d(\bar{G}(w) + \alpha \xi_1(w) - \bar{F}(w)) & w \leq \delta \\ \bar{F}'(w) &= -\lambda d \left( \bar{G}(w) - \bar{F}(w)\bar{G}(w-\delta) + \alpha(\xi_1(w) + \xi_2(w) - \xi_3(w)\bar{F}(w)) \right) & w > \delta \\ \xi'_1(w) &= A\xi_1(w) + \bar{F}(w)\mu & w \leq \delta \\ \xi'_1(w) &= A\xi_1(w) & w > \delta \\ \xi'_2(w) &= \bar{F}(w-\delta)^{d-1}\bar{F}(w)\mu + A\xi_2(w) & w > \delta \\ \xi'_3(w) &= \bar{F}(w-\delta)^{d-1}\mu + A\xi_3(w) & w > \delta,\end{aligned}$$

with boundary condition  $\xi_1(0) = \xi_2(\delta) = \xi_3(\delta) = 0$ .

*Proof.* This follows from Proposition 27 by defining :

$$\begin{aligned}\xi_1(w) &= \int_0^{\min\{w,\delta\}} e^{(w-u)A} \bar{F}(u) du \mu \\ \xi_2(w) &= \int_0^{w-\delta} F(u)^{d-1} \bar{F}(u+\delta) e^{(w-u-\delta)A} du \mu \\ \xi_3(w) &= \int_0^{w-\delta} \bar{F}(u)^{d-1} e^{(w-u-\delta)A} du \mu.\end{aligned}$$

$\square$

### 3.6.3 $\text{Red}_{\text{iid}}(d)$

If we were to analyse  $\text{Red}_{\text{iid}}(d)$  in the same manner as we did for  $\text{Red}_{\text{eq}}(d)$ , we would again find that the ccdf of the workload distribution satisfies equation (3.7). In the case of  $\text{Red}_{\text{iid}}(d)$ , we find for arbitrary  $0 < w < r$ :

$$c_d(w, r) = f(w) \cdot [g(r-w)\bar{F}_{U+X}(r)^{d-1} + (d-1)\bar{G}(r-w)f_{U+X}(r)\bar{F}_{U+X}(r)^{d-2}], \quad (3.16)$$

where  $U$  and  $X$  are random variables with distribution  $F$  and  $G$ . Plugging (3.16) in (3.7) one finds an FDE describing the workload distribution. This equation is hard to solve and it is not immediately clear how to simplify it. However, as shown in [39] the following result holds (the proof of which we summarize in a few words, for more details see [39]):

**Proposition 29.** *The equilibrium workload distribution associated to  $\text{Red}_{\text{iid}}(d)$  satisfies the following DIDE:*

$$\bar{F}'(w) = -\lambda d \bar{F}_{R_1}(w)^{d-1} (\bar{F}_{R_1}(w) - \bar{F}(w)), \quad (3.17)$$

with  $\bar{F}_{R_1}(w) = \bar{G}(w) + (g * \bar{F})(w)$ , the probability that the response time is at least  $w$  if a job is sent to only 1 server.

*Proof.* If one were to send only one replica, this replica has a response time which is at least  $w$  if and only if either the job size is at least  $w$  or the job size is exactly  $u < w$  and the workload of the queue to which the replica is sent is at least  $w - u$ . This shows that  $\bar{F}_{R_1}(w) = \bar{G}(w) + (g * \bar{F})(w)$ .

The probability that a potential arrival increases the workload of the queue at the cavity from a value under  $w$  to a value above  $w$  is equal to the probability that all  $d - 1$  replicas which are sent to other queues have a response time which is at least  $w$ , the response time of the replica sent to the queue at the cavity is at least  $w$  and the queue at the cavity has a workload which is at most  $w$ . Note that the individual response times  $R_1$  can be written as a product because of the independence assumption of both the workload processes and the job sizes.  $\square$

For PH-distributed job sizes we find a result which is similar to Corollary 26:

**Corollary 30.** *If job sizes have a PH distribution with parameters  $(\alpha, A)$  then (3.17) simplifies to the following ODE:*

$$\begin{aligned} \bar{F}'(w) &= -\lambda d (\bar{G}(w) + \alpha \xi(w))^{d-1} (\bar{G}(w) + \alpha \xi(w) - \bar{F}(w)), \\ \xi'(w) &= A \xi(w) + \bar{F}(w) \mu, \end{aligned}$$

with  $\mu = -A \mathbf{1}$  and boundary condition  $\xi(0) = 0$ .

*Proof.* This follows in a similar manner as Corollary 26, but here we set  $\xi(w) = \int_0^w e^{(w-u)A} \mu \bar{F}(u) du$ .  $\square$

**Remark 30.** One can again generalize this result to conditional PH-distributions and combinations of PH-distributions and discrete distributions.

In case of independent replicas there are several possible definitions for the waiting time, that is, it could be the time until the first replica enters service or the time until the replica that first completes service enters service. As such we only consider the response time, the ccdf of which is:

$$\bar{F}_R(w) = (\bar{G}(w) + (g * \bar{F})(w))^d (= \bar{F}_{R_1}(w)^d).$$

### 3.6.4 Numerical considerations

Throughout all sections which contain numerical examples, we exclusively make use of job size distributions which have a mean equal to one. In particular we focus our attention to the following job size distributions:

- *Exponential Job Sizes* : Job sizes have an exponential distribution with mean equal to one.
- *Deterministic Job Sizes* : Job sizes are always equal to one.
- *Bounded Pareto Job Sizes* : Job sizes are bounded Pareto with lower bound 0.2, upper bound 72 and  $\alpha = 1.1$ , meaning  $\mathbb{E}[G] = 1$  and  $\mathbb{E}[G^2] = 10$ .
- *Hyperexponential Job Sizes* : Job sizes are hyperexponential with two phases and balanced means, chosen such that  $\mathbb{E}[G] = 1$ . When the SCV is not specified, we set  $\mathbb{E}[G^2] = 10$ .
- *Erlang Job Sizes* : Job sizes which have an Erlang- $k$  distribution with 2 up to 50 phases such that  $\mathbb{E}[G] = 1$ .

**Remark 31.** For bounded Pareto job sizes, we need to resort to solving a DIDE which is  $O(M^2)$ , for all other job size distributions the required computation time is only  $O(M)$ . Here  $M$  denotes the number of control points used to numerically represent  $\bar{F}$ .

## Obtaining the system load $\bar{F}(0)$

Note that Theorem 25, Corollary 26, Proposition 29 and Corollary 30 do not specify a boundary condition for  $\bar{F}(0)$ . This is not surprising as  $\bar{F}(0)$  corresponds to the unknown actual system load. We have the following Lemma which is used as a basis for an algorithm to find  $\bar{F}(0)$ :

**Lemma 31.** *Let  $\lambda > 0, d \in \{2, 3, \dots\}$  be such that the associated system is stable then the following are equivalent:*

- $\bar{F}(w)$  is a solution to (3.9) resp. (3.17) and  $\inf_{w>0} \bar{F}(w) = 0$ ,
- $\bar{F}(w)$  is the unique ccdf of the workload equilibrium for  $\text{Red}_{\text{eq}}(d)$  resp.  $\text{Red}_{\text{id}}(d)$ .

*Proof.* Obviously, if  $\bar{F}(w)$  is the ccdf of the workload equilibrium, it is a solution to the associated FPE and thus also of the associated DIDE, moreover  $\inf_{w>0} \bar{F}(w) = 0$  holds for any ccdf.

We should still show that the solution  $\bar{F}$  is indeed a ccdf if it is a solution to (3.9) or (3.17) and  $\inf_{w>0} \bar{F}(w) = 0$ . The uniqueness then follows from Conjecture 1. We show that for arbitrary  $t > 0$ , we have: if for all  $w < t$ ,  $\bar{F}'(w) \leq 0$  and  $\bar{F}(w) \geq 0$  then  $\bar{F}'(t) \leq 0$ , this shows that  $\bar{F}$  is a decreasing function as it is positive. For (3.9) it suffices to note that  $\bar{F}(t-u) - \bar{F}(t)$  appearing in the integral equals  $-\int_{t-u}^t \bar{F}'(v) dv \geq 0$ . For (3.17) we note that:

$$\begin{aligned} F_{R_1}(t) &= \bar{G}(t) + \int_0^t \bar{F}(u)g(t-u) du \\ &\geq \bar{G}(t) + \int_0^t \bar{F}(t)g(t-u) du \\ &= \bar{G}(t) + G(t)\bar{F}(t) \\ &\geq \bar{F}(t). \end{aligned}$$

We have thus shown that if  $\bar{F}$  satisfies the stated conditions, it is also a non-increasing function, as  $\inf_{w>0} \bar{F}(w) = 0$  we find that  $\lim_{w \rightarrow \infty} \bar{F}(w) = 0$ . This shows that  $\bar{F}$  is indeed a ccdf.  $\square$

Based on the result shown in Lemma 31 we obtain an algorithm which can be used to find the ccdf  $\bar{F}$  which is the solution for (3.9) and (3.17). Also, we present a simple method to check whether for a given job size distribution,  $\lambda$  and  $d$  the system is stable (i.e. the equilibrium workload distribution is not infinite). Note that the DIDE however still makes sense when the system is

unstable: we find the boundary condition  $\bar{F}(0) = 1$  and from this it is not hard to see that both for  $\text{Red}_{\text{eq}}(d)$  and  $\text{Red}_{\text{iid}}(d)$  we find  $\bar{F}(w) = 1$  for all  $w$ , i.e. the system load is almost surely infinite.

We employ the following bisection algorithm to find the value of  $\bar{F}(0)$  for which the associated solution satisfies  $\inf_{w>0} \bar{F}(w) = 0$ :

1. Set  $\text{lb} = 0$  and  $\text{ub} = 1$ ,
2. Compute  $y = \inf_{w>0} \bar{F}(w)$ , where  $\bar{F}(w)$  is computed as the solution of (3.9) or (3.17), with boundary condition  $\bar{F}(0) = x_0 = \frac{\text{lb}+\text{ub}}{2}$ .
3. Set  $\text{lb} = x_0$  if  $y < 0$  otherwise set  $\text{ub} = x_0$ , return to Step 2.

Due to Lemma 31, we are certain this algorithm converges, provided that  $\inf_{w>0} \bar{F}(w)$  is increasing as a function of the boundary condition  $\bar{F}(0)$ . Actually all we need is if  $\bar{F}_1(0) < \bar{F}_2(0)$  and  $\inf_{w>0} \bar{F}_2(w) < 0$  then  $\inf_{w>0} \bar{F}_1(w) < 0$ . Unfortunately this statement appears to be hard to prove and we only managed to confirm this numerically (see also Figure 3.8 and Section 8.2).

We now provide some deeper insight into how well the algorithm performs. For this discussion let us focus on  $\text{Red}_{\text{eq}}(d)$  and note that the discussion for  $\text{Red}_{\text{iid}}(d)$  is completely analogous. We need to numerically solve (3.9) for each step of the algorithm, which takes  $O(N^2)$  resp.  $O(N)$  time for continuous resp. discrete or PH distributed job sizes (use Corollary 26). For a function  $f$ , we let  $\|f\|_\infty = \sup_{w>0} |f(w)|$  denote its supremum norm. In Figure 3.6, we first compute the limiting distribution  $\bar{F}_\infty$  which satisfies (3.9) and  $\bar{F}(0)$  is chosen such that  $|\inf_{w>0} \bar{F}(w)| < 10^{-10}$ . We let  $\bar{F}_n$  denote the solution to (3.9) after  $n$  steps have been taken in the algorithm. We show the difference  $\|\bar{F}_n - \bar{F}_\infty\|_\infty$  for  $n$  varying from 1 to 34 (note that 34 is the first value  $n$  for which  $2^{-n} < 10^{-10}$ ). Figure 3.6a is for exponential job sizes,  $\lambda = 0.48$  and  $d = 2, 3, 4, 5$  while Figure 3.6b is for  $d = 2$ ,  $\lambda = 0.7$  and varying (i.e. exponential, deterministic, bounded Pareto and hyperexponential) job sizes. We observe that the accuracy of  $\bar{F}_n$  increases exponentially which means that  $\|\bar{F}_n - \bar{F}_\infty\|_\infty \approx |\bar{F}_n(0) - \bar{F}_\infty(0)|$ . We now show  $\bar{F}_n$  for  $n = 1, \dots, 34$  in Figure 3.7 (on a logarithmic scale, negative values are discarded). Rather than labelling each line, we increase the linewidth as  $n$  increases. We clearly observe that as  $n$  increases,  $\bar{F}_n$  gets increasingly closer to being an actual ccdf. Moreover we see that none of the lines cross, which supports the claim that if  $\bar{F}_1(0) < \bar{F}_2(0)$ , then for all  $w : \bar{F}_1(w) < \bar{F}_2(w)$ .

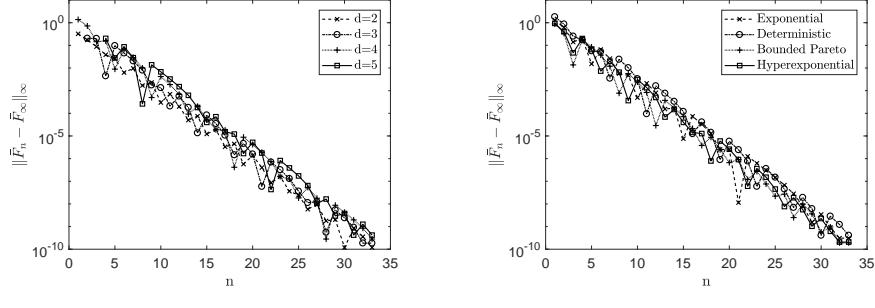


Figure 3.6: Convergence of  $\bar{F}_n$ , the ccdf found after  $n$  steps to the limiting distribution ccdf  $\bar{F}_\infty$ .

In Figure 3.8 we show  $\inf_{w>0} \bar{F}(w)$ , where  $\bar{F}$  is the solution of (3.9) as a function of the used initial value  $\bar{F}(0)$ . In Figure 3.8a, job sizes are deterministic,  $\lambda = 0.4$  and  $d = 2, 3, 4, 5$ . Here we observe that for certain values of  $\lambda, d$  and  $\bar{F}(0)$  the infimum might be  $-\infty$ , but it is clearly monotone increasing and close to linear with a steeper ascend close to  $\bar{F}(0) = 1$ . In fact, all of these curves converge to 1 as  $\bar{F}(0)$  converges to 1 and the incline close to 1 is so steep that it is not even visible in Figure 3.8. In Figure 3.8b, bounded Pareto job sizes are used with  $d = 2$  and  $\lambda = 0.1, 0.3, 0.5, 0.7$ , we observe the same behaviour as for deterministic job sizes: the trajectory is close to linear with a steep incline when  $\bar{F}(0)$  gets close to one. This curve, let us denote it by  $y = f(x)$ , is in fact the one for which we need to find the value  $x_0$  that satisfies  $f(x_0) = 0$  (i.e. this value  $x_0$  is exactly the  $\bar{F}(0)$  boundary condition for which  $\bar{F}$  becomes a ccdf). The system is deemed unstable if and only if  $y = f(x)$  does not cross zero in  $[0, 1]$  and this happens when  $f(x) < 0$  for all  $x \in [0, 1]$ , meaning there is a discontinuity in 1 as we necessarily have  $f(1) = 1$  as mentioned before. Moreover, as  $y = f(x)$  is a curve which is close to linear, better (i.e. faster) algorithms could be used to obtain a root of  $f$ , e.g. a simple Newton iteration would converge extremely fast in this case.

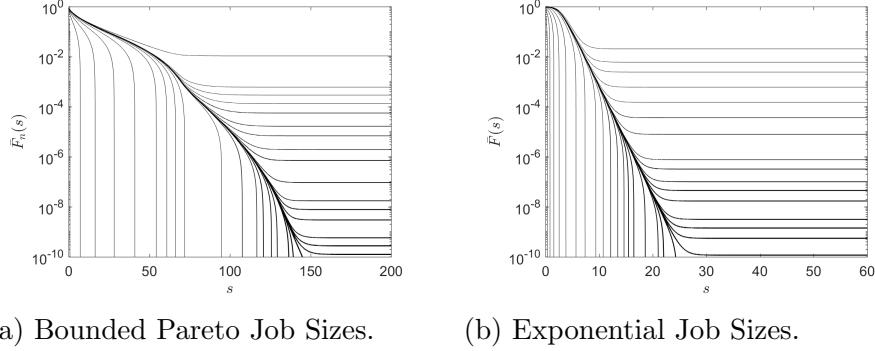


Figure 3.7: Plot of  $\bar{F}_n$  with  $n = 1, \dots, 34$ ,  $\lambda = 0.7$  and  $d = 2$  for bounded Pareto resp. exponential job size distributions. The linewidth is increased with  $n$ .

## Obtaining the stability region $\lambda_{\max}$

Let  $\lambda_{\max}$  be defined as the highest value value of  $\lambda$  for which a system is still stable. That is, for any  $\lambda < \lambda_{\max}$  the system is stable whilst for any  $\lambda \geq \lambda_{\max}$  it is unstable. We know that for  $\text{Red}_{\text{eq}}(d)$ ,  $\lambda_{\max} \in [1/d, 1]$ . To obtain an algorithm which allows to compute  $\lambda_{\max}$  we note that if we pick  $\varepsilon > 0$  small and we find  $\lambda > 0$  such that for this  $\lambda$  and  $\bar{F}(0) = 1 - \varepsilon$  we have  $\inf_{w>0} \bar{F}(w) = 0$ , then the system load for this  $\lambda$  is  $1 - \varepsilon$ . If we now pick  $\varepsilon > 0$  small enough this means that the system is close to instability, and this value can then be used as an approximation for  $\lambda_{\max}$ . We verify this method in Section 3.7.2 where we show that the approximation of  $\lambda_{\max}$  obtained in this manner is at least accurate up to 0.001.

## 3.7 Validation of the model

### 3.7.1 Finite system accuracy

In this section we use the algorithm presented in Section 3.6.4 to find the limiting workload distribution for  $\text{Red}_{\text{eq}}(d)$ , where we find the value of  $\bar{F}(0)$  such that  $\bar{F}$  is a ccdf.

We compare the equilibrium workload distribution with the simulated workload distribution for a finite system with  $N$  servers. We do this for 4 of the main job size distributions: exponential, deterministic, bounded Pareto

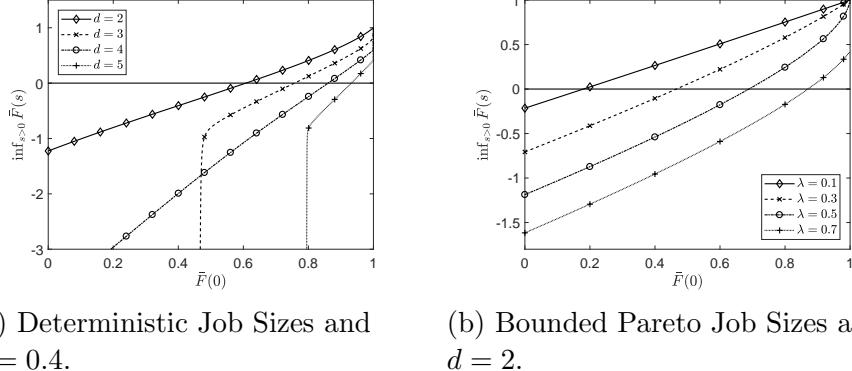


Figure 3.8: Plot of  $\inf_{w>0} \bar{F}(w)$  as a function of  $\bar{F}(0)$  for the solution of (3.9).

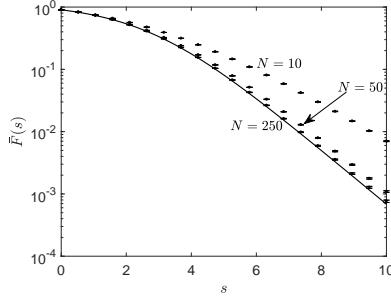
and hyperexponential. All simulation runs simulate the system up to time  $t = 10^7/N$  and use a warm-up period of 30%. We simulate a system of  $N = 10, 50, 250$  servers. The results are shown in Figure 3.9. We see that as  $N$  increases the approximation provided by the DIDE becomes more accurate (which supports Conjecture 1). Note that a similar figure can easily be made for the response time distribution and  $\text{Red}_{\text{iid}}(d)$ .

### 3.7.2 Stability region

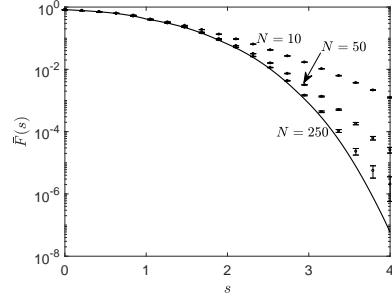
In this section we apply the algorithm presented in Section 3.6.4 to obtain the maximum value  $\lambda_{\max}$  for which the system is still stable. For this purpose, we first compute the value of  $\lambda_{\max}$  for a system with  $d = 2$  and deterministic, exponential resp. hyperexponential job sizes. We find:

- $\lambda_{\max} = 0.80554\dots$  for deterministic job sizes,
- $\lambda_{\max} = 0.81669\dots$  for exponential job sizes,
- $\lambda_{\max} = 0.83441\dots$  for hyperexponential job sizes.

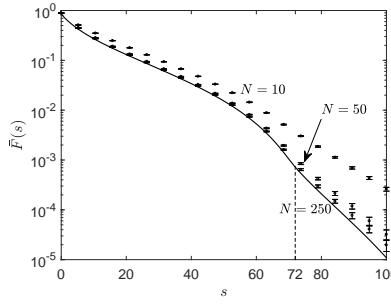
Note that in this example  $\lambda_{\max}$  increases as the job size variability increases. For each distribution, we set  $\lambda = \lambda_{\max} - 0.001$  and simulate a system with  $N = 300$ , arrival rate  $\lambda$  and the corresponding job size distribution for a time of  $2 \cdot 10^4$ . We observe in Figure 3.10a that the system indeed appears to be stable. In Figure 3.10b we observe that, when taking arrival rate



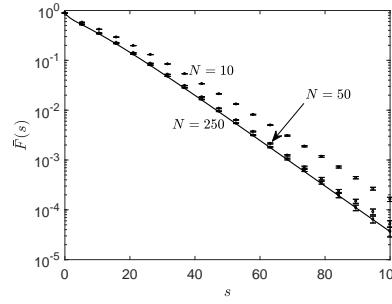
(a)  $\lambda = 0.7$ , exponential job sizes.



(b)  $\lambda = 0.6$ , deterministic job sizes.



(c)  $\lambda = 0.7$ , bounded Pareto job sizes ( $\text{max}=72$ ).



(d)  $\lambda = 0.7$ , hyperexponential job sizes.

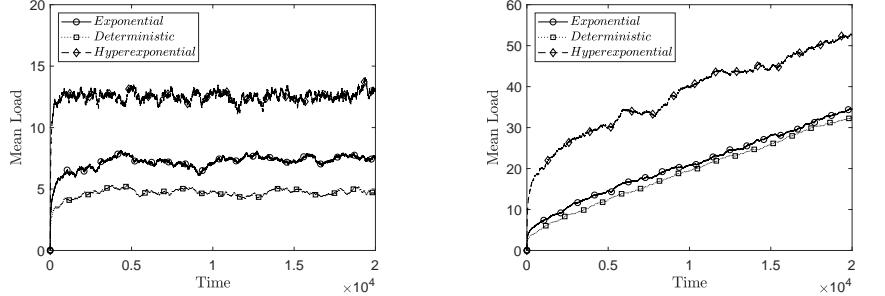
Figure 3.9: For the  $\text{Red}_{\text{eq}}(d)$  policy: Limiting workload distribution vs. simulation for  $N$  servers. The full line represents the solution of the DIDE/DDE/ODE, which is compared with the simulated 95% confidence intervals.

$\lambda = \lambda_{\text{max}} + 0.001$  and simulating the system with  $N = 300$  seems to result in an unstable system for each job size distribution. This suggests that our method to obtain  $\lambda_{\text{max}}$  is indeed quite accurate.

## 3.8 Numerical experiments for $\text{Red}_{\text{eq}}(d)$

### 3.8.1 Mean response time and workload distribution

In Figures 3.11 and 3.12 we show the actual workload  $\bar{F}(0)$  and the mean response time  $\mathbb{E}[R] = 1 + \int_0^\infty \bar{F}(w)^d dw$  of the  $\text{Red}_{\text{eq}}(d)$  policy as a function



(a) Arrival rate  $\lambda = \lambda_{\max} - 0.001$ , where  $\lambda_{\max}$  depends on the job size distribution.  
(b) Arrival rate  $\lambda = \lambda_{\max} + 0.001$ , where  $\lambda_{\max}$  depends on the job size distribution.

Figure 3.10: The mean workload of a simulated system with  $N = 300$  servers,  $d = 2$  and deterministic, exponential resp. hyperexponential job sizes.

of the arrival rate  $\lambda$  (recall  $\mathbb{E}[G] = 1$ ). From Figure 3.11a, it is clear that the stability region not only depends on the mean and the variance of the job size distribution, but also on higher moments (as  $\mathbb{E}[G^2] = 10$  for both the bounded Pareto and hyperexponential job sizes, see also Figure 3.15b). This makes the question of stability for  $\text{Red}_{\text{eq}}(d)$  for general job size distributions a hard problem (which in turn makes proving Conjecture 1 hard). We can already infer from the plot that the more variable the job size distribution, the lower the associated workload. From Figure 3.11b, it is obvious that  $\lambda_{\max}$  (defined as the supremum of the arrival rates  $\lambda$  for which  $\bar{F}(0) < 1$ ) decreases and the workload increases as a function of  $d$  (we have numerically verified that this also holds for the other job size distributions considered). For a more detailed discussion on  $\lambda_{\max}$ , see Section 3.8.3. Note that, as one would expect from a system that employs redundancy, the workload increases as a concave function as the arrival rate  $\lambda$  increases.

We show in Figure 3.12a that, despite the fact that the workload for the less variable jobs is consistently higher than that of the more variable ones, the same does not hold for the response times. We see that adding variability to the job size distribution also increases the mean response time (for  $\lambda$  sufficiently bounded away from instability). From Figure 3.12b it is clear that only for small values of  $\lambda$  there is a reduction in response time by increasing  $d$ : this reduction is due to the fact that for small arrival rates a

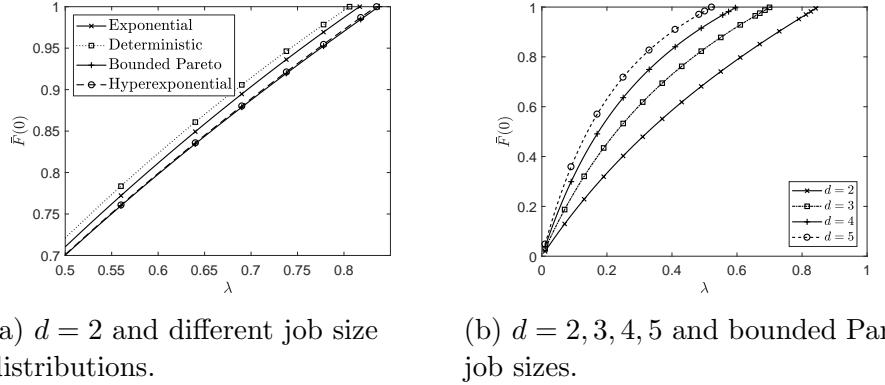


Figure 3.11: Workload  $\bar{F}(0)$  as a function of the arrival rate  $\lambda$  for  $\text{Red}_{\text{eq}}(d)$ .

job is more likely to find an idle server by increasing  $d$ , but as  $\lambda$  increases higher values of  $d$  cause too much extra load on the servers which causes an increased response time. In all plots in Figure 3.12 we observe that mean response times stay relatively small until  $\lambda$  is close to  $\lambda_{\max}$  at which point the mean response time explodes to infinity. This effect is more visible as the SCV of the jobs decreases and as the value of  $d$  increases.

### 3.8.2 Impact of the job size variability

We now investigate how well  $\text{Red}_{\text{eq}}(d)$  behaves as a function of the job sizes distribution's SCV. In Figure 3.13 we show the mean response time as a function of the job sizes' SCV. On  $[0, 1]$  we use deterministic, Erlang and exponential job sizes with mean one (see Figure 3.13a), while on  $(1, 40]$  we use a hyperexponential distribution with balanced means and mean one (see Figure 3.13b). In both figures, we fix  $\lambda = 0.45$  and  $d = 2, 3, 4, 5$ . We observe that the mean response time generally increases more or less linearly as a function of the SCV. The most notable exception occurs when  $d = 5$  and the SCV is close to zero. In this case  $\mathbb{E}[R]$  decreases as the system with deterministic jobs is close to instability and increasing the job size variability somewhat increases the value of  $\lambda_{\max}$ , which causes  $E[R]$  to decrease. Moreover, we observe that increasing  $d$  in case of deterministic job sizes also increases the mean response time, whilst for large SCV this is not necessarily the case. This makes sense as for jobs with low variability, the risk of picking a server that is serving a large job is smaller than for more variable jobs,

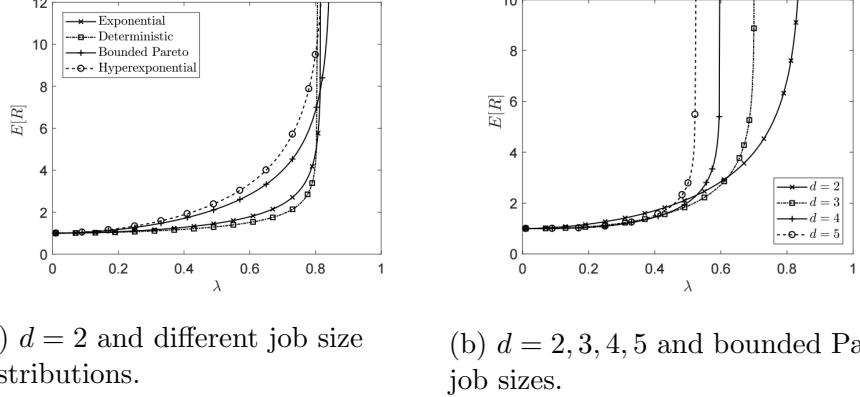


Figure 3.12: Mean response time  $E[R] = (1 + \int_0^\infty \bar{F}(w)^d ds)$  as a function of the arrival rate  $\lambda$  for  $\text{Red}_{\text{eq}}(d)$ .

meaning there is less incentive to increase  $d$  and increasing  $d$  also increases the amount of work on the servers.

In Figure 3.14 we make the same plots as in Figure 3.13, but now instead of fixing  $\lambda$  and taking multiple values of  $d$ , we fix  $d = 2$  and consider multiple values of  $\lambda$ . As expected, we observe that increasing  $\lambda$  also increases the mean response time and the slope of  $E[R]$  as a function of the SCV. Moreover we again observe that a decrease for low job size variability occurs when the system is close to instability.

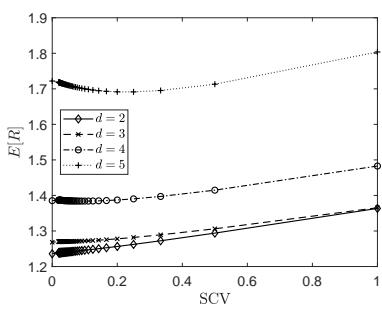
### 3.8.3 Stability

Let  $\bar{F}$  satisfy (3.9), we then find for all  $t$  for which  $\bar{F}(w) \geq 0, w \in [0, t]$ :

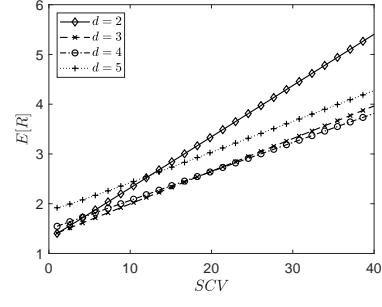
$$\bar{F}'(w) \leq -\lambda d \bar{G}(w)(1 - \bar{F}(w)).$$

This shows that, if there exists an  $\varepsilon > 0$  for which  $\bar{G}(\varepsilon) > 0$  (i.e. job sizes are not identically zero) and  $\bar{F}(0) < 1$ , then in the limit  $d \rightarrow \infty$  we find that  $\bar{F}(w)$  falls off at an unbounded speed. This shows that as  $d$  tends to infinity,  $\lambda_{\max}$  tends to zero. It is also intuitively clear that this is the case as for  $d = N$  we find that  $\text{Red}_{\text{eq}}(d)$  becomes an M/G/1 queue with arrival rate  $\lambda N$  where  $N$  tends to infinity.

This fact is also reflected in Figure 3.15a, where we show the evolution of  $\lambda_{\max}$  as a function of  $d$  for  $\text{Red}_{\text{eq}}(d)$  with deterministic job sizes. We observe



(a) Erlang job sizes with mean one and SCV on the  $x$ -axis.



(b) Hyperexponential job sizes with balanced means, mean one and the SCV on the  $x$ -axis.

Figure 3.13: Mean response time  $E[R] = (1 + \int_0^\infty \bar{F}(w)^d ds)$  as a function of the job sizes' SCV for  $d = 2, 3, 4, 5$  and  $\lambda = 0.45$ .

that for  $d = 1$ ,  $\lambda_{\max} = 1$  (naturally as this is simply an M/D/1 queue), as  $d$  increases there is first a sharp drop in  $\lambda_{\max}$  until  $\lambda_{\max} \approx 0.2$  around  $d = 20$  after which we see that the curve slowly converges to its horizontal asymptote at  $\lambda_{\max} = 0$ .

In Figure 3.15b, we compare the value of  $\lambda_{\max}$  for other job sizes to  $\lambda_{\max}$  for deterministic job sizes. We observe that the difference starts at zero (as for any M/G/1 queue  $\lambda_{\max} = 1$ ), then jumps up for  $d = 2$  and  $d = 3$  after which it decays to zero. We observe that the difference in stability region increases as the tail of the job size distribution is more fat. The difference in  $\lambda_{\max}$  is however fairly modest (no larger than 0.06).

### 3.8.4 Variance of the response time distribution

We now take a closer look at the behaviour of the variance of the response time distribution. To compute the variance, it is best to first compute the ccdf of the squared response time  $\bar{F}_{R^2}$  and then integrate, i.e. we compute the variance as:

$$\text{Var}(R) = \int_0^\infty \bar{F}_{R^2}(w) dw - \left( \int_0^\infty \bar{F}_R(w) ds \right)^2.$$

This is numerically more stable as it avoids the need to differentiate  $\bar{F}_R$  to obtain its density  $f_R$ . Do note that the computation of  $\bar{F}_{R^2}$  requires

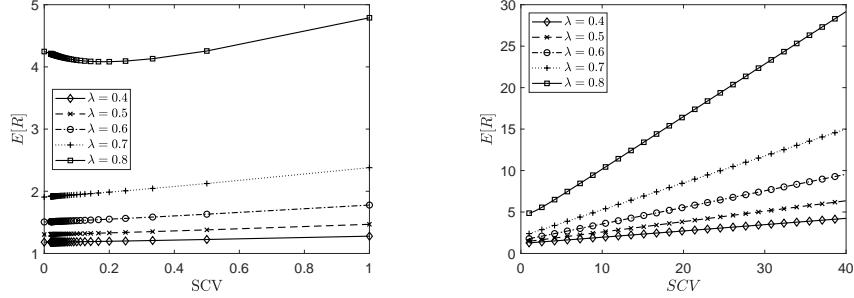


Figure 3.14: Mean response time  $E[R] = (1 + \int_0^\infty \bar{F}(w)^d ds)$  as a function of the job sizes' SCV.

a quadratically wider  $w$  range  $[0, w_{\max}]$  than  $\bar{F}_R$  in order to ensure that  $\bar{F}_{R^2}(w_{\max})$  is sufficiently small.

In Figure 3.16 we show the variance of the response time distribution as a function of the arrival rate  $\lambda$  for deterministic and exponential job sizes. We observe in Figure 3.16a that  $\text{Var}(R)$  remains very small until it explodes when  $\lambda$  approaches  $\lambda_{\max}$ . As long as  $\lambda < \lambda_{\max} - 0.01$ , we observe that  $\text{Var}(R) < 1$ . When taking a closer look at the curves for small  $\lambda$ , we see that the variance decreases as  $d$  increases, this is due to the fact that, for low loads, having more replicas increases the chance of finding an empty server. However, as  $d$  increases,  $\lambda_{\max}$  decreases which makes  $\text{Var}(R)$  explode to infinity faster.

For exponential job sizes (see Figure 3.16b), we also observe that the variance explodes as  $\lambda$  approaches  $\lambda_{\max}$  and for small  $\lambda$  we still have a higher variance for smaller  $d$ . We see that for small values of  $\lambda$  the variance stays around 1 (this is due to the fact that for small loads all incoming jobs have a high probability of finding an idle server). The variance does however increase more quickly than for deterministic job sizes. Whereas for deterministic job sizes the workload at all queues increases at a similar speed, for an exponential job size distribution the workloads at the different servers will be less balanced as an arrival of a large job drives the workload of the  $d$  selected servers up by a large amount.

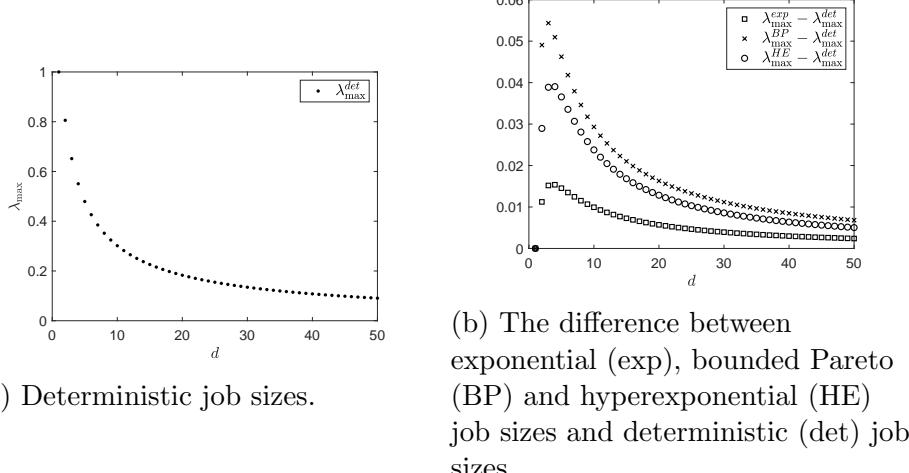


Figure 3.15: The evolution of  $\lambda_{\max}$  as a function of  $d$  for deterministic, exponential, bounded Pareto and hyperexponential job sizes in the  $\text{Red}_{\text{eq}}(d)$  model.

### 3.8.5 Tail of the Response Time distribution

In Figure 3.17 we show  $\bar{F}_R$  for  $d = 1, \dots, 6$  and  $\lambda = 0.48$ . We note that for both exponential and bounded Pareto job sizes, the system becomes unstable for  $d \geq 7$ , therefore these are no longer shown. We see in both Figures 3.17a and 3.17b that for identical replicas ( $d \geq 2$ ) the tail of the response time distribution is lighter than for a classic M/G/1 queue ( $d = 1$ ). However, the probability of having a very small response time is larger for the M/G/1 queue than for the case of identical replicas, especially for  $d = 6$ . This is due to the fact that, while replicas decrease the probability that a job ends up in a long queue, the queues are more heavily loaded, which decreases the probability of finding a queue with a very small workload (especially for  $d = 6$  as the system is close to instability in this case). This figure also confirms that as the job size variability increases, the gain from having replicas increases.

In Figure 3.18a we compare the ccdf for various job size distributions,  $d = 2$  and  $\lambda = 0.5$ . The tails behave as expected: the fatter the tail of the job size distribution, the fatter the tail of the response time distribution. However what is interesting to note is that for sufficiently small values of  $w$  (around  $w < 4$ ), the value of  $\bar{F}(w)$  is greater for the less variable job sizes. Moreover, when looking at Figure 3.18b, we see that this effect is

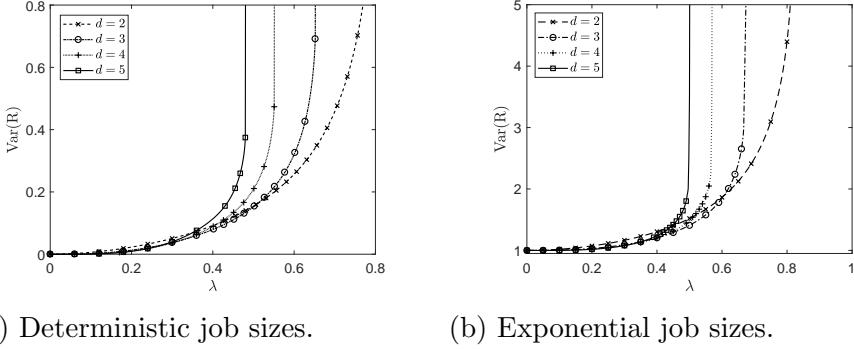


Figure 3.16: The variance of the Response time distribution as a function of the arrival rate  $\lambda$  for  $d = 2, 3, 4, 5$  in the  $\text{Red}_{\text{eq}}(d)$  model.

strengthened when  $\lambda$  increases. This can be understood by recalling that  $\lambda_{\max}$  is smaller for less variable job size distributions.

### 3.8.6 $\text{Red}_{\text{eq}}(d)$ with cancellation delay

In this subsection, we look at the impact of having a cancellation delay on the performance of  $\text{Red}_{\text{eq}}(d)$ . In Figure 3.19 we take  $\lambda = 0.7, d = 2$  and consider 4 distributions for  $X$ : Erlang with 2 phases, mean one and SCV  $1/2$ , Exponential and Hyperexponential with balanced means and  $SCV = 2$  and 3. We observe in Figure 3.19a that the system load increases in a concave manner as the cancellation delay  $\delta$  increases. Moreover we observe that as job sizes are more variable, this increase is less steep. Further, we observe in Figure 3.19b, that while for a small delay, the mean response time increases as the job size variability increases, this relation is reversed for a delay of  $\delta \geq 0.8$ . Finally we note that the system becomes unstable as we increase the cancellation delay  $\delta$ , and the maximum point of stability  $\lambda_{\max}$  increases as the SCV of the job size increases.

## 3.9 Comparison $\text{Red}_{\text{eq}}(d)$ and $\text{Red}_{\text{iid}}(d)$

In this section, we take a look at  $\text{Red}_{\text{iid}}(d)$  by comparing it to  $\text{Red}_{\text{eq}}(d)$ . We first show that  $\text{Red}_{\text{iid}}(d)$  always performs better than  $\text{Red}_{\text{eq}}(d)$ . Afterwards, we revisit some of the numerical experiments from Section 3.8 to indicate key differences between the two models. These results show why it may be

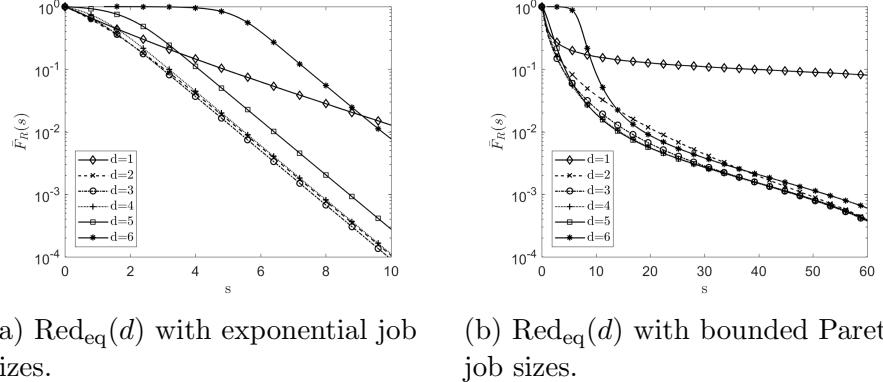


Figure 3.17: Logarithmic plot of the response time distribution for  $\text{Red}_{\text{eq}}(d)$  with exponential resp. bounded Pareto job sizes,  $\lambda = 0.48$  and varying values of  $d$ .

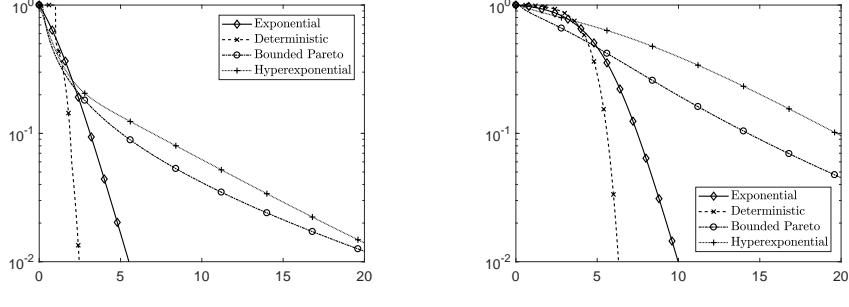
misleading to apply  $\text{Red}_{\text{iid}}(d)$  as a model when the replica sizes are in fact not independent.

### 3.9.1 $\text{Red}_{\text{iid}}(d)$ stochastically outperforms $\text{Red}_{\text{eq}}(d)$

Through a simple coupling argument we can show that the workload (and thus also the response time) for  $\text{Red}_{\text{iid}}(d)$  is always lower than for  $\text{Red}_{\text{eq}}(d)$ .

**Proposition 32.** *Suppose we have  $d \leq N < \infty$  servers, incoming job sizes follow a general distribution and the arrival process has a general distribution. Then the workload and response time distribution of the system of  $N$  servers operating under  $\text{Red}_{\text{iid}}(d)$  is always stochastically lower than for the same system operating under  $\text{Red}_{\text{eq}}(d)$ .*

*Proof.* Suppose the two systems are coupled such that arrivals occur at the same time and choose the same servers in both systems. At the first arrival, both systems are still empty and the new workload of a selected server for  $\text{Red}_{\text{iid}}(d)$  is given by  $\min_{i=1}^d X_i$  and  $X$  for  $\text{Red}_{\text{eq}}(d)$ , where  $X, X_1, \dots, X_d$  are independent and have distribution  $G$ . The inequality  $\min_{i=1}^d X_i \leq_d X$  obviously holds (where  $\leq_d$  denotes the distributional inequality). By induction, we assume that at an arbitrary arrival instant in the future, the chosen servers for  $\text{Red}_{\text{iid}}(d)$  have workload  $U_1, \dots, U_d$  while the chosen servers for



(a)  $\text{Red}_{\text{eq}}(d)$  for varying job sized and  $\lambda = 0.5$ .

(b)  $\text{Red}_{\text{eq}}(d)$  for varying job sized and  $\lambda = 0.8$ .

Figure 3.18: Logarithmic plot of the response time distribution for  $\text{Red}_{\text{eq}}(d)$  for all main job size distributions,  $\lambda = 0.5$  resp.  $\lambda = 0.8$  and  $d = 2$ .

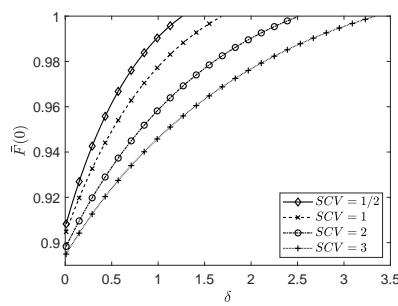
$\text{Red}_{\text{eq}}(d)$  have workload  $V_1, \dots, V_d$  with  $U_i \leq_d V_i$  for all  $i$ . We find (with  $X, X_1, \dots, X_d$  independent random variables with distribution  $G$ ):

$$\begin{aligned} \max \left\{ U_1, \min_{i=1}^d \{U_i + X_i\} \right\} &\leq_d \max \left\{ V_1, \min_{i=1}^d \{V_i + X_i\} \right\} \\ &\leq_d \max \left\{ V_1, \min_{i=1}^d \{V_i\} + X \right\} \end{aligned}$$

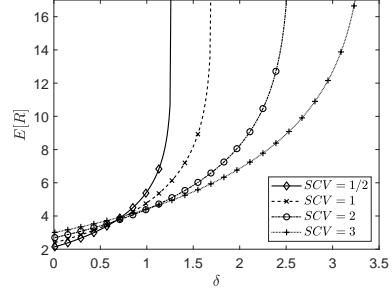
this shows (by permuting  $U_1, \dots, U_d$ ) that after an arrival instant the workload for  $\text{Red}_{\text{iid}}(d)$  is still stochastically smaller than that of  $\text{Red}_{\text{eq}}(d)$ . As the service is constant at rate 1 in both systems this shows that the inequality regarding the workload process indeed holds.

For the response times, we note that when a job as the one above arrives, its response time is exactly given by  $\min_{i=1}^d \{U_i + X_i\}$  resp.  $\min_{i=1}^d \{V_i\} + X$  for the  $\text{Red}_{\text{iid}}(d)$  resp.  $\text{Red}_{\text{eq}}(d)$  model. This shows by the above discussion that the distributional inequality for the response time distributions also holds.  $\square$

**Remark 32.** For deterministic job sizes equal to one, it is obvious that  $\text{Red}_{\text{eq}}(d)$  and  $\text{Red}_{\text{iid}}(d)$  become equivalent. We can also show this analytically by looking at Theorem 25 and Proposition 29, in Proposition 29 we find for deterministic job sizes that:  $\bar{F}_{R_1}(w) = 1$ ,  $w \leq 1$  and  $\bar{F}_{R_1}(w) = \bar{F}(w - 1)$  otherwise. This allows one to verify that (3.17) reduces to the same expression as (3.9) in case of deterministic job sizes.



(a) Plot of the system load as a function of the cancellation delay.



(b) Plot of the mean response time as a function of the cancellation delay.

Figure 3.19: Plots for  $\text{Red}_{\text{eq}}(d)$  with cancellation delay for  $X$  Erlang with 2 phases, mean one and  $\text{SCV} = 1/2$ , Exponential and Hyperexponential with balanced means and  $\text{SCV} = 2, 3$ ,  $d = 2$  and  $\lambda = 0.7$ .

### 3.9.2 Mean response time and workload distribution

In this section, we take another look at the setting in Section 3.8.1 for  $\text{Red}_{\text{iid}}(d)$  instead of  $\text{Red}_{\text{eq}}(d)$ . Figure 3.20 shows the workload as a function of the arrival rate  $\lambda$  and should be compared to Figure 3.11. We observe in Figure 3.20a that the workload equals  $\lambda$  for exponential job sizes (a fact which is shown in [39]). For bounded Pareto and hyperexponential job sizes, we also observe a close to linear growth as a function of  $\lambda$  with a less steep slope, implying that the stability region is larger for the more variable job size distributions. This is to be expected as the minimum of two more variable job size distributions has a smaller mean than the minimum of two exponential distributions. In Figure 3.20b we observe that, despite the fact that workload decreased when going from  $d = 1$  to  $d = 2$ , the workload increases when we further increase the value of  $d$  in case of bounded Pareto job sizes. This is further discussed in Section 3.9.4, where the stability is investigated.

In Figure 3.21, we observe the mean response time as a function of  $\lambda$  for the same settings as in Figure 3.20 (it should be compared to Figure 3.12). We observe some similarity between  $\text{Red}_{\text{eq}}(d)$  and  $\text{Red}_{\text{iid}}(d)$ : the mean response time is very low until  $\lambda$  gets very close to  $\lambda_{\max}$  at which point it snaps and goes to infinity. It is obvious that mean response times for  $\text{Red}_{\text{iid}}(d)$  lie far below the mean response times for  $\text{Red}_{\text{eq}}(d)$  and also the point at which

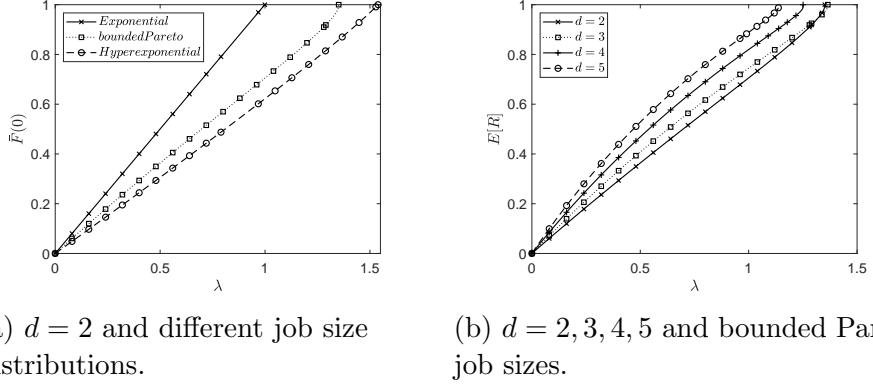


Figure 3.20: Workload  $\bar{F}(0)$  as a function of the arrival rate  $\lambda$  for  $\text{Red}_{\text{iid}}(d)$ . This Figure should be compared to Figure 3.11.

it snaps (i.e.  $\lambda_{\max}$ ) lies way further to the right.

### 3.9.3 Impact of job size variability

In this section, we take the same setting as in Section 3.8.2. We observe in Figure 3.22 that  $\text{Red}_{\text{iid}}(d)$  behaves completely different as a function of the SCV compared to  $\text{Red}_{\text{eq}}(d)$ . The mean response time decreases sharply as the SCV increases, moreover taking a higher value of  $d$  is always beneficial irrespective of the job size variability. In Figure 3.23, we observe that increasing the value of  $\lambda$  has no effect on the behaviour of  $\mathbb{E}[R]$  w.r.t. the SCV, it still decreases monotonically as the SCV increases. These Figures should be compared to Figures 3.13 and 3.14. These results further illustrate the inappropriateness of assuming independence for systems where the replicas do not have independent sizes.

Things are even more clear when we take the quotient of the mean response time for the  $\text{Red}_{\text{eq}}(d)$  policy and the  $\text{Red}_{\text{iid}}(d)$  policy for the same parameter settings. In Figure 3.24, we show the quotient of the data found in Figure 3.13 and Figure 3.22. We observe that increasing the job size variability and the number of chosen servers  $d$  both increase the mismatch between  $\text{Red}_{\text{eq}}(d)$  and  $\text{Red}_{\text{iid}}(d)$  dramatically. Furthermore Figure 3.25 depicts the quotient of the mean response times for  $\text{Red}_{\text{eq}}(d)$  resp.  $\text{Red}_{\text{iid}}(d)$  found in Figures 3.14 resp. Figure 3.23. We observe that the mismatch is even further increased by taking a higher value for  $\lambda$ .

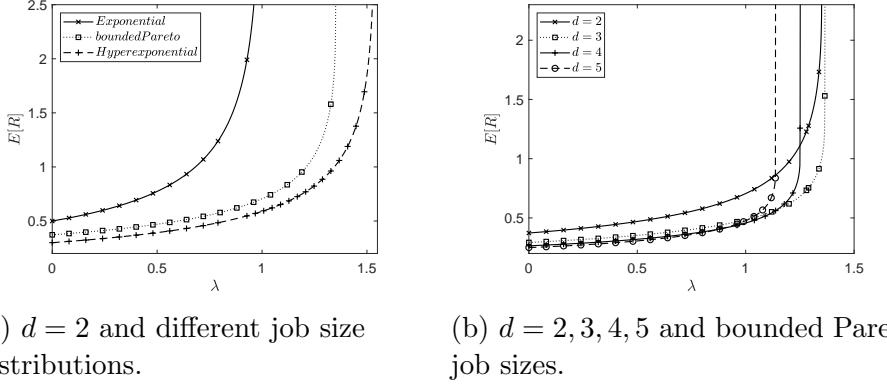
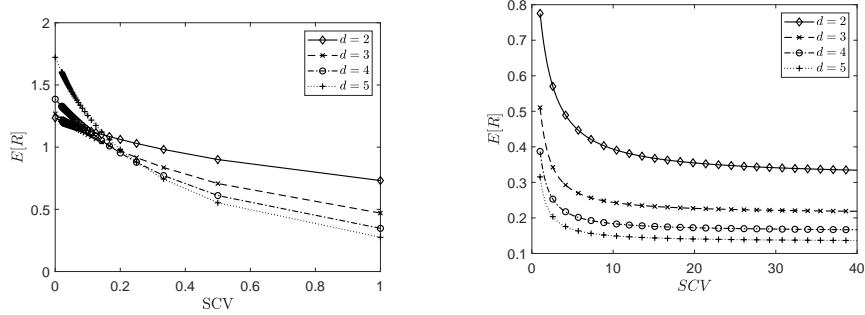


Figure 3.21: Mean response time  $E[R] = (1 + \int_0^\infty \bar{F}(w)^d dw)$  as a function of the arrival rate  $\lambda$  for  $\text{Red}_{\text{iid}}(d)$ . This Figure should be compared to Figure 3.12.

### 3.9.4 Stability

We now reuse the setting in Figure 3.15, the results are shown in Figure 3.26a. For deterministic job sizes the value of  $\lambda_{\max}$  is obviously identical for  $\text{Red}_{\text{iid}}(d)$  and  $\text{Red}_{\text{eq}}(d)$ . For other distributions, we observe a completely different picture, for hyperexponential job sizes:  $\lambda_{\max}$  increases quickly at first and then increases to a horizontal asymptote. For exponential job sizes the value of  $\lambda_{\max}$  is constant and equal to 1 (as shown in [39]). For bounded Pareto job sizes, we observe that first,  $\lambda_{\max}$  increases but afterwards it decreases to zero. It is in fact easy to show that for any job size distribution which has a lower bound, the value of  $\lambda_{\max}$  decreases to zero. Indeed, if the job sizes are lower bounded by a value  $a > 0$ , then we find for all  $w < a$  that  $\bar{F}_{R_1}(w) = 1$ , therefore by (3.17)  $\bar{F}'(w) = -\lambda d(1 - \bar{F}(w))$  for  $w < a$ . This ODE has the solution  $\bar{F}(w) = 1 - (1 - \bar{F}_0)e^{\lambda d w}$  (with  $\bar{F}(0) = \bar{F}_0$ ). As  $d \rightarrow \infty$  we see that for any  $\bar{F}_0 < 1$ ,  $\bar{F}(w)$  decreases to  $-\infty$  and is thus not a ccdf.

For hyperexponential job sizes, it seems like  $\lambda_{\max}$  converges to some constant around 1.8. We can indeed show that this is the case: the value of  $\lambda_{\max}$  for sufficiently large  $d$  is approximated by  $1/(\mathbb{E}[\min_{i=1}^d \{G_i\}] \cdot d)$ , where  $G_i$  are i.i.d. distributed as  $G$ . For  $d = N$  this approximation is exact for any job size distribution as the queue behaves like an M/G/1 queue with arrival rate  $\lambda N$  and processing time  $\min_{i=1}^d \{G_i\}$ . For exponential job sizes we have  $\mathbb{E}[\min_{i=1}^d \{G_i\}] \cdot d = 1$  for all  $d$ . For hyperexponential job sizes we



(a) Erlang job sizes with mean one and SCV on the  $x$ -axis.

(b) Hyperexponential job sizes with balanced means, mean one and SCV on the  $x$ -axis.

Figure 3.22: Mean response time  $\mathbb{E}[R] = (1 + \int_0^\infty \bar{F}(w)^d dw)$  as a function of the job sizes' SCV for  $d = 2, 3, 4, 5$  and  $\lambda = 0.45$  for the  $\text{Red}_{\text{iid}}(d)$  model. This Figure should be compared to Figure 3.13.

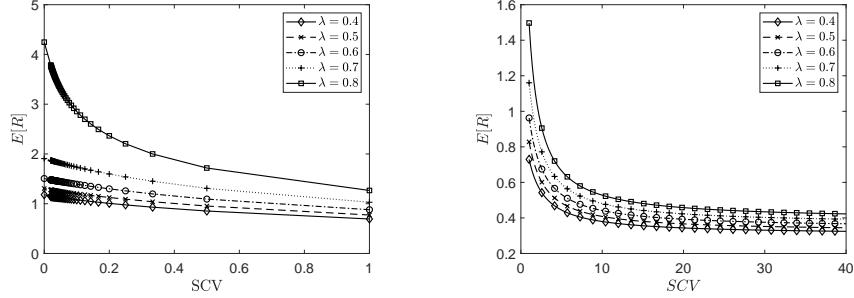
find  $\lim_{d \rightarrow \infty} \lambda_{\max} = 1.79 \dots$ . We illustrate this approximation as a function of  $d$  for bounded Pareto job sizes in Figure 3.26b.

### 3.9.5 Tail of the response time distribution

In Figure 3.27 we show the tail of the response time distribution when replicas are independent (for the same setting as when replicas were assumed to be identical in Figure 3.17). For independent replicas the discussion is much simpler, as both the exponential and bounded Pareto distribution are sufficiently variable, there is a clear advantage by making independent replicas. When considering distributions with a lighter tail, the results would be more similar to the case of identical replicas (see e.g. Figure 3.23a versus Figure 3.14a).

## 3.10 Future work

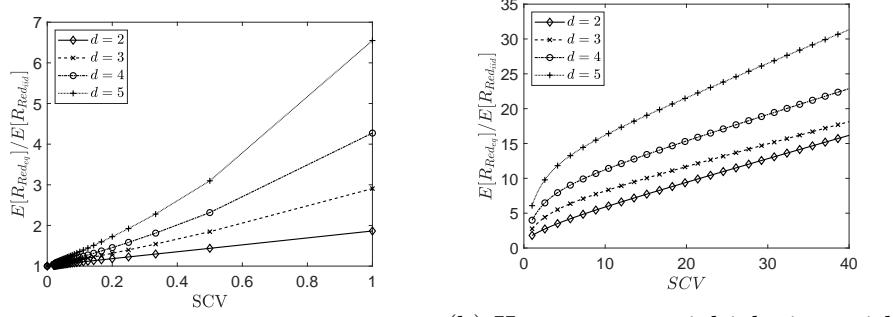
An important generalization is to look at the  $S\&X$  model introduced in [38]. The  $\text{Red}_{\text{eq}}(d)$  model corresponds to the  $S\&X$  model with no slowdown (i.e.,  $S = 1$ ), which implies that the replica that starts execution first also finishes first. As such it is always better to cancel the other replicas as soon as one starts execution. The  $\text{Red}_{\text{iid}}(d)$  model corresponds to the  $S\&X$  model



(a)  $d = 2$ ,  $\lambda = 0.4, 0.5, 0.6, 0.7, 0.8$   
and Erlang job sizes with mean one  
and SCV on the  $x$ -axis.  
(b)  $d = 2$ ,  $\lambda = 0.4, 0.5, 0.6, 0.7, 0.8$   
and hyperexponential job sizes with  
mean one and SCV on the  $x$ -axis.

Figure 3.23: Mean response time  $E[R] = (1 + \int_0^\infty \bar{F}(w)^d dw)$  as a function of the job sizes' SCV for the  $\text{Red}_{\text{iid}}(d)$  model. This Figure should be compared to Figure 3.14.

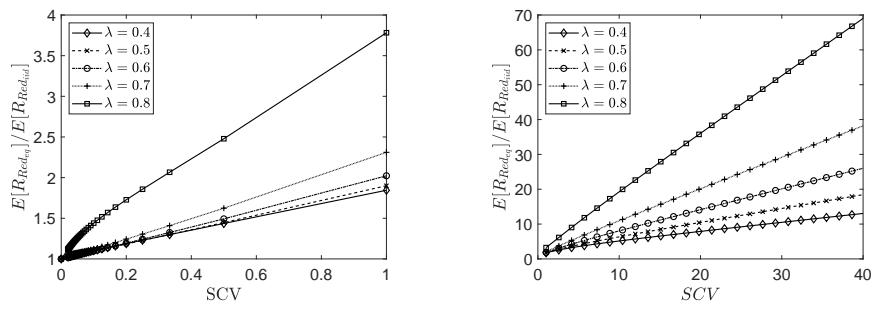
with deterministic job sizes (i.e.,  $X = 1$ ), which implies that if job sizes are more variable than exponential, extra replicas reduce latency. As such it is always better to replicate on as many servers as possible. However, with the  $S\&X$  model different replicas may experience different slowdowns and cancellation on start may no longer be superior. It is not hard to obtain general expressions for  $c_d(t, w, r)$  and  $C_d(t, r)$  for the  $S\&X$  model, which may lead to a similar differential equation with unknown boundary condition. In fact, in the next chapter we develop a general framework which also allows to analyse the  $\text{Red}(d)$  policy for the  $S\&X$  policy. Proving Conjecture 1 would give a theoretical basis for the analysis provided here (as was done for other load balancing schemes in [21]). We note that this is also an open problem for replication with i.d.d. replicas considered in [39]. It might be worth trying to explicitly solve the DIDE (3.9) for certain job size distributions.



(a) Erlang job sizes with mean one and SCV on the  $x$ -axis.

(b) Hyperexponential job sizes with balanced means, mean one and the SCV on the  $x$ -axis.

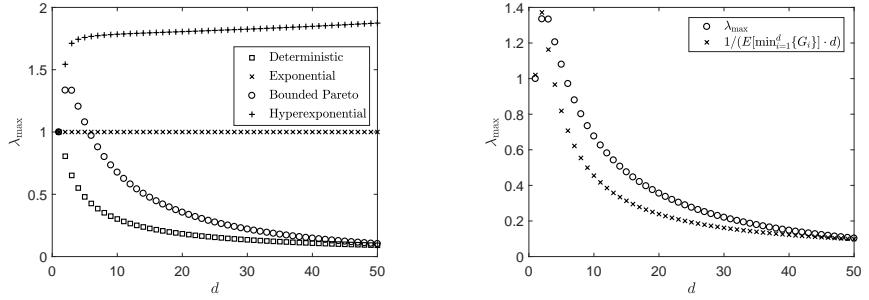
Figure 3.24: Quotient of the mean response time for identical replicas and independent replicas as a function of the job sizes' SCV for  $d = 2, 3, 4, 5$  and  $\lambda = 0.45$ .



(a)  $d = 2, \lambda = 0.4, 0.5, 0.6, 0.7, 0.8$  and Erlang job sizes with mean one and SCV on the  $x$ -axis.

(b)  $d = 2, \lambda = 0.4, 0.5, 0.6, 0.7, 0.8$  and hyperexponential job sizes with mean one and SCV on the  $x$ -axis.

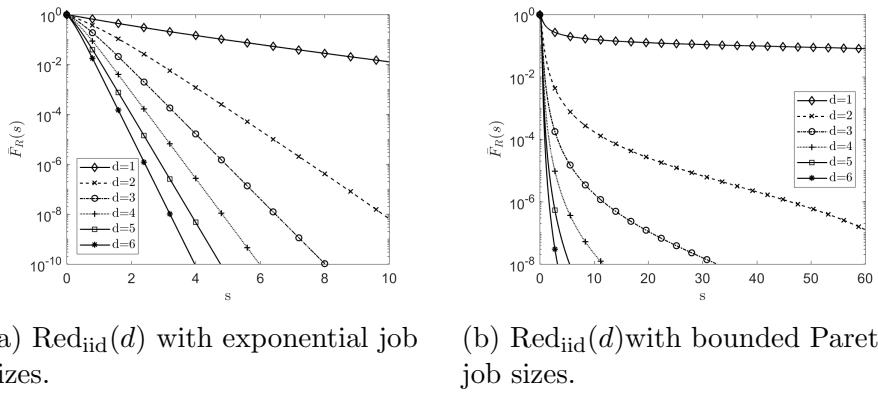
Figure 3.25: Quotient of the mean response time for identical replicas and independent replicas as a function of the job sizes' SCV for the  $Red_{iid}(d)$  model.



(a) Different job size distributions.  
This Figure should be compared to  
Figure 3.15

(b) Plot of how well  
 $1/(\mathbb{E}[\min_{i=1}^d G_i] \cdot d)$  approximates  
 $\lambda_{\max}$  for bounded Pareto job sizes.

Figure 3.26: Plot of  $\lambda_{\max}$  as a function of  $d$  for  $\text{Red}_{\text{iid}}(d)$ .



(a)  $\text{Red}_{\text{iid}}(d)$  with exponential job sizes.

(b)  $\text{Red}_{\text{iid}}(d)$  with bounded Pareto job sizes.

Figure 3.27: Logarithmic plot of the response time distribution for  $\text{Red}_{\text{iid}}(d)$  for exponential resp. bounded Pareto job sizes with  $\lambda = 0.48$  and varying values of  $d$ . These Figures shoul be compared to Figure 3.17.

# Chapter 4

## Performance analysis of workload dependent load balancing policies

What goes up must come down.  
Spinnin' wheel got to go 'round.

---

Blood, Sweat & Tears

### Abstract

Load balancing plays a crucial role in achieving low latency in large distributed systems. Recent load balancing strategies often rely on replication or use placeholders to further improve latency. However assessing the performance and stability of these strategies is challenging and is therefore often simulation based. In this chapter we introduce a unified approach to analyse the performance and stability of a broad class of workload dependent load balancing strategies. This class includes many replication policies, such as replicate below threshold, delayed replication and replicate only small jobs, as well as strategies for fork-join systems.

We consider systems with general job size distributions where jobs may experience server slowdown. We show that the equilibrium workload distribution of the cavity process satisfies an FDE and conjecture that the cavity process captures the limiting behavior of the system as its size tends to infinity.

We study this FDE in more detail for a variety of load balancing policies and propose a numerical method to solve it. The numerical method relies on a FPI or a simple Euler iteration depending on the type of FDE involved. We further show that additional simplifications can be made if certain distributions are assumed to be phase-type.

Various numerical examples are included that validate the numerical method and illustrate its strength and flexibility.

## 4.1 Introduction

Latency minimization is an important consideration in large scale data networks, server farms, cloud and grid computing, etc. A key role in achieving low latency is played by the load balancer responsible for distributing the jobs among the available servers. Popular load balancing schemes include the *join-shortest-queue among d randomly selected queues* (SQ( $d$ )) [67, 95, 20, 4] and the *join-idle-queue* (JIQ) [59, 84, 34] scheme. Under these schemes any incoming job is immediately assigned to a single server in the system.

A recent trend to further reduce latency is to use redundancy, that is, to assign an incoming job to multiple servers by distributing replicas of a job among the servers [7]. Initially this form of redundancy was introduced to combat unexpected server slowdowns, that is, a short job may suddenly experience an exceptionally long delay even if the server has low load. When redundancy is used, one can either cancel all remaining replicas as soon as one completes service [39] or as soon as a replica starts service (see [11] and chapter 2). The latter is useful to reduce the time that a job spends waiting in the queue, but is less effective when servers are subject to unexpected slowdowns. Fork-join based systems are another area where redundancy has been introduced to reduce latency [49, 81, 50]. In a fork-join system, a task is subdivided into sub-tasks which are executed on different servers and finally merged back as soon as the sub-tasks have been completed. Thus if one sub-task is delayed, so is the complete task. By introducing redundancy it suffices that only a subset of the sub-tasks complete.

To assess the performance of these load balancing schemes, most prior work relied on mean-field models, that is, studied the limiting behavior as the number of servers in the system becomes large under the assumption of asymptotic independence (an assumption that is very hard to prove for general service times, see [21]). In case of SQ( $d$ ) and JIQ, where jobs are

assigned immediately to a single server, the stability condition is simple and the system state is fully captured by the queue length at the different servers (plus the remaining job size in case of general job sizes). For systems with redundancy such a state description no longer works and even the system stability becomes complicated as replicas increase the actual workload and too much replication can easily lead to system instability [81].

Most prior analytical work on systems with redundancy focused on the redundancy- $d$  (Red( $d$ )) policy which replicates incoming jobs on  $d$  randomly selected servers, where the remaining replicas are either cancelled as the first replica starts or completes service. Product forms for the system state of LL( $d$ ) resp. Red( $d$ ) under the assumption of exponential job sizes resp. exponential job sizes and replicas that have independent sizes were presented in [11, 39]. Furthermore, in [10] a recent token based framework to analyse product forms and relevant performance measures for a variety of central queue based multi server models including LL( $d$ ) and Red( $d$ ) models was also introduced. A mean-field model for Red( $d$ ) with cancellation on completion was developed in [39] for independent replicas and in Chapter 3 for identical replicas. Red( $d$ ) with cancellation on start, which corresponds to assigning the job to the least loaded server, was analysed in Chapter 2. The stability issue of Red( $d$ ) with cancellation on completion was avoided in [37] by the RIQ policy, which replicates incoming jobs only on the idle servers among a set of  $d$  randomly selected servers (to mitigate the effect of server slowdown). This also simplified the performance analysis somewhat as existing results on vacation queues could be leveraged.

Another important contribution of [37] exists in introducing the S&X model. Under this model any replica has the same inherent job size  $X$ , but the actual service time of a replica on a server equals  $S$  times  $X$ , where  $S$  represents the slowdown that is assumed independent among replicas (as it depends on the server). This model is clearly closer to reality than assuming that all the replicas have independent job sizes (which is known to yield misleading insights such as more replication always reduces response times).

While Chapters 2 and 3 studied two different systems with redundancy, both develop a mean-field model that studies the evolution of the workload at a server. In this chapter we show that a very broad class of load balancing policies that rely on the workload information at a set of randomly selected servers can be analysed in a unified manner. More specifically, using the cavity process introduced in [20] we show that the workload distribution at a server is the solution of an FDE under the assumption of asymptotic

independence. We further study this FDE for a variety of load balancing policies belonging to this class under the S&X model. These include many load balancing schemes of practical interest for which no analytical method to assess their performance existed so far. Examples include policies that use delayed replication, replicate only on servers with a workload below some threshold, replicate only small jobs, replication in fork-join queues, etc.

In this chapter, we make the following contributions:

1. We define the cavity process for a broad class of workload dependent load balancing policies, characterise its transient evolution and show that its equilibrium environment is the solution of an FDE.
2. We show that many practical load balancing policies fit within our class of workload dependent policies and study their FDEs under the S&X model with general job size and slowdown distributions.
3. We propose different numerical methods to solve these FDEs, present numerical results for both the stability and response times and validate the accuracy of our approach using simulation.
4. We demonstrate that the numerical method can be further simplified if some of the distributions are phase-type.

With respect to the numerical method, we distinguish four different types of FDEs:

- *Type 1:* Future independent policies with unknown system load.
- *Type 2:* Future dependent policies with unknown system load.
- *Type 3:* Future independent policies with known system load.
- *Type 4:* Future dependent policies with known system load.

For each policy we obtain an FDE of the form  $\bar{F}'(w) = T(\bar{F}(w))$ ,  $w \in A_w$ . For the future independent policies we have  $A_w \subseteq [0, w]$  (Type 1, 3), which allows us to solve these policies using a simple forward Euler scheme. For the future dependent policies  $A_w \not\subseteq [0, w]$  (Type 2, 4), for these policies we rely on a FPI to obtain the equilibrium workload distribution. The second distinction is made on whether or not the system load,  $\bar{F}(0)$  is known (Type 3, 4) or unknown (Type 1, 2). When the load is unknown we leverage  $\bar{F}(\infty) = 0$  in order to obtain  $\bar{F}(0)$  using a binary search, otherwise we simply use the boundary condition on  $\bar{F}(0)$ .

## 4.2 Structure of this chapter

This chapter is organized as follows: In Section 4.3 we give a brief intuitive feel for the type of results presented in this chapter. In Section 4.4 we describe the model considered in this chapter in more detail. The terminology of the queue at the cavity is introduced in Section 4.5. This is followed by the analysis of the transient and equilibrium behaviour of the queue at the cavity in Section 4.6. We then apply our general result to many examples in Section 4.7. The equations for these examples are further studied when certain random variables are PH distributed in Section 4.8. In Section 4.9 we consider the considered policies in case there is no slowdown. In Section 4.10 we propose a numerical method to find the equilibrium distribution and the stability region from the results of Section 4.7 and 4.8. Results that validate our approach are given in the Section 4.11, future work is suggested in Section 4.12.

## 4.3 Informal intuition

In this chapter, we formalize the idea of the level crossing argument which we have previously illustrated for the LL( $d$ ) and Red( $d$ ) policy. In particular, for any workload dependent load balancing policy, we balance the load solely based on the workload information. Therefore, the only state descriptor we require for the queue at the cavity is its workload. We may therefore denote the equilibrium workload distribution of the cavity queue by a random variable  $U \in [0, \infty)$ .

We note that, when using a power of  $d$  choices algorithm, the queue at the cavity is selected for a potential queue with rate  $\lambda \cdot d$ . At an arrival instant, the load at the cavity queue is distributed as  $U$ , and we denote by  $\mathcal{Q}(U)$  the random variable which represents the load at the cavity queue right after the potential arrival. The *level crossing argument* at workload  $w$  now formalizes as stating that:

$$-\bar{F}'(w) = f(w) = \lambda d \mathbb{P}\{U \leq w, \mathcal{Q}(U) > w\}. \quad (4.1)$$

Indeed, the left hand side denotes the downcrossing rate (as the cavity queue finishes work at a constant rate equal to one), while the right hand side denotes the up-crossing rate over level  $w$ . After we prove (4.1), this allows one to more easily analyse workload dependent load balancing policies.

**Example 33.** For the  $LL(d)$  policy, we denote by  $U_2, \dots, U_d$  a set of i.i.d. copies of  $U$  and  $X$  a random job. We have  $\mathcal{Q}(U) = U + X$  if  $U < U_2, \dots, U_d$  while  $\mathcal{Q}(U) = U + X$  with probability  $\frac{1}{j+1}$  if  $U = 0 = U_i$  for exactly  $j$  values of  $i \in \{2, \dots, d\}$ . We can therefore compute  $\mathbb{P}\{U \leq w, \mathcal{Q}(U) > w\}$  as:

$$F(0) \cdot \sum_{j=0}^{d-1} \binom{d-1}{j} \frac{1}{j+1} F(0)^j \bar{F}(0)^{d-j-1} \bar{G}(w) + \int_0^\infty f(u) \bar{F}(u)^{d-1} \bar{G}(w-u) du.$$

Taking a second look at Section 2.3, one can see that these are the same equations as the ones presented there. Indeed, in Section 2.3, we already used the level crossing argument for the  $LL(d)$  policy!

**Example 34.** For the  $Red(d)$  policy, we again denote by  $U$  the workload of the cavity queue,  $U_2, \dots, U_d$  i.i.d. copies of  $U$  and  $X$  a job size. One finds that  $\mathcal{Q}(U) = \max\{U, \min_{i=1}^d \{U_i\} + X\}$  (with  $U_1 = U$ ). One can again see how this formalizes the analysis carried out in Section 3.3.

In this chapter, we first show that (4.1) can indeed be used to analyse workload dependent load balancing policies by first considering the transient regime and letting  $t \rightarrow \infty$ . Having derived (4.1), we apply this approach to many (more advanced) load balancing policies in a setting where job sizes have some inherent size  $X$  and all selected servers may experience some i.i.d. slowdown  $S_i$ .

## 4.4 Model description

We consider a generic power-of- $d$  system consisting of  $N$  identical, infinite buffer servers which serve jobs in a FCFS manner (here  $N$  is usually assumed to be large). Arrivals occur according to a Poisson( $\lambda N$ ) process and the service rate at each server equals one. Whenever a job arrives,  $d$  distinct servers are chosen uniformly at random (with or without replacement). The job then creates some (or possibly no) added work on each of the  $d$  chosen servers depending on the load balancing policy used. The policy is chosen such that the added work (i.e. the actual arrival size) solely depends on the workload at each of the chosen servers (and other variables, independent of the chosen servers). For the load balancing policies considered in this chapter, this added work consists of either the arriving job, partial execution of the job or other overheads due to placeholders as in the  $LL(d, k, \delta)$  policy

studied in Section 4.7.5. We shall henceforth refer to this type of model as a workload dependent load balancing policy. Note that for this model with finite  $N$ , the process which only keeps track of the workload at each server, is a Markov process.

## 4.5 Cavity process

We employ the cavity process methodology introduced in [20] to formulate a general method to obtain the transient and equilibrium workload distribution for a workload dependent load balancing policy in the mean-field regime. We first provide some intuition as to why the study of a queue at the cavity might be of interest. Looking at the many server system, instead of attempting to capture the global evolution of all  $N$  workload distributions, we single out one queue which we henceforth refer to as the queue at the cavity. It is not hard to see that, as arrivals occur at rate  $\lambda N$  and each arrival selects  $d$  queues, the queue at the cavity is selected with a rate equal to  $\lambda d$ . Every time it is selected, we have to add some (or possibly no) work to it where the amount of work depends on the workload of the  $d$  selected queues. As we are not keeping track of the workload at any of the  $d - 1$  other selected queues, we simply generate their workload as a random variable which is independent of but identically distributed as the workload of the queue at the cavity at the time of the arrival. This method is known to yield exact results as  $N \rightarrow \infty$  for some policies (those for which Conjecture 1 holds) and can often be used as a good approximation for sufficiently high values of  $N$  (see Section 4.11).

In the cavity process method, potential arrivals occur according to a  $\text{Poisson}(\lambda d)$  process. Whenever a potential arrival occurs, we create  $d - 1$  random variables with the same distribution as the queue at the cavity, add the actual arrival size to the queue at the cavity and discard these  $d - 1$  random variables again. Concretely: let  $U_1, \dots, U_d$  denote the (i.i.d.) workloads at the  $d$  chosen servers just before the potential arrival, where w.l.o.g.  $U_1$  represents the queue at the cavity. Suppose we are given some additional random variables  $V_1, \dots, V_r$  (e.g., job size or server slowdown variables) that influence the added work. Then, we denote by  $\mathcal{Q}(U_1, \dots, U_d, V_1, \dots, V_r)$  the random variable which represents the new workload in the queue at the cavity  $U_1$ . We call a potential arrival to  $U_1$  an actual arrival if and only if  $\mathcal{Q}(U_1, \dots, U_d, V_1, \dots, V_r) > U_1$ . Note that while potential arrivals occur ac-

cording to a Poisson( $\lambda d$ ) process, the rate of actual arrivals strongly depends on the chosen policy and is generally hard to compute. Furthermore, depending on the load balancing policy, the actual arrival comprises of jobs that are either served completely at this server, jobs that are partially executed at the server or even other overhead like fetching a job which is no longer available. To illustrate what  $\mathcal{Q}$  signifies, we present a few simple examples for policies which have been studied before.

**Example 35.** Consider the  $LL(d)$  policy studied in Chapter 2, where an incoming job of a certain size joins the least loaded server among  $d$  selected servers. In this case  $r = 1$ ,  $V_1 = X$  is the job size and  $\mathcal{Q}(U_1, \dots, U_d, X)$  is equal to  $U_1 + X$  if  $U_1 < \min_{i=2}^d U_i$  and it is equal to  $X$  with probability  $\frac{1}{m}$  if  $U_j = 0$  for exactly  $m$  choices of  $j$  including  $j = 1$ . Otherwise  $\mathcal{Q}(U_1, \dots, U_d, X) = U_1$ .

**Example 36.** Two other examples are  $Red(d)$  with independent resp. identical replicas as studied in chapter 3, where an incoming job replicates itself onto  $d$  servers and experiences an independent resp. identical service time on each server. The job is then cancelled as soon as one of the replicas finishes. For the case of independent replicas, we have  $r = d$  and  $V_i = X_i$  where  $X_i, i = 1, \dots, d$  are the i.i.d. job size variables. In this case, we have  $\mathcal{Q}(U_1, \dots, U_d, X_1, \dots, X_d) = \max\{U_1, \min_{i=1}^d \{U_i + X_i\}\}$ , indeed, a replica of the job finishes service by time  $\min_{i=1}^d \{U_i + X_i\}$ . For the case when the replicas are identical, we have  $r = 1$  and  $V_1 = X$  where  $X$  is the job size. A replica finishes service by time  $\min_{i=1}^d \{U_i\} + X$ , which yields  $\mathcal{Q}(U_1, \dots, U_d, X) = \max\{U_1, \min_{i=1}^d \{U_i\} + X\}$ .

**Definition 37** (Cavity Process). Let  $\mathcal{H}(t)$ ,  $t \geq 0$ , be a set of probability measures on  $[0, \infty)$  called the environment process. The cavity process  $U^{\mathcal{H}(\cdot)}(t)$ ,  $t \geq 0$ , takes values in  $[0, \infty)$  and is defined as follows. Potential arrivals occur according to a Poisson process with rate  $\lambda d$ . When a potential arrival occurs at time  $t$ , the cavity process  $U^{\mathcal{H}(\cdot)}(t)$  becomes  $\mathcal{Q}(U^{\mathcal{H}(\cdot)}(t-), U_2, \dots, U_d, V_1, \dots, V_r)$ . Here  $U_2, \dots, U_d$  are  $d - 1$  independent random variables with law  $\mathcal{H}(t-)$ , and  $V_1, \dots, V_r$  are random variables which are independent of the process  $U^{\mathcal{H}(\cdot)}(\cdot)$ . The cavity process decreases at rate one during periods without arrivals and is lower bounded by zero.

We now define the cavity process associated to the equilibrium environment process, which is such that the cavity process itself has distribution  $\mathcal{H}(t)$  at time  $t$ :

**Definition 38** (Equilibrium Environment). *When a cavity process  $U^{\mathcal{H}(\cdot)}(\cdot)$  has distribution  $\mathcal{H}(t)$  for all  $t \geq 0$ , we say that  $\mathcal{H}(\cdot)$  is an equilibrium environment process. Further, a probability measure  $\mathcal{H}$  is called an equilibrium environment if  $\mathcal{H}(t) = \mathcal{H}$  for all  $t$  and  $U^{\mathcal{H}(\cdot)}(t)$  has distribution  $\mathcal{H}$  for all  $t$ .*

A modularized program for analyzing load balancing systems by using the cavity process method was presented in [20]. In this program, one essentially needs to show asymptotic independence, which allows to assume that the workloads at the different queues become independent random variables and justifies that the behaviour of the entire system can be described by the behaviour of the queue at the cavity. One then needs to find a defining equation for the equilibrium behaviour of the queue at the cavity. This equation is given by (4.4) for our model. We use this equation to study several workload dependent load balancing policies. As will become apparent further on, all workload dependent load balancing policies which have been studied in the mean-field regime thus far can be analysed using this approach.

The asymptotic independence between the different queues is something which is very difficult to prove in general. For the models considered in this chapter, known proof techniques only exist for:

- The LL( $d$ ) policy (see [22]).
- Any convex combination of LL( $d_j, K_j$ ) (with arbitrary job sizes) and Red( $d_j, K_j$ ) (with exponential job sizes and i.i.d. replicas) (see [83]).

We believe [83] presents the most general result and its proof allows to be generalized to other workload dependent load balancing policies. However, something all these proofs have in common is that any server is busy with probability  $\lambda$  times the mean job size. That is, no redundant work is executed. Moreover, for all these policies, more redundancy is always better, that is increasing the value of  $d$  always yields lower response times. We therefore do not think the proofs found in these papers allow to be generalized to a policy such as Red( $d$ ) with identical replicas.

However, we believe that for all policies considered in this chapter, the queues in the limiting regime satisfy this asymptotic independence property and then proceed with applying the modularized program. The remarkable accuracy between the performance measures obtained using our method and those obtained via simulation (see Section 4.11) supports our belief that Conjecture 1 indeed holds.

**Remark 33.** *The results in this chapter characterize the queue at the cavity associated to workload dependent policies. In case Conjecture 1 fails to hold for a policy, one can still analyse the associated queue at the cavity regardless and this may be used as an (accurate) approximation for the actual model.*

## 4.6 Mean-field analysis

### 4.6.1 Transient behaviour

We start with the transient behavior. Note that at each time  $t$ , the pdf of the workload of the queue at cavity, i.e.,  $f(t, \cdot)$  integrates to  $\int_0^\infty f(t, u) du = \bar{F}(t, 0)$ . As typically,  $\bar{F}(t, 0) < 1$  we have a point mass at zero which is equal to  $F(t, 0)$ . For the transient behaviour, we obtain the following PDIDE:

**Theorem 39.** *The workload of the queue at the cavity satisfies the following PDIDE:*

$$\begin{aligned} \frac{\partial f(t, w)}{\partial t} - \frac{\partial f(t, w)}{\partial w} = & -\lambda d \left[ f(t, w) \mathbb{P}\{\mathcal{Q}(U^{\mathcal{H}(\cdot)}(t-)) > w \mid U^{\mathcal{H}(\cdot)}(t-) = w\} \right. \\ & - F(t, 0) \mathbb{P}\{\mathcal{Q}(U^{\mathcal{H}(\cdot)}(t-)) = w \mid U^{\mathcal{H}(\cdot)}(t-) = 0\} \\ & \left. - \int_0^w f(t, u) \mathbb{P}\{\mathcal{Q}(U^{\mathcal{H}(\cdot)}(t-)) = w \mid U^{\mathcal{H}(\cdot)}(t-) = u\} du \right] \end{aligned} \quad (4.2)$$

$$\frac{\partial F(t, 0)}{\partial t} = -\lambda d \left[ F(t, 0) \mathbb{P}\{\mathcal{Q}(U^{\mathcal{H}(\cdot)}(t-)) = w \mid U^{\mathcal{H}(\cdot)}(t-) = 0\} + f(t, 0^+) \right], \quad (4.3)$$

for  $w > 0$ , where  $f(t, w^+) = \lim_{v \downarrow w} f(t, v)$ .

*Proof.* Let  $\Delta > 0$  be arbitrary, and consider the events where the workload of the queue at cavity becomes  $w$  at time  $t + \Delta$  by looking at its value at time  $t$  and the behaviour in  $[t, t + \Delta]$ . As potential arrivals occur according to a Poisson process, all events which involve more than one arrival in  $[t, t + \Delta]$  are  $o(\Delta)$ . We do distinguish the following three events which do not involve more than one arrival on  $[t, t + \Delta]$ :

- The queue at the cavity has workload  $w + \Delta$  at time  $t$  and its workload is not increased by arrivals in  $[t, t + \Delta]$ , this occurs with density:

$$Q_1 = \left( 1 - \lambda d \int_0^\Delta \mathbb{P} \left\{ \begin{array}{l} \mathcal{Q}(U^{\mathcal{H}(\cdot)}((t+v)-) > w + \Delta - v, \\ U^{\mathcal{H}(\cdot)}((t+v)-) = w + \Delta - v \end{array} \right\} dv \right).$$

- The queue at the cavity has workload 0 at time  $t$  and its workload is increased to  $w + (\Delta - v)$  at time  $t + v$ . This event has density:

$$Q_2 = \lambda d \int_0^\Delta \mathbb{P} \left\{ \mathcal{Q}(U^{\mathcal{H}(\cdot)}((t+v)-) = w + (\Delta - v), U^{\mathcal{H}(\cdot)}((t+v)-) = 0) \right\} dv.$$

- The queue at the cavity's workload lies in the interval  $(v, w + \Delta - v)$  at time  $t + v$  and its workload is increased to  $w + \Delta - v$  by a potential arrival, we find:

$$\begin{aligned} Q_3 &= \lambda d \int_0^\Delta \int_v^{w+\Delta-v} \mathbb{P} \left\{ \begin{array}{l} \mathcal{Q}(U^{\mathcal{H}(\cdot)}((t+v)-) \\ = w + (\Delta - v), U^{\mathcal{H}(\cdot)}((t+v)-) = u \end{array} \right\} du dv. \end{aligned}$$

We therefore find that:

$$f(t + \Delta, w) = Q_1 + Q_2 + Q_3 + o(\Delta),$$

by subtracting  $f(t, w)$  on both sides, dividing by  $\Delta$  and taking the limit  $\Delta \rightarrow 0$  we find that the claimed equality (4.2) indeed holds. We now investigate the boundary condition which characterizes the events associated with the evolution of the workload on  $[t, t + \Delta]$  such that it is zero at time  $t + \Delta$ . The workload can either be zero at time  $t$  and no potential arrivals occur in  $[t, t + \Delta]$  or the workload is in  $(0, \Delta)$  at time  $t$  and no potential arrival in the interval  $[t, t + \Delta]$  increases its workload. We find:

$$\begin{aligned} F(t + \Delta, 0) &= F(t, 0) \left( 1 - \lambda d \cdot \right. \\ &\quad \left. \int_0^\Delta \mathbb{P} \left\{ \mathcal{Q}(U^{\mathcal{H}(\cdot)}((t+v)-) > 0 \mid U^{\mathcal{H}(\cdot)}(t) = 0 \right\} dv \right) + \int_0^\Delta f(t, u) \left( 1 - \lambda d \cdot \right. \\ &\quad \left. \int_0^\Delta \mathbb{P} \left\{ \mathcal{Q}(U^{\mathcal{H}(\cdot)}((t+v)-)) > \max\{0, (u-v)\} \mid U^{\mathcal{H}(\cdot)}(t) = u \right\} dv \right) du + o(\Delta). \end{aligned}$$

By subtracting  $F(t, 0)$  on both sides, dividing by  $\Delta$  and taking the limit  $\Delta \rightarrow 0$  we find that (4.3) indeed holds.  $\square$

### 4.6.2 Equilibrium environment

To compute the equilibrium distribution we need to take the limit  $t \rightarrow \infty$ , thereby leaving out the dependence on  $t$ . In particular, we have  $\frac{\partial f(w)}{\partial t} = 0$ . We now directly derive an FDE for the workload distribution from Theorem 39.

**Theorem 40.** *The equilibrium workload distribution of the queue at the cavity satisfies the following FDE:*

$$\bar{F}'(w) = -\lambda d \mathbb{P}\{U^{\mathcal{H}} \leq w, \mathcal{Q}(U^{\mathcal{H}}) > w\}. \quad (4.4)$$

*Proof.* For convenience we write  $U$  for  $U^{\mathcal{H}(\cdot)}$ . From (4.2) we readily obtain the following by integrating w.r.t.  $w$  once:

$$f(0) - f(w) = -\lambda d \int_0^w \left[ \mathbb{P}\{\mathcal{Q}(U) > u, U = u\} - \mathbb{P}\{\mathcal{Q}(U) = u, U = 0\} - \int_0^u \mathbb{P}\{\mathcal{Q}(U) = u, U = v\} dv \right] du. \quad (4.5)$$

The equality in (4.3) reduces to the boundary condition:

$$f(0) = \lambda d \mathbb{P}\{\mathcal{Q}(0) > 0\} F(0),$$

using the fact that  $\bar{F}'(w) = -f(w)$  we obtain from (4.5):

$$\begin{aligned} \bar{F}'(w) &= -\lambda d \left[ F(0) \mathbb{P}\{\mathcal{Q}(0) > 0\} + \int_0^w \mathbb{P}\{\mathcal{Q}(U) > u, U = u\} du \right] \\ &\quad + \lambda d \int_0^w \mathbb{P}\{\mathcal{Q}(U) = u, U = 0\} du \\ &\quad + \lambda d \int_0^w \int_0^u \mathbb{P}\{\mathcal{Q}(U) = u, U = v\} dv du. \end{aligned} \quad (4.6)$$

Note that the first line in (4.6) is the rate of all possible upward jumps of  $U$  after a potential arrival when the workload in the cavity queue just before the potential arrival satisfies  $U \in [0, w]$ . The first term in the second line in

(4.6) is the rate at which  $U$  jumps to somewhere below  $w$  when  $U = 0$  at the time of a potential arrival. The last term in the second line in (4.6) is the rate at which  $U$  jumps up to somewhere below  $w$  while  $U \in (0, w]$ . The last two events are subsets of the first event and it can be observed that the difference of these events is the rate at which the cavity process jumps to a workload larger than  $w$  for  $U \in [0, w]$ . This yields equality (4.4).  $\square$

**Remark 34.** *The left hand side of (4.4) is  $\bar{F}'(w) = -f(w)$ , where  $f(w)$  is the down-crossing rate through  $w$  while the right hand side is minus the up-crossing rate through  $w$ .*

## 4.7 Load balancing policies

While our main result (Theorem 40) is applicable for any workload dependent load balancing policy as described in Section 4.4, in this section we specialize this result for some practical workload dependent policies. In many classic load balancing methods (like e.g. LL( $d$ ) and SQ( $d$ )), a job is only sent to one server and its processing time solely depends on the speed of that server. There are however many load balancing policies, in particular those which employ some type of redundancy, which use the processing power of multiple servers in order to complete service. In this case, the question arises as to how one should treat the processing time at the different servers. Two popular choices are to assume that the processing time at the chosen servers are independent (see e.g. [39]) or that the processing times are identical (see e.g. Chapter 3). Recently the S&X model was introduced in [37], this model is a combination of identical and independent replicas, each job has a size  $X$  which is identical over all chosen servers and a slowdown  $S$  which is independent over the chosen servers. In this section we analyse all considered policies in a setting which is a generalization of the S&X model which we explain shortly. In this section, we also present various numerical results for these policies to outline some important features. Simulation experiments that validate our approach can be found in Section 4.11.

Each job has an inherent size  $X > 0$  and on each of the servers a job replica experiences some arbitrary slowdown denoted by the variable  $S_i$ . Thus each arrival is defined by a random job size variable  $X$  and  $d$  i.i.d. slowdown random variables  $S_1, \dots, S_d$ . Using the notation of Section 4.6.1 we set  $r = d + 1$ ,  $V_i = S_i$  and  $V_{d+1} = X$ . While the actual processing time of the  $i$ -th replica in the S&X model of [37] then equals  $S_i X$ , we

consider a more general setting. We assume that there exists some function  $g : [0, \infty) \times (0, \infty) \rightarrow (0, \infty)$  which is non-decreasing in both components such that if an arrival occurs, it has size  $g(S_i, X)$  on the  $i^{th}$  chosen server. For any  $s, x > 0$ , define  $g_x(s) = g(s, x)$  and assume it is a strictly increasing, continuous function. Note that in particular its inverse exists and we assume the inverse is differentiable. In our numerical experiments we set  $g(S, X) = X + SX$ , where  $S$  and  $X$  are generally distributed random variables with  $X$  the inherent job size and  $S$  the slowdown variable, such that the processing time cannot be less than  $X$  irrespective of the slowdown.

**Example 41.** Consider the Red( $d$ ) policy where, at each arrival, the job is replicated on  $d$  servers. Suppose the workload resp. the slowdown at each of the  $d$  servers is given by  $U_1, \dots, U_d$  resp.  $S_1, \dots, S_d$  and the job size is  $X$ . In this case we find that the workload  $U_1$  is increased to:

$$\mathcal{Q}(U_1) = \max\{U_1, \min_{i=1}^d \{U_i + g(S_i, X)\}\}.$$

Moreover, the response time is given by:

$$R = \min_{i=1}^d \{U_i + g(S_i, X)\}.$$

Before proceeding with the analysis of the different load balancing policies, we outline some more notations used throughout this chapter. For any sequence of random variables  $Y_1, \dots, Y_n$ , let  $Y_{(k)}$  denote its  $k$ 'th order statistic such that  $Y_{(1)} \leq Y_{(2)} \leq \dots \leq Y_{(n)}$ , and ties are broken at random. In the  $S \& X$  setting, we define  $R_x = U + g_x(S)$  as the sojourn time of a job of size  $x$  if it is sent to a single server with workload  $U$  and slowdown  $S$ . In this case, its ccdf is given by:

$$\begin{aligned} \bar{F}_{R_x}(w) &= \bar{F}_{g_x(S)}(w) + \int_0^w \bar{F}(u) f_{g_x(S)}(w-u) du \\ &= \bar{F}_S(g_x^{-1}(w)) + \sum_{u \leq w} \bar{F}(w-u) \mathbb{P}\{S_d = g_x^{-1}(u)\} + \\ &\quad \int_0^w \bar{F}(w-u) f_{S_c}(g_x^{-1}(u)) \cdot (g_x^{-1})'(u) I_{g_x([0,\infty))}(u) du, \end{aligned} \quad (4.7)$$

where the second equality follows from the fact that the pdf of  $g_x(S_c)$  is given by  $f_{g_x(S_c)}(w) = f_{S_c}(g_x^{-1}(w)) \cdot (g_x^{-1})'(w) I_{g_x([0,\infty))}(w)$  (here  $I_A(u)$  equals one if

$u \in A$  and zero otherwise). Moreover we denote by  $\tilde{X} = g(S, X)$  the job size distribution at a single server. Analogously to (4.7), we find for  $R_{\tilde{X}} = U + \tilde{X}$ :

$$\bar{F}_{R_{\tilde{X}}}(w) = \bar{F}_{\tilde{X}}(w) + \int_0^w \bar{F}(u) f_{\tilde{X}}(w-u) du, \quad (4.8)$$

where the integral can again be split into a discrete and continuous part.

#### 4.7.1 Type 1 : Red( $d, k, \delta$ )

In this section, we analyse the redundancy based policy Red( $d, k, \delta$ ). Under this policy, an arriving job of size  $k \cdot X$  selects  $d$  servers uniformly at random and places an identical replica of size  $X$  at each of the  $d$  servers. When any  $k$  of the  $d$  replicas have received service, the other redundant replicas are cancelled. Additionally we assume that the cancellation of redundant replicas requires a constant time  $\delta \geq 0$ . In other words, this means that once the  $k$ 'th replica has been completed, the other servers continue working on remaining replicas (if they happen to be in service at that server) for a time  $\delta$ . At the end of this section we indicate how this policy is used in practice.

We now show that the FDE in Theorem 40 reduces to a DIDE without a boundary condition, meaning it is a Type 1 policy.

**Proposition 42.** *For the Red( $d, k, \delta$ ) policy, the FDE in equation (4.4) reduces to the following DIDE (recall  $\bar{F}_{R_x}$  and  $\bar{F}_{R_{\tilde{X}}}$  from (4.7-4.8)):*

$$\bar{F}'(w) = -\lambda d(\bar{F}_{R_{\tilde{X}}}(w) - \bar{F}(w)) \quad \text{if } w \leq \delta \\ (4.9)$$

$$\begin{aligned} \bar{F}'(w) = -\lambda d & \left( \int_0^\infty \sum_{j=0}^{k-1} \binom{d-1}{j} F_{R_x}(w-\delta)^j \bar{F}_{R_x}(w-\delta)^{d-j-1} \right. \\ & \left. (\bar{F}_{R_x}(w) - \bar{F}(w)) f_X(x) dx \right) \quad \text{otherwise.} \end{aligned} \quad (4.10)$$

*Proof.* Assume that the selected servers have workloads  $U_1, \dots, U_d$  and an arriving job of size  $X$  experiences slowdown  $S_1, \dots, S_d$  on the selected servers. If  $w \leq \delta$ , it is obvious that  $\mathcal{Q}(U_1) > w$  if and only if its response time at

the queue at the cavity  $R_{\tilde{X}} \stackrel{d}{=} U_1 + g(S_1, X) > w$ . We therefore find from Theorem 40 for  $w \leq \delta$ :

$$\bar{F}'(w) = -\lambda d \mathbb{P}\{U_1 + g(S_1, X) > w, U_1 \leq w\},$$

which leads to (4.9). Now, assume  $w > \delta$ . For  $i = 1, \dots, d$  we define  $Y_i = U_i + g(S_i, X)$  as the *effective workload* after addition of a copy. We have

$$\mathcal{Q}(U_1) = \max \{U_1, \min\{Y_1, Y_{(k)} + \delta\}\}. \quad (4.11)$$

In words, this means that the new workload at the cavity queue can take one of the values below due to the potential arrival.

- $\mathcal{Q}(U_1) = U_1$ . This happens when  $U_1 > Y_{(k)} + \delta$ .
- $\mathcal{Q}(U_1) = Y_1$ . This happens when  $Y_1 < Y_{(k)} + \delta$ .
- $\mathcal{Q}(U_1) = Y_{(k)} + \delta$ . This happens when  $Y_1 \geq Y_{(k)} + \delta$  and  $U_1 \leq Y_{(k)} + \delta$ .

Let us consider the term  $\mathbb{P}\{U_1 \leq w, \mathcal{Q}(U_1) > w\}$ . Note that the event  $\{U_1 \leq w, \mathcal{Q}(U_1) > w\}$  implies that  $Y_1 > w$ . Further,  $Y_1 > w$  implies that  $\mathcal{Q}(U_1) \neq U_1$  and hence from (4.11) we have:

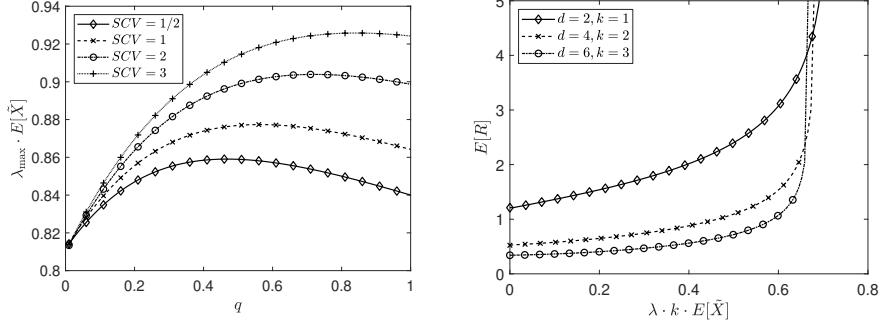
$$\begin{aligned} \mathbb{P}\{U_1 \leq w, \mathcal{Q}(U_1) > w\} &= \mathbb{P}\{U_1 \leq w, \min\{Y_1, Y_{(k)} + \delta\} > w\} \\ &= \mathbb{P}\{U_1 \leq w, Y_1 > w, Y_{(k)} + \delta > w\}. \end{aligned}$$

Now  $Y_{(k)} + \delta > w$  only if at most  $k - 1$  of the sampled queues have an *effective workload* which is bounded above by  $w - \delta$ . Given that  $Y_1 > w$  for the queue at cavity, we have that its effective workload is not bounded by  $w - \delta$ . By conditioning on the random variable  $X$ , we have from Theorem 40, the ansatz property (that the queues have independent workloads) and the discussion above that (4.10) indeed holds.  $\square$

**Remark 35.** In the special case  $d = k$ , this policy reduces to the classic fork-join policy and one finds that  $\sum_{j=0}^{d-1} \binom{d-1}{j} F_{R_x}(w - \delta)^j \bar{F}_{R_x}(w - \delta)^{d-j-1} = 1$ . Therefore we simply have  $\bar{F}'(w) = -\lambda d(\bar{F}_{R_{\tilde{X}}}(w) - \bar{F}(w))$ .

**Remark 36.** Taking  $\delta = 0, k = 1, X \stackrel{d}{=} 1$  and  $g(S, X) = SX = S$ , we find that  $\bar{F}$  satisfies:

$$\begin{aligned} \bar{F}'(w) &= -\lambda d(\bar{F}_{R_1}(w) - \bar{F}(w)) \bar{F}_{R_1}(w)^{d-1} \\ \bar{F}_{R_1} &= \bar{F}_S(w) + \int_0^w \bar{F}(w-u) f_{S_c}(u) du + \sum_u \bar{F}(w-u) \mathbb{P}\{S_d = u\}. \end{aligned}$$



(a)  $\lambda_{\max} \mathbb{E}[\tilde{X}]$  versus  $q$  for different slowdown distributions,  $d = 2$  and  $k = 1$ .  
(b)  $\mathbb{E}[R]$  versus the occupancy for different values of  $d, k$  with  $d/k = 2$ .

Figure 4.1: Numerical examples: Red( $d, k, \delta$ )

*It is not hard to see that these equations correspond to (20-21) in [39], this shows how previous work on Red( $d$ ) with i.i.d. replicas fits into our framework. Furthermore, Proposition 56 indicates how Theorem 25 from Chapter 3 for the case of identical job sizes can be obtained by focusing on the case with no slowdown (i.e.,  $g(S, X) = X$ ) (one still needs to set  $\delta = 0$  and  $k = 1$  in this result).*

**Corollary 43.** *For the Red( $d, k, \delta$ ) policy, the ccdf of the equilibrium response time distribution for the queue at the cavity is given by:*

$$\bar{F}_R(w) = \int_0^\infty \left( \sum_{j=0}^{k-1} \binom{d}{j} F_{R_x}(w)^j \bar{F}_{R_x}(w)^{d-j} \right) f_X(x) dx.$$

*Proof.* This follows from the fact that a job is finished as soon as its  $k$ 'th replica finishes. This time is given by the  $k$ 'th order statistic of  $\{U_i + g(S_i, X)\}$ .  $\square$

## Numerical examples

We pick  $g(s, x) = (s+1)x$ ,  $X$  geometric with parameter 1/2 scaled down such that  $\mathbb{E}[X] = 1$  and set  $S$  equal to zero with probability  $1 - q$  and some other distribution with mean one with probability  $q$ . In Figure 4.1a, we consider  $d = 2, k = 1, \delta = 0.01$  and plot the stability region, i.e.,  $\lambda_{\max} \mathbb{E}[\tilde{X}]$ , as a

function of the slowdown probability  $q$ . Note that this value is constant and equal to one without replication (i.e., when  $k = d = 1$ ). Here  $\lambda_{\max}$  represents the value such that the system is stable for all  $\lambda < \lambda_{\max}$ , but unstable for  $\lambda \geq \lambda_{\max}$ . As explained in Section 4.10.2  $\lambda_{\max}$  is found by looking at the smallest  $\lambda$  such that the FDE no longer has a proper solution.

We considered different slowdown distributions (when the slowdown is non-zero) namely, Erlang with 2 phases, mean 1 and SCV 1/2, exponential with mean one, and Hyperexponential with balanced means, mean one and SCV 2 and 3. We observe in Figure 4.1a that, while for  $q = 0$  the value  $\lambda_{\max}\mathbb{E}[\tilde{X}]$  is evidently the same for different slowdown distributions (as there is no slowdown), a slowdown with a higher coefficient of variation has larger values of  $\lambda_{\max}\mathbb{E}[\tilde{X}]$  for any  $q \in (0, 1]$ . Thus there is a more substantial risk to replicate if the slowdown is less variable. One perhaps surprising observation is the fact that  $\lambda_{\max}\mathbb{E}[\tilde{X}]$  is not monotone, and an optimum value of  $q$  emerges.

In Figure 4.1b we plot  $\mathbb{E}[R]$  as a function of  $\lambda$  for different values of  $d$  and  $k$  while keeping the amount of redundancy fixed to  $d/k = 2$ . We set  $k = 1, 2, 3$  (and consequently  $d = 2, 4, 6$ ) and  $q = 0.2$ . We assume that the job size is geometric with unit mean (as before) and the slowdown is exponential with unit mean. We observe that increasing the number of parts we divide the job into generally decreases the mean response time, but the value of  $\lambda_{\max}$  decreases slightly as  $k$  increases. See Section 4.10 for more details on how to obtain  $\bar{F}$  and  $\lambda_{\max}$ .

## The Red( $d, k, \delta$ ) policy in practice

The Red( $d, k, \delta$ ) policy is of interest in distributed storage systems with coding [49, 81, 82, 50]. Here, a file of size  $k \cdot X$  is encoded into  $d$  sub-files and each of these sub-files is stored on a unique server. Due to the underlying coding scheme involved, any  $k$  of the  $d$  sub-files are sufficient to retrieve the original file. With the Red( $d, k, \delta$ ) policy, once any  $k$  of the  $d$  sub-files are retrieved or downloaded, the original file can be recreated.

### 4.7.2 Type 1 : RTQ( $d, T$ )

In this section we look at the RTQ( $d, T$ ) policy (redundant to threshold queue). For this policy, we select  $d$  queues and replicate on all queues which have a workload of at most  $T$  (or assign the job randomly in case all selected

queues have a workload which exceeds  $T$ ). As soon as one replica finishes service, the others are cancelled. Such a scheme is useful in situations where the communication overhead is costly as a server only needs to send a signal to the dispatcher at the time of upcrossing or downcrossing of the threshold  $T$ . Note that the Replicate on Idle Queue RIQ( $d$ ) policy studied in [37] is a special case of this policy when  $T = 0$ . A policy that was studied by simulation in [37] is THRESHOLD- $n$  where incoming jobs are replicated on servers with at most  $n$  jobs. RTQ( $d, T$ ) can be seen as a workload equivalent of this policy. At the end of this section, we discuss how this policy can be implemented in a real system without using any knowledge about the work at the servers.

As for Red( $d, k, \delta$ ), we again find that (4.4) reduces to a DIDE without boundary condition.

**Proposition 44.** *For the RTQ( $d, T$ ), the FDE (4.4) reduces to the following DIDE:*

$$\bar{F}'(w) = -\lambda d \int_0^\infty \bar{F}_{R_x}(w)^{d-1} (\bar{F}_{R_x}(w) - \bar{F}(w)) f_X(x) dx \quad w \leq T \quad (4.12)$$

$$\begin{aligned} \bar{F}'(w) &= -\lambda d \int_0^\infty \left( B_x(w, T) (\bar{F}(T) + B_x(w, T))^{d-1} \right) f_X(x) dx \\ &\quad - \lambda \int_0^\infty (\bar{F}_{R_x}(w) - \bar{F}(w) - B_x(w, T)) \bar{F}(T)^{d-1} f_X(x) dx \quad w > T. \end{aligned} \quad (4.13)$$

with:

$$B_x(w, T) = \bar{F}_{g_x(S)}(w) F(T) + \int_0^T f_{g_x(S)}(w-u) (\bar{F}(u) - \bar{F}(T)) du.$$

Here  $B_x(w, T)$  is the probability that an arrival of size  $x$  to a single queue increases its workload from somewhere below  $T$  to a value above  $w$ .

*Proof.* We again assume a potential arrival of size  $X$  occurs to servers with workloads  $U_1, \dots, U_d$  where it experiences slowdown  $S_1, \dots, S_d$ . It is not hard to see that for the RTQ( $d, T$ ) policy, we have:

$$\mathcal{Q}(U_1) = \begin{cases} \max \{U_1, \min_{i=1}^d \{U_i + g(S_i, X) \mid U_i \leq T\}\} & \text{if } U_1 \leq T \\ U_1 + g(S_1, X) \text{ w.p. } \frac{1}{d} & \text{if } U_1, \dots, U_d > T \\ U_1 & \text{otherwise,} \end{cases}$$

where  $\min_{i=1}^d \{U_i + g(S_i, X) \mid U_i \leq T\}$  is the minimum taken over those  $i$  for which  $U_i \leq T$ . We compute the probability  $\mathbb{P}\{\mathcal{Q}(U_1) > w, U_1 \leq w, X = x\}$  the result then follows from Theorem 40 (after integrating out  $X$ ). For  $w \leq T$  we find that  $\mathbb{P}\{\mathcal{Q}(U_1) > w, U_1 \leq w, X = x\}$  is equal to:

$$\mathbb{P}\{U_1 + g_x(S_1) > w, U_1 \leq w\} \cdot \mathbb{P}\{(U_1 + g_x(S)) > w \text{ or } U_1 > T\}^{d-1}, \quad (4.14)$$

but note that if  $U > T$  then surely also  $U + g_x(S) > w$  holds. Therefore, (4.14) further simplifies to:

$$\bar{F}_{R_x}(w)^{d-1}(\bar{F}_{R_x}(w) - \bar{F}(w)),$$

this shows the first part. Assume  $w > T$ , we split  $\mathbb{P}\{\mathcal{Q}(U_1) > w, U_1 \leq w, X = x\}$  by writing it as  $\mathbb{P}\{\mathcal{Q}(U_1) > w, U_1 \leq T, X = x\} + \mathbb{P}\{\mathcal{Q}(U_1) > w, T < U_1 \leq w, X = x\}$ . When  $T < U_1$ , the workload can only increase when the workload at all other servers also exceed  $T$ , therefore we have

$$\begin{aligned} \mathbb{P}\{\mathcal{Q}(U_1) > w, T < U_1 \leq w, X = x\} &= \\ \frac{1}{d} \bar{F}(T)^{d-1} \mathbb{P}\{U_1 + g_x(S) > w, T < U_1 \leq w\}. \end{aligned} \quad (4.15)$$

For the workload to jump from below  $T$  to above  $w$ , one requires that all the other selected queues either have a workload which exceeds  $T$  (thus the job will not be replicated on them) or a response time which exceeds  $w$ . This allows us to find:

$$\begin{aligned} \mathbb{P}\{\mathcal{Q}(U_1) > w, U_1 \leq T, X = x\} &= \mathbb{P}\{U_1 + g_x(S) > w, U_1 \leq T\} \\ &\cdot (\bar{F}(T) + \mathbb{P}\{U_1 + g_x(S) > w, U_1 \leq T\})^{d-1} \end{aligned} \quad (4.16)$$

It is not hard to verify that  $B_x(w, T) = \mathbb{P}\{U_1 + g_x(S) > w, U_1 \leq T\}$ . Combining (4.14-4.16) with this equality, we may conclude the proof.  $\square$

From the equilibrium workload distribution, we obtain the response time distribution:

**Corollary 45.** *For the RTQ( $d, T$ ) policy, the ccdf of the equilibrium response*

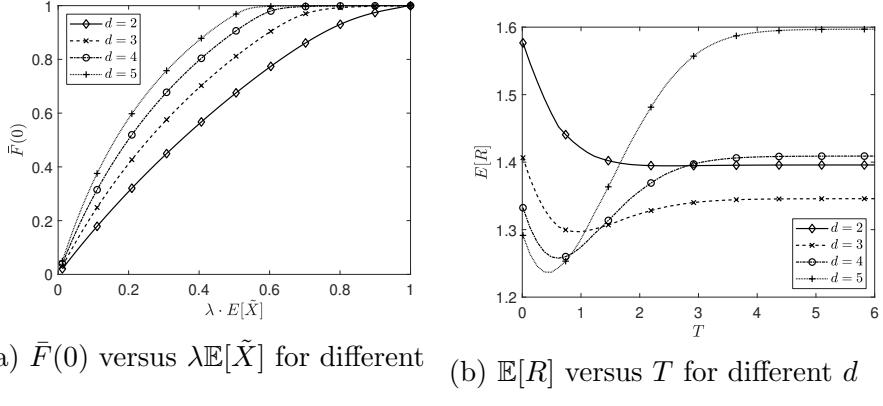


Figure 4.2: Numerical examples:  $\text{RTQ}(d, T)$

time distribution for the queue at the cavity is given by:

$$\begin{aligned} \bar{F}_R(w) &= \bar{F}(T)^d + \int_0^\infty \left( \sum_{k=1}^d \binom{d}{k} \bar{F}(T)^{d-k} B_x(w, T)^k \right) f_X(x) dx \quad w \leq T \\ \bar{F}_R(w) &= \int_0^\infty \left[ \bar{F}(T)^{d-1} (\bar{F}_{R_x}(w) - B_x(w, T)) \right. \\ &\quad \left. + \sum_{k=1}^d \binom{d}{k} \bar{F}(T)^{d-k} B_x(w, T)^k \right] f_X(x) dx \quad w > T \end{aligned}$$

where  $B_x(w, T)$  is defined as in Proposition 44.

*Proof.* For  $w \leq T$  we find that all selected queues which have a workload that exceeds  $T$  won't be able to finish the job in time  $w$ . For each of the queues which have a workload which does not exceed  $T$ , we find that  $B_x(w, T)$  is the probability that they won't finish the job before time  $w$ . For  $w > T$  we first have a term corresponding to the case where all  $d$  selected queues have a workload which exceeds  $T$ . In the second term we consider the case where  $k$  out of  $d$  selected queues have a workload smaller than  $T$ .  $\square$

## Numerical examples

We now consider some numerical examples for the  $\text{RTQ}(d, T)$  policy. We set  $d = 2, 3, 4, 5$ , assume a scaled geometric job size distribution  $X$  as for

$\text{Red}(d, k, \delta)$ , the probability of a slowdown equals  $q = 0.2$  and the slowdown distribution is exponential with unit mean. We take  $g(s, x) = (1 + s)x$  and recall that  $\tilde{X} = g(S, X)$ . In Figure 4.2a, we take  $T = 3$  and show the load of the system given by  $\bar{F}(0)$  as a function of the arrival rate for various values of  $d$ . We observe that the load  $\bar{F}(0)$  increases with  $d$  and may be close to one for moderate values of  $\lambda \mathbb{E}[\tilde{X}]$ . Nevertheless the system remains stable as long as  $\lambda \mathbb{E}[\tilde{X}] < 1$  since we never replicate on queues with a workload exceeding  $T$ . Understanding the actual system load may be of interest with respect to the energy usage of the servers.

In Figure 4.2b, we consider the same setting, but fix  $\lambda = 0.4$  and show the mean response time for the system as a function of the threshold  $T$ . We note that  $\mathbb{E}[R]$  stabilizes as  $T$  increases. This is due to the fact that for sufficiently large  $T$  the workload at all sampled queues is below the threshold and the system behaves nearly identical to the  $\text{Red}(d, 1, 0)$  policy. For  $d = 2$  we observe that the mean response time  $\mathbb{E}[R]$  decreases monotonically with  $T$ , this is due to the fact that the load is sufficiently low such that it is optimal to replicate all incoming jobs. For higher values of  $d$ , we notice that  $\mathbb{E}[R]$  initially decreases and subsequently increases. Remarkably we observe that  $d = 5$  yields the lowest mean response time by choosing the optimal  $T$  but also has the worst performance when we pick  $T$  too large. Further note that the optimal value of  $T$  decreases in  $d$ .

## The RTQ( $d, T$ ) policy in practice

This policy assumes the servers know their workload (or at least whether or not their workload exceeds some threshold  $T$ ). This information is generally not available, therefore we could implement this policy by replicating each job onto  $d$  chosen servers and cancelling the replicas which have not yet entered service after some time  $T$ . If none of the replicas started service by time  $T$ , one replica is retained at random and the other  $d - 1$  are cancelled.

**Remark 37.** *One could argue that a more realistic policy would be to assume that an incoming job is assigned to a primary server and  $d - 1$  other servers are selected at random. The replicas on the other  $d - 1$  non-primary servers get cancelled if they did not enter service by time  $T$ . When one of the replicas completes service, the other  $d - 1$  replicas are automatically cancelled. For this policy, no information on the workload is required and the only communication needed is when one job finishes service. This policy can be studied*

analogously.

#### 4.7.3 Type 2 : DR( $d, T$ )

We now analyse the Delayed Replication policy: DR( $d, T$ ). This policy has a Type 2 FDE because, as we shall see shortly, when rewriting (4.4) the right hand side depends on  $\bar{F}(u)$  for  $u > w$ . With this load balancing policy, a job is sent to an arbitrary server (which we call the primary server) on arrival. If the server has not finished service of this job within some fixed time  $T \geq 0$ , the job is replicated on  $d - 1$  other servers that are chosen uniformly at random. A cancellation on completion policy is then employed on these  $d$  replicas of the same job. Note that for  $T = 0$  this policy reduces to Red( $d$ ) while for  $T = \infty$ , it reduces to random assignment. We obtain the following result:

**Proposition 46.** *For the DR( $d, T$ ), the FDE (4.4) reduces to the following FDE:*

$$\begin{aligned} \bar{F}'(w) = -\lambda \int_0^\infty (\bar{F}_{R_x}(w) - \bar{F}(w)) \\ \left( (d-1)\bar{F}_{R_x}(w)^{d-2}\bar{F}_{R_x}(w+T) + 1 \right) f_X(x) dx & \quad w \leq T \\ \end{aligned} \tag{4.17}$$

$$\begin{aligned} \bar{F}'(w) = -\lambda \int_0^\infty (\bar{F}_{R_x}(w) - \bar{F}(w)) \\ \left( (d-1)\bar{F}_{R_x}(w)^{d-2}\bar{F}_{R_x}(w+T) + \bar{F}_{R_x}(w-T)^{d-1} \right) f_X(x) dx & \quad w > T \\ \end{aligned} \tag{4.18}$$

*Proof.* In order to apply Theorem 40, we assume a potential arrival occurs to queues with workloads  $U_1, \dots, U_d$ , and where jobs with job size equal to  $X$  experience a slowdown equal to  $S_1, \dots, S_d$ . We make the distinction whether or not the queue at the cavity is the primary server which is initially selected. We find that  $\mathbb{P}\{\mathcal{Q}(U_1) > w, U_1 \leq w\}$  equals:

$$\mathbb{P}\{\mathcal{Q}(U_1) > w, U_1 \leq w, U_1 \text{ not the primary server } \} \tag{4.19}$$

$$+ \mathbb{P}\{\mathcal{Q}(U_1) > w, U_1 \leq w, U_1 \text{ the primary server } \}. \tag{4.20}$$

For (4.19) we obtain that it is equal to:

$$\frac{d-1}{d} \int_0^\infty (\bar{F}_{R_x}(w) - \bar{F}(w)) \bar{F}_{R_x}(w)^{d-2} \cdot \bar{F}_{R_x}(w+T) f_X(x) dx, \quad (4.21)$$

which reads: the queue at the cavity has a response time which exceeds  $w$  but its workload does not exceed  $w$ , the other  $d-2$  selected servers' response times all exceed  $w$  and the primary server's response time exceeds  $w+T$ . For (4.20) we make a distinction between the cases  $w \leq T$  and  $w > T$ . For  $w \leq T$ , we find that it equals:

$$\frac{1}{d} \int_0^\infty (\bar{F}_{R_x}(w) - \bar{F}(w)) f_X(x) dx, \quad (4.22)$$

while for  $w > T$ , we find that the job is replicated onto the other servers after time  $T$  thus to finish after the queue at the cavity reaches workload  $w$ , they must process the job in at least  $w-T$  time.

$$\frac{1}{d} \int_0^\infty (\bar{F}_{R_x}(w) - \bar{F}(w)) \cdot \bar{F}_{R_x}(w-T)^{d-1} f_X(x) dx, \quad (4.23)$$

where we note that  $\bar{F}_{R_x}(w) = 1$  if  $w < 0$ . Putting (4.21-4.23) together, we obtain the sought result.  $\square$

**Remark 38.** For  $T = \infty$ ,  $\bar{F}_{R_x}(w+T) = 0$  and (4.17) reduces to:  $\bar{F}'(w) = -\lambda(\bar{F}_R(w) - \bar{F}(w))$ . It is not hard to see that this indeed corresponds to the DIDE for the random assignment policy. On the other hand, for  $T = 0$ , (4.18) reduces to  $\bar{F}'(w) = -\lambda d \int_0^\infty f_X(x)(\bar{F}_{R_x}(w) - \bar{F}(w)) \bar{F}_{R_x}(w)^{d-1} dx$ , which indeed corresponds to the Red( $d, 1, 0$ ) policy.

**Corollary 47.** For the DR( $d, T$ ) policy, the ccdf of the equilibrium response time distribution for the queue at the cavity is given by:

$$\begin{aligned} \bar{F}_R(w) &= \bar{F}_{R_{\tilde{X}}}(w) & w \leq T \\ \bar{F}_R(w) &= \int_0^\infty \bar{F}_{R_x}(w) \bar{F}_{R_x}(w-T)^{d-1} f_X(x) dx & w > T. \end{aligned}$$

*Proof.* We make a distinction between the two cases  $w > T$  and  $w \leq T$ . For  $w \leq T$ , after time  $w$  has elapsed, the job is still only being run on one server, thus the probability that it is still running after that time is the same as for random routing with job size  $\tilde{X}$ . For  $w > T$  we find that  $d-1$  other servers start serving the job after time  $T$ . As they have a backlog of time  $T$ , they should finish the job in time  $w-T$  to respond before time  $w$ .  $\square$

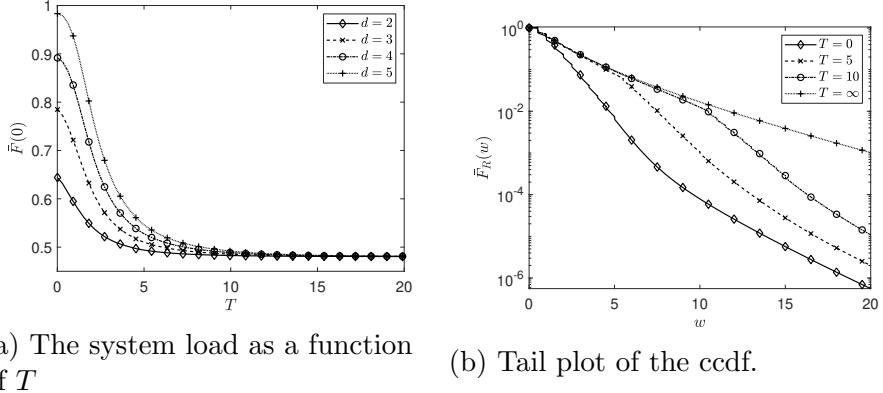


Figure 4.3: Numerical examples:  $\text{DR}(d, T)$

## Numerical examples

We again take the probability of a slowdown  $q = \mathbb{P}\{S > 0\} = 0.2$ ,  $S$  exponential (if it is non-zero),  $X$  geometric and assume that  $\lambda = 0.4$ . In Figure 4.3a we plot  $\bar{F}(0)$  as a function of  $T$ . For all values of  $d$ , we observe that the system load decreases monotonically as a function of  $T$  and converges to  $\lambda \cdot \mathbb{E}[\tilde{X}] = 0.48$  as  $T$  tends to infinity, which is the load for random routing (as expected).

In Figure 4.3b, we plot the ccdf of the response time  $\bar{F}_R(w)$  as a function of  $T$  for different values of  $d$ . We observe that initially the response time distribution is close to that of random routing, but once  $w$  passes  $T$  it quickly starts falling off in a similar fashion as in  $\text{Red}(d)$ , i.e., when  $T = 0$ . Taking another look at Figure 4.3a we observe that when setting  $T \geq 5$  delayed replication can mitigate so-called stragglers while only slightly increasing the load.

### 4.7.4 Type X : Replicate only small jobs

In replication based policies such as  $\text{Red}(d, k, \delta)$ , an incoming job is replicated on all the  $d$  sampled servers. A drawback of this approach is that the stability region of the policy is typically reduced due to the added work arising from the replicas. Therefore from a stability point of view, one may wish to replicate jobs in a selective manner. One possible alternative, also considered in [37] for RIQ, is to only replicate small jobs. For example, if we have some replication based policy (say policy 1) and another policy which does

not use replication (say policy 2), then one can design a new policy where we set some threshold  $\tilde{x} \in [0, \infty)$  and assign jobs with inherent job size  $x \leq \tilde{x}$  as per policy 1 and the remaining jobs using policy 2. To rewrite equation (4.4) for such a generic policy (say policy 3), we assume a job of size  $X$  samples  $d$  queues on arrival where it experiences slowdown  $S_1, \dots, S_d$  respectively and where the workloads are  $U_1, U_2, \dots, U_d$ . For  $i = 1, 2, 3$ , let  $\mathcal{Q}_i(U_1, U_2, \dots, U_d, S_1, \dots, S_d, X)$ , denote the new workload in the queue at cavity after a potential arrival occurs in a system with policy  $i$ . One finds:

$$\begin{aligned} & \mathbb{P}\{\mathcal{Q}_3(U_1, U_2, \dots, U_d, S_1, \dots, S_d, X) > w, U_1 \leq w\} \\ &= \int_0^{\tilde{x}} \mathbb{P}\{\mathcal{Q}_1(U_1, U_2, \dots, U_d, S_1, \dots, S_d, x) > w, U_1 \leq w\} f_X(x) dx \\ &+ \int_{\tilde{x}+}^{\infty} \mathbb{P}\{\mathcal{Q}_2(U_1, U_2, \dots, U_d, S_1, \dots, S_d, x) > w, U_1 \leq w\} f_X(x) dx, \end{aligned} \quad (4.24)$$

where  $\int_{\tilde{x}+}^{\infty}$  denotes the integral starting in  $\tilde{x}$  excluding this value. We can also easily compute the response time distribution for policy 3. Let  $R^{(i)}$  denote the response time of a job sent using policy  $i$  for  $i = 1, 2, 3$  in a system which employs policy 3. The ccdf  $\bar{F}_{R^{(i)}}, i = 1, 2$  can be computed in the same manner as for policy  $i$ , but now using the workload distribution of policy 3. The response time of a general job is then found as:

$$\bar{F}_{R^{(3)}}(w) = \mathbb{P}\{X \leq \tilde{x}\} \bar{F}_{R^{(1)}}(w) + \mathbb{P}\{X > \tilde{x}\} \bar{F}_{R^{(2)}}(w). \quad (4.25)$$

As a simple example, we now analyse a policy which applies  $\text{Red}(d, 1, 0)$  whenever the inherent job size  $X \leq \tilde{x}$  and random assignment otherwise. It is not hard to see that (4.24) simplifies to:

$$\begin{aligned} \bar{F}'(w) &= -\lambda d \int_0^{\tilde{x}} (\bar{F}_{R_x}(w) - \bar{F}(w)) \bar{F}_{R_x}(w)^{d-1} f_X(x) dx \\ &\quad - \lambda \int_{\tilde{x}+}^{\infty} (\bar{F}_{R_x}(w) - \bar{F}(w)) f_X(x) dx. \end{aligned}$$

Note that this is still a Type 1 FDE, which can be analysed in the exact same way as the regular  $\text{Red}(d, 1, 0)$  policy (c.f. Section 4.10). Moreover, letting  $X_1 = (X \mid X \leq \tilde{x})$  and  $X_2 = (X \mid X > \tilde{x})$ , we find that the response time distributions are given by:

$$\bar{F}_{R^{(1)}}(w) = \int_0^{\tilde{x}} \bar{F}_{R_x}(w)^d f_{X_1}(x) dx$$

for the jobs which are replicated and

$$\bar{F}_{R^{(2)}}(w) = \bar{F}_{g(S, X_2)}(w) + \int_0^w f_{g(S, X_2)}(w-u) \bar{F}(u) du$$

for the randomly routed jobs. One can employ (4.25) to obtain the response time distribution for an arbitrary job. There is of course nothing special about this choice for policies 1 and 2, and this approach can be employed to combine any two arbitrary policies.

#### 4.7.5 Type 3 : LL( $d, k, \delta$ )

For the Least Loaded LL( $d, k, \delta$ ) policy, when a job consisting of  $k$  equally sized parts arrives, the dispatcher sends a placeholder for this job to  $d \geq k$  FCFS servers that are sampled uniformly at random. When a placeholder reaches the head of a queue, the server informs the dispatcher and the dispatcher assigns one of the  $k$  parts of the job to the server as long as at least one part remains. We assume that the server requires a time  $\delta \geq 0$  to inform the dispatcher (and to potentially receive the part). Note that this request mechanism corresponds to the late binding mechanism used in [74]. In what follows, we first analyse the LL( $d, k$ )=LL( $d, k, 0$ ) policy. This is followed by the more general analysis for LL( $d, k, \delta$ ),  $\delta \geq 0$ .

**Remark 39.** *Conjecture 1 is proven for the LL( $d, k$ ) policy in [83].*

For the LL( $d, k$ ) policy we denote  $\mathcal{Q}(U_1) = \mathcal{Q}(U_1, \dots, U_d, S_1, \dots, S_d, X)$  for the workload of the cavity queue after a potential arrival of size  $k \cdot X$  occurs, where the  $d$  servers have workloads  $U_1, \dots, U_d$  and experience slowdowns  $S_1, \dots, S_d$ . In case  $U_1 > 0$ ,  $\mathcal{Q}(U_1)$  is equal in distribution to  $U_1 + \tilde{X}$  if  $U_1$  is one of the  $k$  least loaded servers among  $U_1, \dots, U_d$  and  $U_1$  otherwise (recall that  $\tilde{X} = g(S, X)$ ). In case  $U_1 = 0$ ,  $\mathcal{Q}(U_1)$  equals in distribution  $\tilde{X}$  with probability  $\min\left\{1, \frac{k}{|\{i|U_i=0\}|}\right\}$  and 0 otherwise.

Let  $\rho = k\lambda\mathbb{E}[\tilde{X}]$  denote the amount of work that one arrival creates. Note that a system operating under the LL( $d, k$ ) policy is stable iff  $\rho < 1$ . In the following proposition, we derive a DIDE with boundary condition which characterizes the equilibrium workload distribution:

**Proposition 48.** For the  $LL(d, k, \delta)$ , the FDE (4.4) reduces to the following DIDE:

$$\bar{F}'(w) = -\lambda \left( \bar{F}_{\tilde{X}}(w) + H(w) - \int_0^w H(u) f_{\tilde{X}}(w-u) du \right), \quad (4.26)$$

with:

$$H(w) = \sum_{j=1}^d \min\{j, k\} \binom{d}{j} \bar{F}(w)^{d-j} (1 - \bar{F}(w))^j - 1 \quad (4.27)$$

and  $\bar{F}(0) = \rho$ . Equivalently, this DIDE can be written as a FPE:

$$\bar{F}(w) = \rho - \lambda \int_0^w (1 + H(u)) \bar{F}_{\tilde{X}}(w-u) du. \quad (4.28)$$

*Proof.* Suppose that at some arbitrary point in time at equilibrium, a potential arrival of size  $k \cdot X$  arrives to servers with queue lengths  $U_1, \dots, U_d$  and slowdowns  $S_1, \dots, S_d$ . From Theorem 40 we find that the equilibrium workload environment satisfies:

$$\bar{F}'(w) = -\lambda d(\mathbb{P}\{\mathcal{Q}(U) > w, U \leq w\}) = -\lambda d\mathbb{P}\{\mathcal{Q}(U) > w, U = 0\} \quad (4.29)$$

$$-\lambda d\mathbb{P}\{\mathcal{Q}(U) > w, 0 < U \leq w\}. \quad (4.30)$$

It can be seen that (4.29) is equal to the following:

$$\begin{aligned} & -\lambda dF(0) \bar{F}_{\tilde{X}}(w) \cdot \sum_{j=0}^{d-1} \min \left\{ 1, \frac{k}{j+1} \right\} \binom{d-1}{j} \bar{F}(0)^{d-1-j} F(0)^j \\ & = -\lambda \bar{F}_{\tilde{X}}(w)(H(0) + 1) \end{aligned} \quad (4.31)$$

Similarly, (4.30) is equal to the following:

$$-\lambda d \int_0^w f(u) \sum_{j=0}^{k-1} \binom{d-1}{j} (1 - \bar{F}(u))^j \bar{F}(u)^{d-j-1} \bar{F}_{\tilde{X}}(w-u) du. \quad (4.32)$$

To further simplify this, we first define

$$h(u) = f(u) \sum_{j=0}^{k-1} \binom{d-1}{j} (1 - \bar{F}(u))^j \bar{F}(u)^{d-j-1}$$

and show that  $H'(w) = d \cdot h(w)$  where  $H(w)$  is given by (4.27). Towards this, note that

$$H'(w)/f(w) = - \sum_{j=1}^k j(d-j) \binom{d}{j} \bar{F}(w)^{d-j-1} (1-\bar{F}(w))^j \quad (4.33)$$

$$- \sum_{j=k+1}^d k(d-j) \binom{d}{j} \bar{F}(w)^{d-j-1} (1-\bar{F}(w))^j \quad (4.34)$$

$$+ \sum_{j=1}^k j^2 \binom{d}{j} \bar{F}(w)^{d-j} (1-\bar{F}(w))^{j-1} \quad (4.35)$$

$$+ \sum_{j=k+1}^d kj \binom{d}{j} \bar{F}(w)^{d-j} (1-\bar{F}(w))^{j-1}. \quad (4.36)$$

It is not hard to see that (4.35) is equal to:

$$\sum_{j=0}^{k-1} (j+1) \binom{d}{j} (d-j) \bar{F}(w)^{d-j-1} (1-\bar{F}(w))^j.$$

This allows one to find that the sum of (4.33) and (4.35) simplifies to the following.

$$\sum_{j=0}^{k-1} (d-j) \binom{d}{j} \bar{F}(w)^{d-j-1} (1-\bar{F}(w))^j - k(d-k) \binom{d}{k} \bar{F}(w)^{d-k-1} (1-\bar{F}(w))^k. \quad (4.37)$$

Similarly (4.36) can be re-written as

$$k \sum_{j=k}^{d-1} (d-j) \binom{d}{j} \bar{F}(w)^{d-j-1} (1-\bar{F}(w))^j$$

and adding this to (4.34) gives us the following.

$$k(d-k) \binom{d}{k} \bar{F}(w)^{d-k-1} (1-\bar{F}(w))^k. \quad (4.38)$$

The addition of (4.37) and (4.38) equals  $d \cdot h(w)$  which proves that  $H'(w) = d \cdot h(w)$ . We split (4.32) over its continuous and discrete part using  $\bar{F}_{\tilde{X}} =$

$\bar{F}_{\tilde{X}_c} + \bar{F}_{\tilde{X}_d}$ . Using integration by parts and the fact that  $H'(u) = d \cdot h(u)$ , for the continuous part we find:

$$\begin{aligned} \int_0^w d \cdot h(u) \bar{F}_{\tilde{X}_c}(w-u) du &= \bar{F}_{\tilde{X}_c}(0)H(w) - H(0)\bar{F}_{\tilde{X}_c}(w) \\ &\quad - \int_0^w H(u) f_{\tilde{X}_c}(w-u) du. \end{aligned} \quad (4.39)$$

For the discrete part we find by setting  $\iota(w) = \max\{n \mid x_n \leq w\}$  that:

$$\int_0^w d \cdot h(u) F_{\tilde{X}_d}(w-u) du = \bar{F}_{\tilde{X}_d}(0)H(w) - H(0)\bar{F}_{\tilde{X}_d}(w) - \sum_{j=0}^{\iota(w)} p_j H(w-x_j). \quad (4.40)$$

Using (4.31) for (4.29) and the combination of (4.39-4.40) for (4.30) we find that the sought equality indeed holds.

The FPE (4.28) follows by integrating (4.26) w.r.t.  $w$  and applying Fubini.  $\square$

**Remark 40.** In case  $k = 1$ , we find that  $H(w) = -\bar{F}(w)^d$ , and (4.28) reduces to the FPE that was obtained in Chapter 2. In particular, (4.26) yields an alternative method to compute the ccdf of the workload distribution in case job sizes are not PH distributed.

**Remark 41.** In case  $\tilde{X}$  is exponential, one can show that (4.28) is equivalent to a simple ODE:

$$\bar{F}'(w) = \rho - \bar{F}(w) - \lambda(1 + H(w)). \quad (4.41)$$

The proof goes along the same lines as the proof of Theorem 12, unfortunately for  $2 \leq k < d$  this ODE does not have a simple closed form solution.

We define  $\text{ccdf}_\rho$  as the set of all ccdfs which start in  $\rho$ , i.e. functions on  $[0, \rho]^{[0, \infty)}$  which satisfy:  $\bar{F}(0) = \rho$ ,  $\lim_{w \rightarrow \infty} \bar{F}(w) = 0$ , for all  $w, s > 0$ :  $\bar{F}(w+s) \leq \bar{F}(w)$  and  $\lim_{s \rightarrow 0^+} \bar{F}(w+s) = \bar{F}(w)$ . We can show the following:

**Proposition 49.** If we let  $T_d^{(k)} : \text{ccdf}_\rho \rightarrow [0, 1]^{[0, \infty)}$  be defined by:  $T_d^{(k)} \bar{F}(w) = \rho - \lambda \int_0^w (1 + H(u)) \bar{F}_{\tilde{X}}(w-u) du$ . Then we have  $T_d^{(k)} \bar{F} \in \text{ccdf}_\rho$  for all  $\bar{F} \in \text{ccdf}_\rho$ . Moreover for  $\bar{F}_1, \bar{F}_2 \in \text{ccdf}_\rho$  we have (with  $d_K$  the Kolmogorov distance):

$$d_K(T_d^{(k)} \bar{F}_1, T_d^{(k)} \bar{F}_2) < A(d, k, \lambda) d_K(\bar{F}_1, \bar{F}_2), \quad (4.42)$$

with:

$$A(d, k, \lambda) = \sup_{x \in [\rho, 1]} \left( d \sum_{j=1}^{k-1} (k-j)(1-x)^{d-j-1} x^j \right) \leq dk^2 \lambda^{d-k} \xrightarrow{d \rightarrow \infty} 0.$$

*Proof.* To show the first part, let  $\bar{F} \in \text{ccdf}_\rho$  be arbitrary. We note that:

$$1 + H(w) \leq k \cdot \sum_{j=0}^d \binom{d}{j} \bar{F}(w)^{d-j} (1 - \bar{F}(w))^j \leq k,$$

which shows that  $\lim_{w \rightarrow \infty} T_d^{(k)} \bar{F}(w) \leq 0$ . To show the other inequality, we note that for large values of  $w$  we have  $1 + H(w) \geq k \cdot (1 - \bar{F}(w))^j$ .

To show (4.42) we let  $\bar{F}_1, \bar{F}_2 \in \text{ccdf}_\rho$ . It is clear that we should bound:

$$\left| \sum_{j=1}^d \min\{j, k\} \binom{d}{j} \left( \bar{F}_1(w)^{d-j} (1 - \bar{F}_1(w))^j - \bar{F}_2(w)^{d-j} (1 - \bar{F}_2(w))^j \right) \right|.$$

To this end, we define the function  $f_{j,d}(x) = \sum_{j=1}^d \min\{j, k\} \binom{d}{j} x^j \cdot (1-x)^{d-j}$ . We may bound its first derivative by:

$$\begin{aligned} f'_{j,d}(x) &= \sum_{j=1}^d \min\{j, k\} \binom{d}{j} (1-x)^{d-j-1} x^{j-1} (j-d \cdot x) \\ &\leq k \sum_{j=1}^d j \binom{d}{j} (1-x)^{d-j-1} x^{j-1} - d \sum_{j=1}^d \min\{j, k\} \binom{d}{j} (1-x)^{d-j-1} x^j \\ &= d \sum_{j=1}^{k-1} (k-j) (1-x)^{d-j-1} x^j. \end{aligned}$$

By applying the mean value theorem, this completes the proof.  $\square$

**Remark 42.** As (4.26) and (4.28) are equivalent, we find (from the Banach-fixed point theorem) that all these equations have a unique solution provided that  $dk^2 \rho^{d-k} < 1$  (with boundary condition  $\bar{F}(0) = \rho$ ).

**Remark 43.** Setting  $k = 1$ , we see that  $A(d, 1, \lambda) \leq d \lambda^{d-1}$  recovering Theorem 11.

Based on our analysis for LL( $d, k$ ) we now find the following result for the more general setting with arbitrary  $\delta$ .

**Proposition 50.** *For the LL( $d, k, \delta$ ), the FDE (4.4) reduces to the following DIDE:*

$$\begin{aligned}\bar{F}'(w) &= -\lambda d(1 - \bar{F}(w)) & w \leq \delta \\ \bar{F}'(w) &= -\lambda d(\bar{F}(w - \delta) - \bar{F}(w)) \\ &\quad - \lambda \left( \bar{F}_{\tilde{X}}(w - \delta) + H(w - \delta) - \int_0^{w-\delta} H(u) f_{\tilde{X}}(w - \delta - u) du \right) & w \geq \delta,\end{aligned}$$

with boundary condition  $\bar{F}(0) = \lambda(\mathbb{E}[\tilde{X}] + d\delta)$

*Proof.* This easily follows by applying the fact that  $\mathcal{Q}(U_1) = \mathcal{Q}_{LL(d,k)}(U_1) + \delta$ , where  $\mathcal{Q}_{LL(d,k)}$  is defined as the  $\mathcal{Q}$ -function corresponding to the LL( $d, k$ ) policy. The boundary condition follows from the fact that each job brings  $\mathbb{E}[\tilde{X}] + d\delta$  work on average.  $\square$

If there is no slowdown (i.e.  $g(s, x) = x$ ) and jobs are split into  $k$  identical parts, then it is obvious that the response time of a job is given by its response time at the server with the  $k$ 'th lowest workload for which the ccdf is easy to compute. When computing the response time for the LL( $d, k, \delta$ ) policy in the  $S\&X$  model, things are more involved. The queue with the highest workload need not be the last queue to finish serving its part of the job. The response time of a job is given by  $R = \max_{i=1}^k \{U_{(i)} + g(S_i, X)\}$ . We find:

$$\bar{F}_R(w + \delta) = 1 - \int_0^\infty \mathbb{P} \left\{ \max_{i=1}^k \{U_{(i)} + g_x(S_i)\} \leq w \right\} f_X(x) dx$$

and  $\bar{F}_R(w) = 1$  for  $w \leq \delta$ . By applying the fact that the pdf of the joint distribution  $(U_{(1)}, \dots, U_{(k)})$  in  $u_1, \dots, u_k$  is given by  $\frac{d!}{(d-k)!} f(u_1) \cdots f(u_k) \bar{F}(u_k)^{n-k}$  we find:

$$\begin{aligned}\mathbb{P} \left\{ \max_{i=1}^k \{U_{(i)} + g_{\frac{x}{k}}(S_i)\} \leq w \right\} &= \frac{d!}{(d-k)!} \int_0^w f(u_1) F_{g_{\frac{x}{k}}(S)}(w - u_1) \\ &\quad \int_{u_1}^w f(u_2) F_{g_{\frac{x}{k}}(S)}(w - u_2) \int_{u_2}^w \cdots \int_{u_{k-1}}^w F_{g_{\frac{x}{k}}(S)}(w - u_k) f(u_k) \bar{F}(u_k)^{d-k} du_k \dots du_1.\end{aligned}\tag{4.43}$$

When  $d > k \geq 2$ , this integral becomes hard to solve, we may therefore first compute the workload distribution and thereafter use simulation to obtain

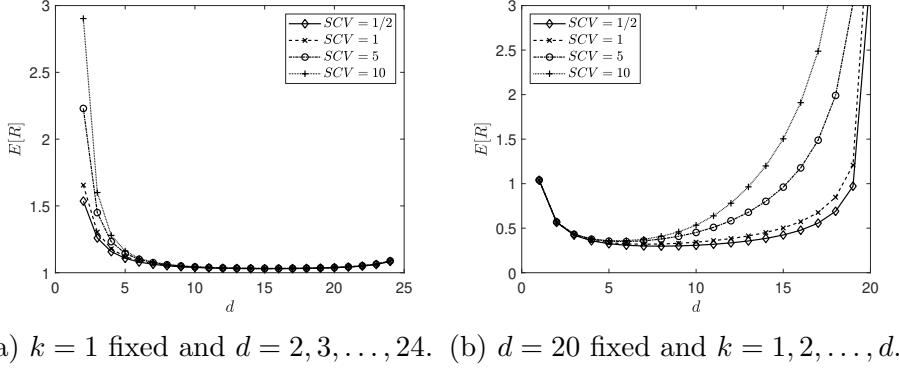


Figure 4.4: Numerical examples:  $\text{LL}(d, k, \delta)$

the response time distribution. More precisely, whenever an arrival occurs, we simulate  $X, S_1, \dots, S_d$  according to their distribution and  $U_1, \dots, U_d$  as i.i.d. random variables distributed as the obtained workload distribution. One can then simply apply the  $\text{LL}(d, k, \delta)$  policy to obtain the response time for this specific set of simulated values. Note that in this simulation one need not keep track of any values, simply simulate arrivals and compute their response times based on the obtained workload distribution.

## Numerical examples

We investigate how the choice of  $d$  and  $k$  impact the mean response time in the  $\text{LL}(d, k, \delta)$  policy described above in case there is no slowdown and each incoming job is split into  $k$  equal parts. We fix  $\lambda = 0.8$  and  $\delta = 0.01$ , for the jobs we take Erlang distributed job sizes with mean 1 and SCV  $1/2$ , exponential job sizes with mean one, and Hyperexponential job sizes with balanced means, unit mean and SCV 5 resp. 10.

In Figure 4.4a, we observe that increasing the value of  $d$  decreases the mean response time where the gain is greater for the more variable jobs. As  $d$  becomes large, the mean response times seem to coincide for all values of the SCV. For  $d$  sufficiently large the mean response time starts to increase due to the extra load coming from the request time  $\delta$ . Note that the system becomes unstable for  $d = 25$  as the system load is  $0.8 + 0.01 \cdot 25 \cdot 0.8 = 1$ . In Figure 4.4b, we observe that splitting a job into multiple parts initially yields a gain in response times, even driving response times below 1. As we divide

jobs into more and more parts, we observe that the response times start to increase as the number of servers we can choose decreases. Note that there is a very strong increase when going from  $k = 19 = d - 1$  to  $k = 20 = d$  jobs, this is due to the fact that at  $k = d$  we completely loose the power of  $d$  choices.

## The $\text{LL}(d, k, \delta)$ policy in practice

The  $\text{LL}(d, k)$  policy finds its application in systems with parallel processing and multi-task jobs [97]. For such applications, one can reinterpret the  $\text{LL}(d, k)$  policy in a slightly different manner as follows. We assume that an arriving job has a service requirement of  $k \cdot X$  which is split into  $k$  identical parts. An arriving job samples  $d$  servers at random and the  $k$  subtasks of the job receive service from the  $k$  least-loaded servers among the  $d$  that were sampled. We say a job has been processed when all of its subtasks are processed. Note that while in  $\text{Red}(d, k)$ , we can have upto  $d$  subtasks being served simultaneously (due to its cancellation on completion nature), in  $\text{LL}(d, k)$  one can have at most  $k$  subtasks to be simultaneously in service. In this sense,  $\text{LL}(d, k)$  can be viewed, as a cancellation on start version of the  $\text{Red}(d, k)$  policy.

**Remark 44.** *For applications such as distributed storage, we need to consider the  $\text{LL}(d, k)$  policy with  $k$  identically distributed (not necessarily independent nor identical) jobs denoted by  $X_1 \dots X_k$ . Let  $\tilde{\mathcal{Q}}$  denote the  $\mathcal{Q}$  function associated to the  $\text{LL}(d, k)$  policy with identically distributed sub-jobs. It is not hard to see that if we let  $X \stackrel{d}{=} X_1$  and let  $S_1, \dots, S_d$  denote independent slowdown variables we have:*

$$\tilde{\mathcal{Q}}(U_1, \dots, U_d, S_1, \dots, S_d, X_1, \dots, X_k) \stackrel{d}{=} \mathcal{Q}(U_1, \dots, U_d, S_1, \dots, S_d, X).$$

Therefore we have:

$$\mathbb{P}\{\tilde{\mathcal{Q}}(U) > w, U \leq w\} = \mathbb{P}\{\mathcal{Q}(U) > w, U \leq w\},$$

which entails that the FDE one finds for this more general model is the same as for simply taking identical job sizes. Thus, the analysis is the same as for identical job sizes, and we may restrict ourselves to the case of identically distributed sub-jobs. Note that this is indeed consistent with the encoding of  $X$  into  $d$  sub files: only the first  $k$  sub files are used.

Additional variants to  $\text{LL}(d, k, \delta)$  are for example: sending the largest job to the least loaded server, the second largest to the second least loaded etc. One could also allow to send multiple jobs to the same server if that server has a very low load.

#### 4.7.6 Type 4 : JTQ( $d, T$ )

In the  $\text{JTQ}(d, T)$  policy, on arrival of a job,  $d$  servers are sampled and the job is served on a randomly chosen server (among the  $d$  servers) which has a workload below the threshold  $T$ . In case none of the sampled  $d$  servers has a workload of at most  $T$ , the job is randomly routed to one of the  $d$  servers. When  $T = \infty$ , this reduces to random routing. Like the  $\text{RTQ}(d, T)$  policy,  $\text{JTQ}(d, T)$  is a policy that has low communication overhead as the servers need to only inform the dispatcher about their level crossings of the threshold  $T$ .

We can use the following proposition to compute its equilibrium workload distribution.

**Proposition 51.** *For the  $\text{JTQ}(d, T)$ , the FDE (4.4) reduces to the following FDE:*

$$\begin{aligned}\bar{F}'(w) &= -\lambda \left( \frac{1 - \bar{F}(T)^d}{1 - \bar{F}(T)} (\bar{F}_{R_{\tilde{X}}}(w) - \bar{F}(w)) \right) & w \leq T \\ \bar{F}'(w) &= -\lambda \left( \frac{1 - \bar{F}(T)^d}{1 - \bar{F}(T)} A(w, T) \right. \\ &\quad \left. + \bar{F}(T)^{d-1} \left[ \bar{F}_{R_{\tilde{X}}}(w) - \bar{F}(w) - A(w, T) \right] \right) & w > T,\end{aligned}$$

with  $A(w, T) = \bar{F}_{\tilde{X}}(w)F(T) + \int_{w-T}^w f_{\tilde{X}}(u)(\bar{F}(w-u) - \bar{F}(T)) du$  and  $R_{\tilde{X}}$  the marginal response time. Here we have the boundary condition  $\bar{F}(0) = \lambda \mathbb{E}[\tilde{X}]$ .

*Proof.* Note that for this policy,  $\mathcal{Q}(U_1) = \mathcal{Q}(U_1, \dots, U_d, \tilde{X})$  is given by:

$$\begin{aligned}\mathcal{Q}(U_1) &= U_1 + \tilde{X} & w.p. \frac{1}{|\{k \in \{1, \dots, d\} \mid U_k \leq T\}|} \text{ if } U_1 \leq T \\ \mathcal{Q}(U_1) &= U_1 + \tilde{X} & w.p. \frac{1}{d} \text{ if } U_1, \dots, U_d > T \\ \mathcal{Q}(U_1) &= U_1 & \text{otherwise.}\end{aligned}$$

We first compute the probability  $\mathbb{P}\{\mathcal{Q}(U_1) > w, U_1 \leq w\}$  for the case  $w \leq T$ . We find that it is equal to:

$$\begin{aligned} & \left( \sum_{j=0}^{d-1} \binom{d-1}{j} F(T)^j \bar{F}(T)^{d-1-j} \frac{1}{j+1} \right) \mathbb{P}\{U_1 + \tilde{X} > w, U_1 \leq w\} \\ &= \left( \frac{1}{d} \sum_{j=1}^d \binom{d}{j} \frac{F(T)^j \bar{F}(T)^{d-j}}{F(T)} \right) \mathbb{P}\{U_1 + \tilde{X} > w, U_1 \leq w\} \\ &= \frac{1 - \bar{F}(T)^d}{dF(T)} \mathbb{P}\{U_1 + \tilde{X} > w, U_1 \leq w\}. \end{aligned}$$

This shows the first part. Assume  $w > T$ , we first write  $\mathbb{P}\{\mathcal{Q}(U_1) > w, U_1 \leq w\}$  as  $\mathbb{P}\{\mathcal{Q}(U_1) > w, U_1 \leq T\} + \mathbb{P}\{\mathcal{Q}(U_1) > w, T < U_1 \leq w\}$ . We then find that  $\mathbb{P}\{\mathcal{Q}(U_1) > w, U_1 \leq T\}$  is given by:

$$\frac{(1 - \bar{F}(T)^d)}{dF(T)} \mathbb{P}\{U_1 + \tilde{X} > w, U_1 \leq T\}.$$

while  $\mathbb{P}\{\mathcal{Q}(U_1) > w, T < U_1 \leq w\}$  is given by:

$$\begin{aligned} & \frac{1}{d} \mathbb{P}\{U_2, \dots, U_d > T, U + \tilde{X} > w, T < U \leq w\} \\ &= \frac{1}{d} \bar{F}(T)^{d-1} (\bar{F}_{R_X}(w) - \bar{F}(w) - \mathbb{P}\{U \leq T, U + \tilde{X} > w\}). \end{aligned}$$

One finds that  $A(w, T) = \mathbb{P}\{U \leq T, U + \tilde{X} > w\}$ , which completes the proof.  $\square$

**Remark 45.** For numerical considerations we may define  $H(w) = \bar{F}(w)\delta_{\{w \leq T\}}$ , we find that  $A(w, T) = \bar{G}(w) + \int_0^w g(w-u)H(u)du$  which can be quickly computed as it is a convolution. Here  $\delta_{\{w \leq T\}}$  is one when  $w \leq T$  and zero otherwise.

**Corollary 52.** For the  $JTQ(d, T)$  policy, the ccdf of the equilibrium response time distribution for the queue at the cavity is given by:

$$\begin{aligned} \bar{F}_R(w) &= \frac{1 - \bar{F}(T)^d}{1 - \bar{F}(T)} \left( \bar{F}_{\tilde{X}}(w)F(T) + \int_0^T f_{\tilde{X}}(w-u)(\bar{F}(u) - \bar{F}(T)) du \right) \\ &+ \bar{F}(T)^{d-1} \left( \bar{F}_{\tilde{X}}(w-T)\bar{F}(T) + \int_0^{w-T} f_{\tilde{X}}(w-T-u)\bar{F}(T+u) du \right) \end{aligned}$$

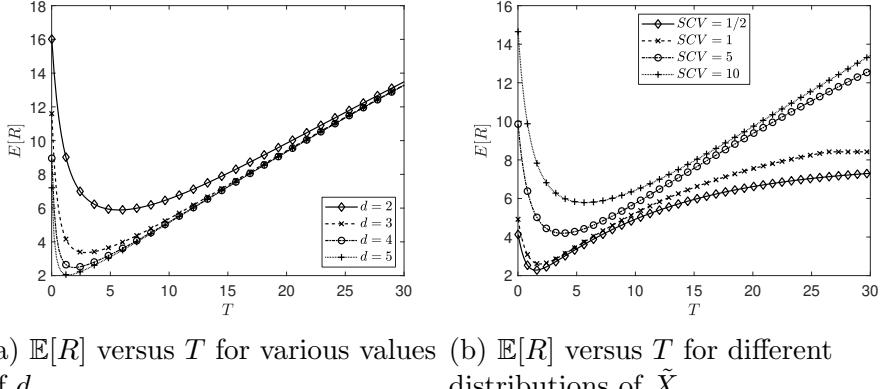


Figure 4.5: Numerical examples:  $JTQ(d, T)$

*Proof.* This follows from the fact that  $\bar{F}_R(w)$  is given by:

$$(1 - \bar{F}(T)^d)\mathbb{P}\{U + \tilde{X} > w \mid U \leq T\} + \bar{F}(T)^d\mathbb{P}\{U + \tilde{X} > w \mid U > T\}.$$

□

## Numerical examples

We now consider some numerical examples for the  $JTQ(d, T)$  policy without slowdown. In Figure 4.5a, we compare the mean response time  $\mathbb{E}[R]$  as a function of  $T$  for different values of  $d$ . We assume that  $\lambda = 0.9$  and  $\tilde{X}$  is exponential with unit mean. First, note that  $T = 0$  corresponds to  $JIQ(d)$ . On the other hand,  $T = \infty$  corresponds to the random routing policy. We see that for different values of  $d$ ,  $\mathbb{E}[R]$  first decreases and then increases to the same value (due to the resemblance to random routing policy for large  $T$ ) indicating that the parameter  $T$  should be chosen suitably. Furthermore, the optimal value of  $T$  decreases with increase in  $d$ . Note that as  $d$  tends to infinity,  $JIQ(d)$  achieves vanishing delays as expected.

In Figure 4.5b, we show  $\mathbb{E}[R]$  versus  $T$  with different distributions for  $\tilde{X}$ : Erlang with mean 1 and SCV 1/2, exponential with mean one, and Hyperexponential with balanced means, mean one and SCV 5 and 10. We set  $\lambda = 0.9$  and  $d = 2$ . For the different distributions, we notice that  $\mathbb{E}[R]$  again has the same shape as in Figure 4.5a. Furthermore, the optimal value of  $T$  increases with the SCV.

## 4.8 Phase type distribution

While the analysis presented till now assumes that the job sizes and slow-down variables are generally distributed, in this section, we focus on the case where certain random variables have a PH distributions and for all  $x$ ,  $g_x^{-1}(w)$  is linear in  $w$ . Note that any distribution on  $[0, \infty)$  can be approximated arbitrarily close with a PH distribution, moreover there are various tools available online for matching PH distributions (see e.g. [52], [75]).

A PH distribution with pdf  $b(\cdot)$  and cdf  $B(\cdot)$  with  $B(0) = 0$  is fully characterized by a stochastic vector  $\alpha \in \mathbb{R}^n$  and a subgenerator matrix  $A \in \mathbb{R}^{n \times n}$  such that  $\bar{B}(w) = \alpha e^{Aw} \mathbf{1}$  and  $b(w) = \alpha e^{Aw} \mu$  with  $n \in \mathbb{N}$ ,  $\mu = -A\mathbf{1}$  and  $\mathbf{1}$  an  $n \times 1$  column vector consisting of ones. We note that for the choice of  $g(s, x) = (s+1)x$  that we consider in our numerical examples, and the choice of  $g(s, x) = sx$  as considered in [37],  $g_x^{-1}(w)$  is indeed linear in  $w$ .

The importance of the results in this section mostly relate to the computation time of the numerical solution methods. The idea is to replace integral equations with a system of differential equations which are numerically less cumbersome to compute. If we let  $M$  denote the number of control points used to define  $\bar{F}$ , the results involving PH distributions generally reduce computational complexity by one order. For example, we find that solving the DIDE given in Proposition 48 with a discrete job size distribution  $X$  requires  $O(M^2)$  time, whilst the DDE given by Proposition 55 can be solved in  $O(M)$  time.

### 4.8.1 Red( $d, k, \delta$ )

For the Red( $d, k, \delta$ ) policy, the integral is hidden in  $\bar{F}_{R_x}(w)$  (see (4.7)). We show that a simplification in the analysis is possible for obtaining  $\bar{F}_{R_x}$ . Note that this speed-up applies to all policies which use  $\bar{F}_{R_x}$  in their associated FDE:

**Proposition 53.** *Assume  $g_x^{-1}(w)$  is linear in  $w$ . Let  $a(x) = \frac{\partial(g_x)^{-1}}{\partial w}$  and assume  $S$  is PH distributed with parameters  $(\alpha, A)$ . We find:*

$$\begin{aligned}\bar{F}_{R_x}(w) &= \bar{F}_S(g_x^{-1}(w)) + \alpha a(x) \xi_x(w) \\ \xi'_x(w) &= \bar{F}(w - g_x(0))\mu + A\xi_x(w)a(x), & w > g_x(0) \\ \xi_x(w) &= 0 & w \leq g_x(0),\end{aligned}$$

with  $\mu = -A\mathbf{1}$ .

*Proof.* As  $S$  is PH distributed, we find:

$$\bar{F}_{R_x}(w) = \bar{F}_S(g_x^{-1}(w)) + \alpha \int_{g_x(0)}^w \bar{F}(w-u) e^{g_x^{-1}(u)A} \mu du a(x).$$

The result follows by letting  $\xi_x(w) = \int_{g_x(0)}^w \bar{F}(w-u) e^{g_x^{-1}(u)A} \mu du$ . Indeed, we first use a substitution to write it as:  $\xi_x(w) = \int_{g_x(0)}^w \bar{F}(u) e^{g_x^{-1}(w-u)A} \mu du$ , one may then simply apply the Leibniz integral rule to differentiate  $\xi_x(w)$ .  $\square$

**Remark 46.** Using the result in Proposition 53 we can rewrite the DIDE given in (4.9-4.10) as:

$$\begin{aligned} \bar{F}'(w) &= -\lambda d \left[ \int_0^\infty (\bar{F}_S(g_x^{-1}(w)) + \alpha a(x) \xi_x(w)) f_X(x) dx - \bar{F}(w) \right] & w \leq \delta \\ \bar{F}'(w) &= -\lambda d \left( \int_0^\infty \sum_{j=0}^{k-1} \binom{d-1}{j} \left( 1 - \bar{F}_S(g_x^{-1}(w-\delta)) \right. \right. \\ &\quad \left. \left. + \alpha a(x) \xi_x(w-\delta) \right)^j (\bar{F}_S(g_x^{-1}(w-\delta)) + \alpha a(x) \xi_x(w-\delta))^{d-j-1} \right. \\ &\quad \cdot \left. \left( \bar{F}_S(g_x^{-1}(w)) + \alpha a(x) \xi_x(w) - \bar{F}(w) \right) f_X(x) dx \right) & w > \delta \\ \xi_x(w) &= 0 & w \leq g_x(0) \\ \xi'_x(w) &= \bar{F}(w-g_x(0))\mu + A\xi_x(w)a(x), & w > g_x(0). \end{aligned}$$

It is not hard to see how to generalize this result in case  $S$  is a combination of a discrete and a PH distributed random variable. In our numerical examples, we assumed that  $S$  is PH distributed with probability  $q$  and zero with probability  $1-q$  and  $g(s, x) = (s+1)x$ . Let us denote  $\mathcal{A} = (S \mid S > 0)$  the PH distribution  $S$  has with probability  $q$ , assume it has parameters  $(\alpha, A)$  and let  $\mu = -A\mathbf{1}$ . It is not hard (See also the proof of Proposition 54) to show that:

$$\begin{aligned} \bar{F}_{R_x}(w) &= 1 & \text{if } w \leq x \\ \bar{F}_{R_x}(w) &= q \left( \bar{F}_{\mathcal{A}} \left( \frac{w-x}{x} \right) + \frac{\alpha \xi_x(w)}{x} \right) + (1-q)\bar{F}(w-x) & \text{if } w > x \\ \xi_x(w) &= 0 & \text{if } w \leq x \\ \xi'_x(w) &= \bar{F}(w-x)\mu + \frac{A\xi_x(w)}{x} & \text{if } w > x. \end{aligned}$$

The case of no slowdown and PH distributed job sizes can be found in Section 4.9.

### 4.8.2 RTQ( $d, T$ )

For the RTQ( $d, T$ ) policy, we have an additional integral for  $B_x(w, T)$  besides  $\bar{F}_{R_x}(w)$ . We show how this integral can be eliminated in case  $S$  is a combination of a discrete and a PH distribution:

**Proposition 54.** *If in the setting of Proposition 44,  $S$  is PH distributed with parameters  $(\alpha, A)$ ,  $\mu = -A\mathbf{1}$  with probability  $q$ , and zero otherwise and  $(g_x^{-1})'(w) = a(x)$  does not depend on  $w$ , then:*

$$B_x(w, T) = \bar{F}_S(g_x^{-1}(w))F(T) + q\alpha\varphi_x(w)a(x) + (1-q)(\bar{F}(w - g_x(0)) - \bar{F}(T))I_{[g_x(0), T+g_x(0)]}(w),$$

where  $\varphi_x(w)$  satisfies:

$$\begin{aligned} \varphi_x(w) &= 0 & w \leq g_x(0) \\ \varphi'_x(w) &= (\bar{F}(w - g_x(0)) - \bar{F}(T))\mu + A\varphi_x(w)a(x) & g_x(0) < w \leq T + g_x(0) \\ \varphi'_x(w) &= A\varphi_x(w)a(x) & T + g_x(0) < w. \end{aligned}$$

*Proof.* Let  $\mathcal{A} = (S \mid S > 0)$ , one finds that  $\int_0^T f_{g_x(S)}(w-u)(\bar{F}(u) - \bar{F}(T)) du$  equals:

$$q \int_0^T f_{g_x(\mathcal{A})}(w-u)(\bar{F}(u) - \bar{F}(T)) du + (1-q)(\bar{F}(w - g_x(0)) - \bar{F}(T))I_{[g_x(0), T+g_x(0)]}(w).$$

Furthermore as  $g_x(\mathcal{A}) \geq g_x(0)$  we find that  $f_{g_x(\mathcal{A})}(w-u) = 0$  for  $w-u < g_x(0)$  which happens if  $w - g_x(0) < u$ . The result now follows by setting:

$$\varphi_x(w) = \int_0^{\min\{T, (w-g_x(0))\}} e^{g_x^{-1}(w-u)A}(\bar{F}(u) - \bar{F}(T)) du \mu.$$

□

As for Red( $d, k, \delta$ ), we obtain an alternative characterization of the equilibrium workload distribution for the RTQ( $d, T$ ) policy by combining Proposition 44, 53 and 54. The case of no slowdown and PH distributed job sizes is discussed in Section 4.9.

### 4.8.3 LL( $d, k, \delta$ )

In this section, we look at the scenario where the actual job size  $\tilde{X} = g(S, X)$  is PH distributed. It was already noted in Chapter 2 and Chapter 3 that for the LL( $d$ ) policy, when job sizes are PH distributed, the associated DIDE can be reduced to a DDE which can be solved more efficiently. We show a similar result for LL( $d, k, \delta$ ).

**Proposition 55.** *If  $\tilde{X}$  is PH distributed with parameters  $(\alpha, A)$  we find that the DIDE given in Proposition 50 simplifies to the following DDE:*

$$\begin{aligned}\bar{F}'(w) &= -\lambda d(1 - \bar{F}(w)) && \text{if } w \leq \delta \\ \bar{F}'(w) &= -\lambda d(\bar{F}(w - \delta) - \bar{F}(w)) - \lambda \left( \bar{F}_{\tilde{X}}(w - \delta) \right. \\ &\quad \left. + H(w - \delta) - \alpha \xi(w - \delta) \right) && \text{if } w \geq \delta \\ \xi'(w) &= A \xi(w) + H(w) \mu,\end{aligned}$$

with boundary condition  $\xi(0) = 0$  and  $\mu = -A \mathbf{1}$ .

*Proof.* This easily follows from Proposition 48 by noting that  $f_{\tilde{X}}(w) = \alpha e^{wA} \mu$  and setting  $\xi(w) = \int_0^w H(u) f_{\tilde{X}}(w-u) du$ .  $\square$

This result can be further generalized in case  $\tilde{X}$  is a combination of a discrete and a PH distribution.

## 4.9 No slowdown

In this section we take another look at some of the policies studied in Section 4.7, and revisit them under the assumption that the servers experience no slowdown (i.e.  $g(S, X) = X$ ). We first note that the analysis for LL( $d, k, \delta$ ) and JTQ( $d, T$ ) under the no slowdown assumption easily follows by taking  $\tilde{X} = X$ . We now focus on the Red( $d, k, \delta$ ), RTQ( $d, T$ ) and DR( $d, T$ ) policy for the case without slowdown.

### 4.9.1 Red( $d, k, \delta$ )

For Red( $d, k, \delta$ ) with no slowdown, we find that the results in Proposition 42 can be simplified, which allows to obtain an analog to Proposition 53 in case  $X$  is PH distributed.

**Proposition 56.** *The ccdf of the workload distribution for  $\text{Red}(d, k, \delta)$  without slowdown satisfies the following DIDE:*

$$\bar{F}'(w) = -\lambda d \left( \bar{F}_X(w) + \int_0^w f_X(w-u) \bar{F}(u) du - \bar{F}(w) \right) \quad w \leq \delta \quad (4.44)$$

$$\bar{F}'(w) = -\lambda d \left[ \bar{F}_X(w) - \bar{F}(w) \bar{F}_X(w-\delta) \right. \quad (4.45)$$

$$\begin{aligned} &+ \int_0^\delta \bar{F}(x) f_X(w-x) dx \\ &+ \left. \int_0^{w-\delta} L(w-x-\delta) (\bar{F}(w-x) - \bar{F}(w)) f_X(x) dx \right] \quad w > \delta \end{aligned} \quad (4.46)$$

with  $L(w) = \sum_{j=0}^{k-1} \binom{d-1}{j} F(w)^j \bar{F}(w)^{d-j-1}$ . Furthermore if  $X$  is PH distributed with parameters  $(\alpha, A)$  and  $\mu = -A\mathbf{1}$  we find that the above DIDE reduces to a DDE:

$$\bar{F}'(w) = -\lambda d \left( \bar{F}_X(w) + \alpha \xi_1(w) - \bar{F}(w) \right) \quad w \leq \delta$$

$$\begin{aligned} \bar{F}'(w) &= -\lambda d \left( \bar{F}_X(w) - \bar{F}(w) \bar{F}_X(w-\delta) \right. \\ &\quad \left. + \alpha (\xi_1(w) + \xi_2(w) - \xi_3(w) \bar{F}(w)) \right) \quad w > \delta \end{aligned}$$

$$\xi'_1(w) = A \xi_1(w) + \bar{F}(w) \mu \quad w \leq \delta$$

$$\xi'_1(w) = A \xi_1(w) \quad w > \delta$$

$$\xi'_2(w) = L(w-\delta) \bar{F}(w) \mu + A \xi_2(w) \quad w > \delta$$

$$\xi'_3(w) = L(w-\delta) \mu + A \xi_3(w) \quad w > \delta.$$

with boundary condition  $\xi_1(0) = \xi_2(\delta) = \xi_3(\delta) = 0$ .

*Proof.* We find:

$$\bar{F}_{R_{\tilde{X}}}(w) = \mathbb{P}\{U + X \geq w\} = \bar{F}_X(w) + \int_0^w f_X(w-x) \bar{F}(x) dx.$$

Moreover we have:

$$\bar{F}_{R_x}(w) = \bar{F}(w-x).$$

This allows us to compute:

$$\begin{aligned}\bar{F}'(w) &= -\lambda d \int_0^\infty L(w-x-\delta) (\bar{F}(w-x) - \bar{F}(w)) f_X(x) dx \\ &= -\lambda d \int_0^w L(w-x-\delta) (\bar{F}(w-x) - \bar{F}(w)) f_X(x) dx - \lambda d \bar{F}_X(w) F(w).\end{aligned}$$

From this it is clear that (4.44-4.46) holds.

Now assume that  $X$  is PH distributed with parameters  $(\alpha, A)$ . This last statement follows by defining:

$$\begin{aligned}\xi_1(w) &= \int_0^w e^{(w-x)A} \bar{F}(x) dx \mu & w \leq \delta \\ \xi_1(w) &= \int_0^\delta e^{(w-x)A} \bar{F}(x) dx \mu & w > \delta \\ \xi_2(w) &= \int_0^{w-\delta} L(x) \bar{F}(x+\delta) e^{(w-x-\delta)A} \mu dx \\ \xi_3(w) &= \int_0^{w-\delta} L(x) e^{(w-x-\delta)A} \mu dx\end{aligned}$$

□

### 4.9.2 RTQ( $d$ )

For RTQ( $d, T$ ) with no slowdown, we find that the result in Proposition 44 can be simplified, which allows to obtain an analogous simplification to Proposition 53, 54 in case  $X$  is PH distributed.

**Proposition 57.** *The ccdf of the workload distribution for RTQ( $d$ ) without*

slowdown satisfies the following DIDE:

$$\begin{aligned}\bar{F}'(w) = -\lambda d & \left[ F(w)\bar{F}_X(w) + \int_0^w \bar{F}(w-x)^{d-1} \right. \\ & \left. (\bar{F}(w-x) - \bar{F}(w)) f_X(x) dx \right] \quad w \leq T \end{aligned}\quad (4.47)$$

$$\begin{aligned}\bar{F}'(w) = -\lambda d & \left[ F(T)\bar{F}_X(w) + \int_0^T \bar{F}(T-x)^{d-1} \right. \\ & \left. (\bar{F}(T-x) - \bar{F}(T)) f_X(x+w-T) dx \right] \\ & - \lambda \bar{F}(T)^{d-1} \left[ (\bar{F}(T) - \bar{F}(w)) \bar{F}_X(w-T) \right. \\ & \left. + \int_0^{w-T} (\bar{F}(w-x) - \bar{F}(w)) f_X(x) dx \right] \quad w > T \end{aligned}\quad (4.48)$$

When  $X$  has a PH distribution with parameters  $(\alpha, A)$  and  $\mu = -A\mathbf{1}$  we find that (4.47)-(4.48) simplifies to the following DDE:

$$\begin{aligned}\bar{F}'(w) = -\lambda d & \left[ F(T)\bar{F}_X(w) + \alpha \cdot (\xi_1(w) - \xi_2(w)\bar{F}(w)) \right] \quad w \leq T \\ \bar{F}'(w) = -\lambda d & \left[ F(T)\bar{F}_X(w) + \alpha (\xi_1(w) - \xi_2(w)\bar{F}(T)) \right] - \lambda \bar{F}(T)^{d-1} \\ & \left[ (\bar{F}(T) - \bar{F}(w))\bar{F}_X(w-T) + \alpha (\xi_3(w) - \bar{F}(w)F_X(w-T)) \right] \quad w > T \\ \xi'_1(w) = A\xi_1(w) & + \bar{F}(w)^d \mu \quad w \leq T \\ \xi'_1(w) = A\xi_1(w) & \quad w > T \\ \xi'_2(w) = A\xi_2(w) & + \bar{F}(w)^{d-1} \mu \quad w \leq T \\ \xi'_2(w) = A\xi_2(w) & \quad w > T \\ \xi'_3(w) = A\xi_3(w) & + \bar{F}(w)\mu. \end{aligned}$$

With boundary condition  $\xi_1(0) = \xi_2(0) = \xi_3(T) = 0$ .

*Proof.* We first note that we have  $\bar{F}_{R_x}(w) = \bar{F}(w-x)$  and  $B_x(w, T) = 0$  if  $T \leq w-x$  and  $\bar{F}(w-x) - \bar{F}(T)$  if  $w-x < T$ . This allows us to find for

$w \leq T$ :

$$\bar{F}'(w) = -\lambda d \int_0^\infty f_X(x) \bar{F}(w-x)^{d-1} (\bar{F}(w-x) - \bar{F}(w)) dx$$

which easily simplifies to (4.47). For  $w > T$  we obtain:

$$\begin{aligned} \bar{F}'(w) &= -\lambda d \int_{w-T}^\infty (\bar{F}(w-x) - \bar{F}(T)) \bar{F}(w-x)^{d-1} f_X(x) dx \\ &\quad - \lambda \int_0^\infty \left[ \int_0^{w-T} (\bar{F}(w-x) - \bar{F}(w)) f_X(x) dx \right. \\ &\quad \left. + \int_{w-T}^\infty (\bar{F}(T) - \bar{F}(w)) f_X(x) dx \right]. \end{aligned} \quad (4.49)$$

In order to conclude that (4.48) indeed holds, it suffices to note that (4.49) is equal to:

$$\begin{aligned} &- \lambda d \left[ \int_{w-T}^w (\bar{F}(w-x) - \bar{F}(T)) \bar{F}(w-x)^{d-1} f_X(x) dx \right. \\ &\quad \left. + \int_w^\infty (1 - \bar{F}(T)) f_X(x) dx \right]. \end{aligned}$$

The result for PH distributed job sizes  $X$  follows by defining:

$$\begin{aligned} \xi_1(w) &= \int_0^w \bar{F}(w-x)^d e^{xA} \mu dx & w \leq T \\ \xi_1(w) &= \int_0^T \bar{F}(T-x)^d e^{(x+w-T)A} dx & w > T \\ \xi_2(w) &= \int_0^w \bar{F}(w-x)^{d-1} e^{xA} \mu dx & w \leq T \\ \xi_2(w) &= \int_0^T \bar{F}(T-x)^{d-1} e^{(x+w-T)A} dx & w > T \\ \xi_3(w) &= \int_0^{w-T} \bar{F}(w-x) e^{xA} \mu dx. \end{aligned}$$

□

### 4.9.3 DR( $d, T$ )

**Proposition 58.** *The ccdf of the workload distribution for DR( $d, T$ ) without slowdown satisfies the following FDE:*

$$\begin{aligned}\bar{F}'(w) &= -\lambda \left[ \int_0^w f_X(x) (\bar{F}(w-x) - \bar{F}(w)) ((d-1) \right. \\ &\quad \left. (\bar{F}(w-x)^{d-2} \bar{F}(w+T-x)+1) dx \right. \\ &\quad \left. + (1-\bar{F}(w)) \int_0^T f_X(w+x) ((d-1)\bar{F}(T-x)+1) dx \right. \\ &\quad \left. + d\bar{F}_X(w+T)(1-\bar{F}(w)) \right] & w \leq T \\ \bar{F}'(w) &= -\lambda \left[ \int_0^{w-T} (\bar{F}(w-x) - \bar{F}(w))(d-1) \right. \\ &\quad \left. (\bar{F}(w-x)^{d-2} \bar{F}(w+T-x) + \bar{F}(w-T-x)^{d-1}) f_X(x) dx \right. \\ &\quad \left. + \int_{w-T}^w f_X(x) (\bar{F}(w-x) - \bar{F}(w)) \right. \\ &\quad \left. ((d-1)\bar{F}(w-x)^{d-2} \bar{F}(w+T-x)+1) dx \right. \\ &\quad \left. + (1-\bar{F}(w)) \int_0^T f_X(x+w) ((d-1)\bar{F}(w-x)+1) dx \right. \\ &\quad \left. + (1-\bar{F}(w))\bar{F}_X(w+T) \right] & w > T.\end{aligned}$$

*Proof.* This follows from Proposition 46 by simple computation.  $\square$

## 4.10 Numerical method

In this section we discuss the numerical algorithm used to generate the numerical examples for the system stability and workload/response time distribution presented in this chapter. As stated earlier, a comparison with results obtained using time consuming simulation experiments are presented in Section 4.11. Moreover, the numerical analysis we present is a generalization of the analysis presented for Red( $d$ ) with identical replicas in Section 3.6.4. In that section we give a more detailed discussion of the suggested algorithm.

### 4.10.1 Computing the workload distribution

The equilibrium workload distribution can be obtained from a simple forward Euler scheme for future independent policies (Type 1 and Type 3) as the right hand side of these equations only depends on  $\bar{F}(u)$  for  $u \leq w$ . For future dependent policies (Type 2 and Type 4), we observe that the right hand side also depends on  $\bar{F}(u)$  for  $u > w$ , therefore, one may rely on a FPI to obtain the equilibrium workload distribution. However, note that Theorem 40 does not specify a boundary condition for  $\bar{F}(0)$ . This is not surprising as  $\bar{F}(0)$  corresponds to the actual system load which is unknown for some policies. When this load is known, i.e. for Type 3 and Type 4 systems, we can simply use this system load as a boundary condition, i.e. set  $\bar{F}(0) = \lambda(d\delta + \mathbb{E}[\tilde{X}])$  for  $\text{LL}(d, k, \delta)$  and  $\bar{F}(0) = \lambda\mathbb{E}[\tilde{X}]$  for  $\text{JTQ}(d, T)$ . However for other policies (mostly those that contain some type of redundancy),  $\bar{F}(0)$  is unknown.

We do know that if  $\bar{F}$  is the ccdf of a workload distribution, it must satisfy  $\inf_{w>0} \bar{F}(w) = 0$ . Based on this trivial observation, we obtain an algorithm which can be used to find the solution  $\bar{F}$  for (4.4) when the system is stable (i.e. the equilibrium workload distribution is not infinite) and an algorithm to obtain the highest value of  $\lambda$  for which it is still stable. To this end we first define two simple operators,  $T_1 : (0, 1) \rightarrow \mathbb{R}^{[0,\infty)}$  and  $T_2 : \mathbb{R}^{[0,\infty)} \times (0, 1) \rightarrow \mathbb{R}^{[0,\infty)}$ . Here  $T_1$  maps a value  $x_0$  to the solution found by solving the corresponding DIDE with boundary condition  $\bar{F}(0) = x_0$ , using a forward Euler iteration. To define  $T_2$  we first define:

$$R_{x_0}\bar{F} = x_0 - x_0 \frac{\bar{F}(0) - \bar{F}}{\bar{F}(0)},$$

which scales  $\bar{F}$  to satisfy  $\bar{F}(0) = x_0$ . Secondly, we define  $\mathcal{H}_d$  as:

$$\mathcal{H}_d\bar{F}(w) = x_0 - \lambda d \int_0^w \bar{F}'(u) du = x_0 - \lambda d \int_0^w \mathbb{P}\{\mathcal{Q}(U) > u, U \leq u\} du.$$

We now let  $T_2(\bar{F}, x_0)$  denote the operator which first applies  $R_{x_0}$  to  $\bar{F}$  and then repeatedly applies  $\mathcal{H}_d$  until  $\|\bar{F} - \mathcal{H}_d\bar{F}\|_\infty$  is sufficiently small. Using the operators  $T_1$  and  $T_2$ , we propose an algorithm to obtain the equilibrium workload distribution. This algorithm is basically a simple bisection algorithm (on  $T_1$  for future independent and  $T_2$  for future dependent policies), where we look for  $\bar{F}(0)$  such that  $\inf_{w>0} \bar{F}(w) = 0$ .

Step 1: Set  $\text{lb} = 0$ ,  $\text{ub} = 1$ ,  $n = 0$  and  $\bar{F}_0(w) = \bar{F}_{\tilde{X}}(w)$ .

- Step 2: Set  $x_0 = \frac{lb+ub}{2}$  and compute  $\bar{F}_{n+1} = T_1(x_0)$  for a future independent resp.  $\bar{F}_{n+1} = T_2(\bar{F}_n, x_0)$  for a future dependent policy.
- Step 3: Compute  $y = \inf_{w>0} \bar{F}_{n+1}(w)$  and increment  $n$  by one.
- Step 4: Set  $lb = x_0$  if  $y \leq 0$  otherwise set  $ub = x_0$ , return to Step 2.

Terminate the algorithm when  $|\inf_{w>0} \bar{F}(w)|$  is sufficiently small.

**Remark 47.** For the policies where  $\bar{F}(0)$  is known, one can simply set  $lb = ub = \bar{F}(0)$  in step 1 and the equilibrium workload distribution is given by  $\bar{F}_1$ .

It is not hard to see that if  $\bar{F}$  satisfies any of the future (in)dependent FDEs considered in this work and  $\inf_{w>0} \bar{F}(w) = 0$ , then  $\bar{F}$  is indeed a ccdf. To this end one essentially needs to show that  $\bar{F}(w)$  is non-increasing. For example for  $\text{Red}(d, k, \delta)$ , one can establish that  $\bar{F}_{R_{\tilde{X}}}(w) \geq \bar{F}(w)$  for all  $w$  and  $\bar{F}_{R_x}(w) \geq \bar{F}(w)$  for all  $w, x$ . From this it then follows that  $\bar{F}$  is indeed decreasing, the property  $\inf_{w>0} \bar{F}(w) = 0$  then ensures that  $\bar{F}(w) \geq 0$  and  $\lim_{w \rightarrow \infty} \bar{F}(w) = 0$ .

However, to be certain that this algorithm converges to the equilibrium workload distribution, one needs to show that if  $\bar{F}_1$  and  $\bar{F}_2$  are two solutions of the same FDE, that satisfy  $\bar{F}_1(0) \leq \bar{F}_2(0)$  and  $\inf_{w>0} \bar{F}_2(w) < 0$  then also  $\inf_{w>0} \bar{F}_1(w) < 0$ . Proving this seems difficult, but numerical experiments suggest that this is indeed the case for all examples considered. For  $\text{LL}(d, k, \delta)$  with exponential job sizes this trivially holds as the DIDE is equivalent to the ODE (4.41) (this is also the case for  $\text{Red}(d)$  with independent replicas and exponential job sizes). Moreover, convergence in our algorithm is not guaranteed (except to some extent for  $\text{LL}(d, k)$  by Theorem 48).

### 4.10.2 Stability

We let  $\lambda_{\max}$  denote the smallest arrival rate  $\lambda$  for which the load balancing policy is no longer stable, i.e., for all  $\lambda < \lambda_{\max}$  stability is ensured while for  $\lambda \geq \lambda_{\max}$  the system is unstable.

In order to approximate the unknown value of  $\lambda_{\max}$  we need to find the smallest value of  $\lambda$  for which there does not exist any  $\bar{F}(0) \in (0, 1)$  s.t. the associated solution of Theorem 40 satisfies  $\inf_{w>0} \bar{F}(w) = 0$ , equivalently we must find the smallest value of  $\lambda$  s.t. for each choice of  $\bar{F}(0)$  we have  $\inf_{w>0} \bar{F}(w) < 0$ . To this end, we pick some sufficiently small  $\varepsilon > 0$  and set  $x_0 = 1 - \varepsilon$ . We then let  $\bar{F} = T_1(x_0)$  resp.  $\bar{F} = T_2(\bar{F}_{\tilde{X}}, x_0)$ , and check

whether  $\inf_{w>0} \bar{F}(w) < 0$ , if it is, we conclude that the system is (or at least is very close to being) unstable and if  $\inf_{w>0} \bar{F}(w) \geq 0$  we conclude that the system is stable. One can thus find an approximation for  $\lambda_{\max}$  using a simple bisection method.

## 4.11 Validation

### 4.11.1 Workload distribution

In this section we illustrate the accuracy of our numerical method, that is, we numerically obtain the equilibrium workload distribution for different policies from their associated FDEs and compare with results obtained via simulation. All simulation experiments are for  $N = 10, 50$  and  $300$  servers, where we simulate the system up to time  $10^7/N$  with a warm-up period of 30% and start with an empty system. The results are presented in Figure 4.6. The plots indicate that while the accuracy of our approximation is quite poor for  $N = 10$ , it becomes more and more accurate as  $N$  increases and is already very accurate for  $N = 300$  in each of the considered cases. In the remainder of this section we list the parameters settings in each of the 5 examples considered in Figure 4.6.

In Figure 4.6a, we validate the  $\text{Red}(d, k, \delta)$  policy found from Propositions 42 and 53 for the parameters  $d = 3, k = 2, \delta = 0.02$ . The slowdown  $S$  is equal to zero with probability  $1 - q = 0.8$  and exponentially distributed with parameter 1 with probability  $q = 0.2$ .  $X$  follows a geometric distribution with parameter  $1/2$  scaled down such that  $\mathbb{E}[X] = 1$ .

In Figure 4.6b, we consider the  $\text{RTQ}(d, T)$  policy with  $d = 2$  and  $T = 3$ . We choose  $\lambda = 0.75$ , while the slowdown and job size distribution are chosen the same as for  $\text{Red}(d, k, \delta)$  above. The equilibrium workload distribution is obtained using the combination of Propositions 44, 53 and 54.

In Figure 4.6c, we consider the  $\text{DR}(d, T)$  policy with parameters  $d = 2, T = 3, \lambda = 0.7$  and  $S$  and  $X$  taken as for  $\text{Red}(d, k, \delta)$  and  $\text{RTQ}(d, T)$  above. The equilibrium workload distribution is obtained using Proposition 46.

We then consider the  $\text{LL}(d, k, \delta)$  policy with the following parameters. We consider  $d = 3, k = 2, \delta = 0.02$  and  $\lambda = 0.9$ . For this policy we assume that  $\tilde{X} = g(S, X)$  follows an hyperexponential distribution with two phases and balanced means (i.e. the load from the large and small jobs is the

same). Furthermore,  $\mathbb{E}[\tilde{X}] = 1$  and its SCV = 9. The equilibrium workload distribution is obtained using Propositions 48 and 55. The accuracy of our approximation method for large  $N$  is illustrated in Figure 4.6d.

Finally, in Figure 4.6e we consider the  $\text{JTQ}(d, T)$  policy with parameters  $d = 2, T = 3, \lambda = 0.7$  with  $\tilde{X}$  the same as for  $\text{LL}(d, k, \delta)$ . The equilibrium workload distribution is obtained using Proposition 51.

These plots are only a small subset of all numerical validation we have done. We have considered other values for the parameters and other slowdown/job size distributions. However the results are all similar to the ones shown in Figure 4.6.

### 4.11.2 Stability

In this subsection, for the  $\text{DR}(d, T)$  and the  $\text{Red}(d, k, \delta)$  policy, we investigate its stability, i.e., the maximum value of arrival rate  $\lambda_{\max}$  such that the system remains stable for  $\lambda < \lambda_{\max}$ , but is unstable for all  $\lambda \geq \lambda_{\max}$ . For the  $\text{Red}(d, k, \delta)$  policy, we consider a system with  $N = 300, d = 2, k = 1, \delta = 0.01$  and  $q = 0.2$ . As earlier, we assume that  $S$  is exponential with probability  $q$  and zero with probability  $1 - q$  and  $X$  is a scaled geometric random variable with parameter  $1/2$  and mean 1. We obtain  $\lambda_{\max}$  using the algorithm presented in Section 4.10.2. We find that for this set of parameters, we have  $\lambda_{\max} = 0.7145$ . To verify this, we consider a simulation of the system with  $N = 300$  and values of  $\lambda$  just above resp. below  $\lambda_{\max}$ . We simulate the system for a time span equal to  $3 \cdot 10^4 \approx 10^7/N$  and start with an empty system. In Figure 4.7a, we observe that while the mean workload for  $\lambda = \lambda_{\max} - 0.001$  seems to converge, it appears to diverge for  $\lambda = \lambda_{\max} + 0.001$ .

A similar experiment was performed for the  $\text{DR}(d, T)$  policy, where we consider  $d = 2, T = 3, q = 0.2$  and again the same slowdown and job size distribution. For this setting of the parameters, we observe that  $\lambda_{\max} = 0.7571$ . Figure 4.7b seems to indicate that our approach to find  $\lambda_{\max}$  is indeed quite accurate even for finite  $N$ .

## 4.12 Future work

This chapter provides a numerical method to practitioners to assess the performance of workload dependent policies without the need to resort to simulation. In some rare cases, analytical results have been found by solving the

FDE obtained in this work (see Chapter 2 and [39]). A theoretical follow-up may exist in proving the asymptotic independence for specific policies. Another alley worth investigating is letting  $d$  scale with  $N$ . A disadvantage of this is that many of the policies become unstable for all  $\lambda > 0$  if  $d_N \rightarrow \infty$ . It would also be of interest to generalize these results to the setting of heterogeneous servers, in this case one would need to take a queue at the cavity for each server type. Another interesting application of this method is the case with energy aware servers, where servers shut down when idle and take some time  $\delta > 0$  to restart when a job arrives.

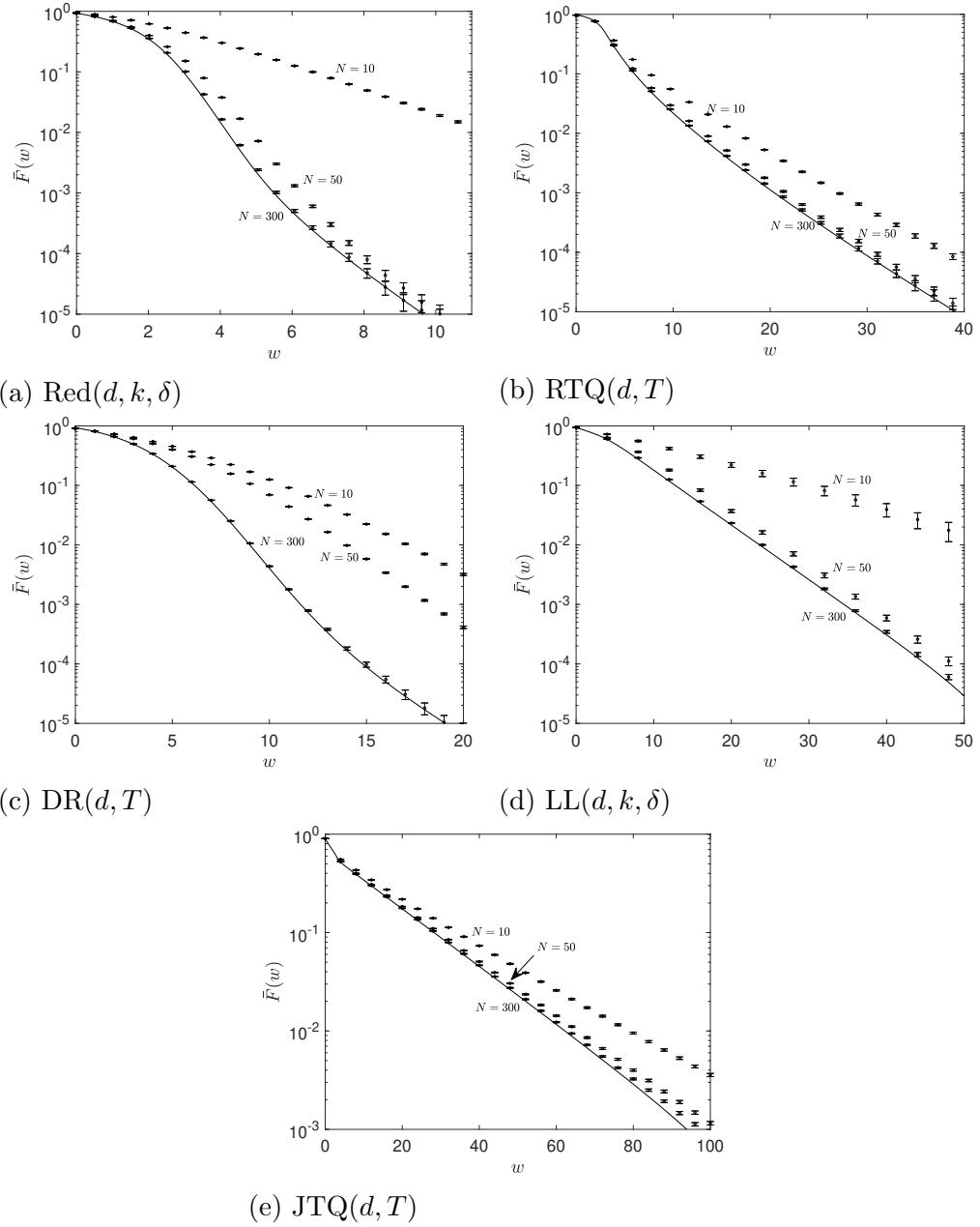
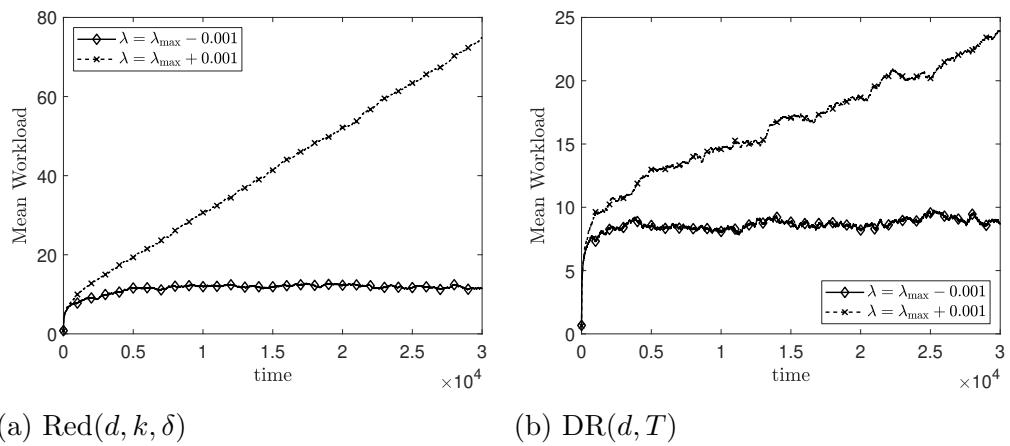


Figure 4.6: Limiting workload distribution vs. simulation for the  $N$  server system with  $\text{Red}(3, 2, 0.02)$ ,  $\text{RTQ}(2, 3)$ ,  $\text{DR}(2, 3)$ ,  $\text{LL}(3, 2, 0.02)$  and  $\text{JTQ}(2, 3)$  policies respectively under different settings of arrival rate  $\lambda$ , service requirement  $X$  and the slowdown variable  $S$ .



(a)  $\text{Red}(d, k, \delta)$

(b)  $\text{DR}(d, T)$

Figure 4.7: Simulations for  $\text{Red}(d, k, \delta)$  and  $\text{DR}(d, T)$  around their respective values of  $\lambda_{\max}$ .

# Chapter 5

## Mean waiting time in large-scale and critically loaded power of $d$ load balancing systems

You load sixteen tons, what do  
you get? Another day older and  
deeper in debt.

---

Tennessee Ernie Ford

### Abstract

Mean field models are a popular tool used to analyse load balancing policies. In some exceptional cases the waiting time distribution of the mean field limit has an explicit form. In other cases it can sometimes be computed as the solution of a set of differential equations or through a recurrence relation. In this chapter we study the limit of the mean waiting time  $\mathbb{E}[W_\lambda]$  as the arrival rate  $\lambda$  approaches 1 for a number of load balancing policies when job sizes are exponential with mean 1 (i.e. when the system gets close to instability). As  $\mathbb{E}[W_\lambda]$  diverges to infinity, we scale with  $-\log(1 - \lambda)$  and present a method to compute the limit  $\lim_{\lambda \rightarrow 1^-} -\mathbb{E}[W_\lambda]/\log(1 - \lambda)$ . We show that this limit has a surprisingly simple form for the load balancing algorithms considered.

More specifically, we present two general results, one holds for policies whose stationary distribution is given as the solution of an ODE while the other result holds for policies whose stationary distribution is given as the solution of a recurrence relation. Any policy for which the associated differential equation/recurrence relation satisfies a list of criteria we find a closed form expression for the above limit.

The distinction we makes corresponds to the distinction between workload and queue length dependent load balancing policies. For workload dependent policies we generally find that the stationary distribution satisfies an ODE and the limiting value is given by  $\frac{B}{A-1}$  for some  $B > 0$  and  $A > 1$ . For the queue length dependent variants we find that the limit is given by  $\frac{B}{\log(A)}$ .

For the well-known LL( $d$ ) resp. SQ( $d$ ) policies which assigns an incoming job to a server with the least work left resp. shortest queue among  $d$  randomly selected servers these criteria are trivially verified. For these policies we prove the limit is given by  $\frac{1}{d-1}$  resp.  $\frac{1}{\log(d)}$ . We further show that the LL( $d, K$ ) resp. SQ( $d, K$ ) policies, which assign batches of  $K$  jobs to the  $K$  least loaded resp. shortest queues among  $d$  randomly selected servers, satisfies the criteria and the limit is equal to  $\frac{K}{d-K}$  resp.  $\frac{1}{\log(1-\frac{d}{K})}$ . For a policy which applies LL( $d_i$ ) resp. SQ( $d_i$ ) with probability  $p_i$ , we show that the limit is given by  $\frac{1}{\sum_i p_i d_i - 1}$  resp.  $\frac{1}{\log(\sum_i p_i d_i)}$ . We further indicate that our main result can also be used for load balancers with redundancy or memory.

In addition, we propose an alternate scaling  $-\log(p_\lambda)$  instead of  $-\log(1-\lambda)$ , where  $p_\lambda$  is adapted to the policy at hand, such that  $\lim_{\lambda \rightarrow 1^-} -\mathbb{E}[W_\lambda]/\log(1-\lambda) = \lim_{\lambda \rightarrow 1^-} -\mathbb{E}[W_\lambda]/\log(p_\lambda)$ , where the limit  $\lim_{\lambda \rightarrow 0^+} -\mathbb{E}[W_\lambda]/\log(p_\lambda)$  is well defined and non-zero (contrary to  $\lim_{\lambda \rightarrow 0^+} -\mathbb{E}[W_\lambda]/\log(1-\lambda)$ ). This allows to obtain relatively flat curves for  $-\mathbb{E}[W_\lambda]/\log(p_\lambda)$  for  $\lambda \in [0, 1]$  which indicates that the low and high load limits can be used as an approximation when  $\lambda$  is close to one or zero.

Our results regarding workload dependen load balancing policies rely on the earlier proven ansatz which asserts that for certain load balancing policies the workload distribution of any finite set of queues becomes independent of one another as the number of servers tends to infinity.

## 5.1 Introduction

Load balancing plays an important role in large scale data networks, server farms, cloud and grid computing. From a mathematical point of view, the load balancing policies we consider in this thesis can be split into two main categories. The first category exists of queue length dependent load balancing policies where the dispatcher collects some information on the number of jobs in some servers and assigns an incoming job using this information. A well studied example of this policy type is the  $SQ(d)$  policy, where an incoming job is assigned to the shortest among  $d$  randomly selected servers (see Section 1.5 and [66, 95]). The second category consists of workload dependent load balancing policies, for these policies the dispatcher balances the load on the servers by employing information on the amount of work that is left on some of the servers (see also Chapter 4). This can be done explicitly if we assume the amount of work on servers is known or implicitly by employing some form of redundancy such as e.g. cancellation on start or late binding (see also [73]). A well studied policy of this type is the  $LL(d)$  policy, where each incoming job joins the server with the least amount of work left out of  $d$  randomly sampled servers (see e.g. Chapter 2).

In order to compute performance metrics such as the mean waiting time, the waiting time distribution, etc. most work relies on mean-field models [54, 83, 46, 48, 20]. Mean field models capture the limiting stationary behavior of the system as the number of servers tends to infinity provided that any finite set of servers becomes independent and identically distributed. Recently this independence was proven for a wide variety of workload dependent load balancing policies in [83]. All but one of the workload dependent policies studied in this work fit into the framework of [83]. The limiting stationary workload can therefore be described by the stationary workload distribution of a single server/queue. In order to analyse this queue, termed *the queue at the cavity*, the stationary workload distribution is characterized by an DIDE, which can sometimes be simplified to a one dimensional ODE in case job sizes are exponential. Throughout this chapter, we assume the job size distribution is exponential with mean one. However, in Section 8.4, we attempt to generalize these results to Erlang- $k$  and Deterministic distributions.

We relate to each system size  $N$  an arrival rate  $\lambda_N$ . To obtain the mean field limit as described earlier, one sets  $\lambda_N = \lambda N$  for some fixed  $\lambda < 1$ . One is often interested in the behaviour of the queueing system as the system

approaches its critical load. To study this, one could set  $\lambda_N = \lambda(N)N$  where  $\lambda(N) \rightarrow 1^-$  as  $N$  tends to infinity. This approach was for example used in [58, 57, 25, 30] to study the SQ( $d$ ) model in heavy traffic. Another approach, which is the one we use here, is to first obtain the stationary distribution of the mean field model with a fixed  $\lambda(N) = \lambda < 1$  and subsequently take the limit  $\lambda \rightarrow 1^-$  of the resulting mean field models. For workload dependent policies, we are not aware of any work where the approach of letting  $\lambda(N) \rightarrow 1^-$  has been considered. However, for the Queue Length dependent load balancing policies, the comparison between our approach and the approach of letting  $\lambda(N) \rightarrow 1^-$  is worth investigating further. For the SQ( $d$ ) policy, it is shown in [66] that  $\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = \frac{1}{\log(d)}$ , with  $W_\lambda$  the waiting time distribution for the SQ( $d$ ) policy with arrival rate  $\lambda$ . However, its proof is a technical computation which relies heavily on the closed form solution of the stationary distribution and does not seem to generalize well.

In this chapter we establish a general result which can be employed to obtain the limit:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = \frac{B}{\log(A)}, \quad (5.1)$$

where  $W_\lambda$  is the waiting time distribution of a workload dependent load balancing policy and  $A$  and  $B$  are values which can be easily computed (see Theorem 60 and Corollary 61). For the queue length dependent variants of the policies we consider, we find that under the (more or less) same assumptions the limiting value is given by:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = \frac{B}{A-1}, \quad (5.2)$$

where the same values for  $A$  and  $B$  are used as in (5.2).

This value can be used as a reference to indicate how well a policy behaves under a high load. As we divide by  $-\log(1-\lambda)$ , we are focussing on load balancing policies where an exponential improvement in the mean waiting time is expected compared to random assignment. For LL( $d$ ) it is indirectly claimed in Theorem 20 that the limit (5.2) is given by  $\frac{1}{d-1}$ , though the proof is incorrect (c.f. the remark after Corollary 62). This formula follows from our result as we find that  $B = 1$  and  $A = d$  for the LL( $d$ ) policy. Moreover, this also recovers the result from [66] that for the SQ( $d$ ) policy the limit in (5.1) is given by  $\frac{1}{\log(d)}$ .

Our result provides a list of sufficient assumptions under which the limit in (5.2) can be computed in a straightforward manner. Although computing the

limit is easy, verifying the listed assumptions may present quite a challenge, one of our main contributions is establishing these assumptions for  $\text{LL}(d, K)$  &  $\text{SQ}(d, K)$ .

We start by applying our method on  $\text{LL}(d)$  and  $\text{SQ}(d)$  providing a first proof for the associated limit for  $\text{LL}(d)$  and confirming the limiting result in [66].

We then apply our method to the  $\text{LL}(d, K)$  resp.  $\text{SQ}(d, K)$  policies (see also [90, 100]). For this policy, jobs are assumed to arrive in batches of size  $K$ , we then sample  $d > K$  servers and the jobs are assigned to the  $K$  queues with the least amount of work resp. least number of jobs left. We show in Section 5.5 that:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = \frac{1}{\frac{d}{K} - 1} = \frac{K}{d-K}. \quad (5.3)$$

for  $\text{LL}(d, K)$  while for  $\text{SQ}(d, K)$  we find that the limit is given by:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = \frac{1}{\log\left(\frac{d}{K}\right)}. \quad (5.4)$$

One of the main technical contributions of the chapter, apart from establishing Theorem 60, exists in verifying the third assumption of this theorem for  $\text{LL}(d, K)/\text{SQ}(d, K)$ .

Next, we consider the  $\text{LL}(d_1, \dots, d_n, p_1, \dots, p_n)/\text{SQ}(d_1, \dots, d_n, p_1, \dots, p_n)$  policies, where with probability  $p_i$  we select  $d_i$  servers and assign the incoming job to the queue with the least amount of work amongst these  $d_i$  selected servers. We show that for  $\text{LL}(d_1, \dots, d_n, p_1, \dots, p_n)$  we have:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = \frac{1}{\sum_{i=1}^n p_i d_i - 1},$$

while for  $\text{SQ}(d_1, \dots, d_n, p_1, \dots, p_n)$  we find:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = \frac{1}{\log\left(\sum_{i=1}^n p_i d_i\right)},$$

We observe that, when the system is highly loaded, the choice of  $p_i$  and  $d_i$  does not matter as long as the total amount of redundancy  $\sum_{i=1}^n p_i d_i$  remains constant. Furthermore we find a general method to investigate which choice of  $p_i$  and  $d_i$  yields smaller response times when  $\lambda < 1$ .

In the special case of  $\text{LL}(1, d, 1 - p, p)$ , this policy applies the power of  $d$  choices only to a proportion of the incoming jobs and assigns the other jobs arbitrarily. For this policy, we find that whenever  $\lambda < 1$  the limiting probability that an arbitrary queue has workload at least  $w$  is given by:

$$\bar{F}(w) = \lambda \left[ \frac{1 - (1 - p)\lambda}{p\lambda^d + (1 - (1 - p)\lambda - p\lambda^d)e^{(d-1)(1-(1-p)\lambda)w}} \right]^{\frac{1}{d-1}}, \quad (5.5)$$

but no such solution appears to exist in general. This closed form expression yields an alternative method to obtain the limiting result. Equivalently this model may be described as having an individual arrival process with rate  $(1 - p)\lambda$  at each server in addition to an  $\text{LL}(d)$  arrival stream with rate  $p\lambda N$ . This type of model was for example studied in [26].

We also argue that our result can be directly used for  $\text{Red}(d)$  with i.i.d. replicas. Furthermore, we use our general result to compute the same limit for load balancing policies with memory at the dispatcher (see Chapter 6 and in particular Section 6.10).

To obtain these results, the main insight we use is the fact that, as  $\lambda$  approaches one, all queues have more or less the same amount of work (see also Figures 5.1 and 5.2). We are able to analytically approximate this amount of work, it represents how well a policy is able to balance loads under a high arrival rate. A similar observation was made in [9], where it was noted that for Redundancy  $d$  under Processor Sharing with identical replicas, the workload at all servers diverges to infinity at an equal rate when  $\lambda$  exceeds  $\frac{1}{d}$ .

While  $\lim_{\lambda \rightarrow 1^-} -\mathbb{E}[W_\lambda]/\log(1 - \lambda)$  is finite and non-zero, the scaling with  $-\log(1 - \lambda)$  is not very insightful when  $\lambda$  is small as  $\lim_{\lambda \rightarrow 0^+} -\mathbb{E}[W_\lambda]/\log(1 - \lambda)$  tends to be zero. We therefore additionally introduce an alternate scaling  $-\log(p_\lambda)$ , where the value of  $p_\lambda$  is policy specific and discussed in Section 5.7, such that the limit for  $\lambda$  tending to one remains the same, while  $\lim_{\lambda \rightarrow 0^+} -\mathbb{E}[W_\lambda]/\log(p_\lambda)$  is well defined and non-zero. It turns out that for this scaling the curve  $-\mathbb{E}[W_\lambda]/\log(p_\lambda)$  is fairly flat for  $\lambda \in [0, 1]$  meaning that the low and high load limits of the alternate scaling can be regarded as a good approximation for low and high loads.

## 5.2 Structure of this chapter

This chapter is structured as follows. In Section 5.3, we give the reader an intuitive simplification of our main proof applied to the  $\text{SQ}(d)$  policy. In

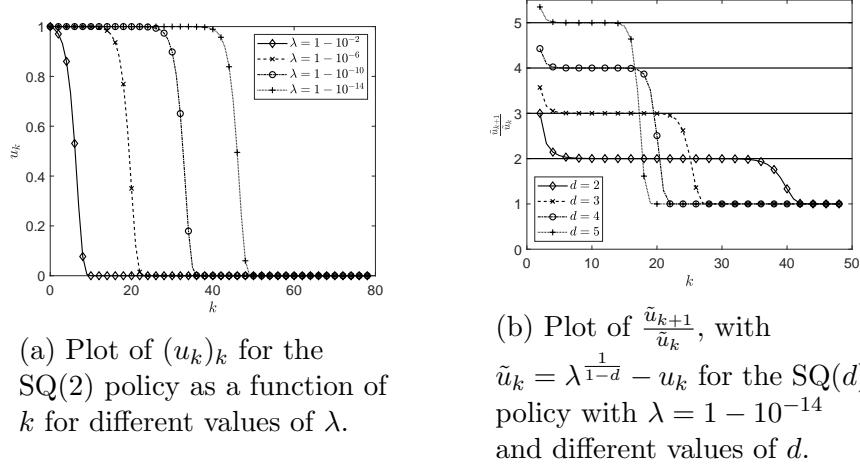


Figure 5.1: Graphical support for the proof of Theorem 59 applied to the SQ( $d$ ) policy.

Sections 5.4.1 and 5.4.2 we illustrate the type of policies for which our result is applicable. In Section 5.4.3, we present the main result and indicate that it is applicable to the LL( $d$ ), SQ( $d$ ) and Red( $d$ ) policies. In Section 5.4.4 we compute the limiting value as  $\lambda \rightarrow 1^-$  for all considered policies and in Sections 5.4.6 and 5.4.5 we give the proofs of the main results. In Section 5.5 we verify the assumptions for LL( $d, K$ ) and SQ( $d, K$ ). In Section 5.6 we cover LL( $d_1, \dots, d_n, p_1, \dots, p_n$ ) and SQ( $d_1, \dots, d_n, p_1, \dots, p_n$ ), here we also consider the case where  $\lambda$  is bounded away from 1 and the special case of LL( $1, d, 1-p, p$ ). We introduce and discuss the alternate scaling by  $-\log(p_\lambda)$  in Section 5.7. We provide a selection of numerical experiments in Section 5.8. Conclusions are drawn and extensions are suggested in Section 5.9.

### 5.3 Informal intuition

The main objective of this chapter is to prove a general result which can be used to compute the limit:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1 - \lambda)}.$$

To this end we show two proofs, one for queue length dependent and one for workload dependent load balancing policies. Both these proofs are quite

technical with many abstract variables. In this section, we specify the proof for the SQ( $d$ ) policy for which we can compute these variables in closed form. This allows for a better understanding of the general proof.

Assume we have a recurrence relation of the form

$$u_{k+1} = T_\lambda(u_k),$$

where  $T_\lambda$  is some positive function. In the special case of the SQ( $d$ ) policy, the recurrence relation in [65] can be rewritten as  $u_{k+1} = \lambda u_k^d$  for  $k > 0$  and  $u_0 = 1$ , yielding the well-known result:

$$u_k = \lambda^{(d^k - 1)/(d-1)}.$$

In Figure 5.1a we observe that, as we increase  $\lambda$ , the value of  $u_k$  remains close to one for larger values of  $k$ , but the shape of the curve when it drops to zero, looks very similar for the different values of  $\lambda$ . This motivates us to define  $N_{\varepsilon,\lambda}$ , which represents the point at which  $u_k$  drops below some threshold close to one. One would then expect that:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = \lim_{\lambda \rightarrow 1^-} -\frac{\sum_{k=0}^{\infty} u_k}{\log(1-\lambda)} = \lim_{\lambda \rightarrow 1^-} -\frac{N_{\varepsilon,\lambda}}{\log(1-\lambda)}$$

the first equality follows from Little's law while the second should follow from the fact that  $u_k \approx 1$  for  $k \leq N_{\varepsilon,\lambda}$  and the sum of the remaining  $u_k$  values remains bounded. More specifically, we define the threshold as  $u_\lambda - \varepsilon$ , where  $u_\lambda$  is a solution of  $u = T_\lambda(u)$  such that  $u_\lambda$  decreases to 1 as  $\lambda$  increases to one. For SQ( $d$ ) we set  $u_\lambda = \lambda^{1/(1-d)}$  and one easily verifies that (with  $\lceil \cdot \rceil$  the ceil function):

$$N_{\varepsilon,\lambda} = \left\lceil \frac{1}{\log(d)} \cdot \log \left( 1 - \frac{\log(\lambda^{1/(1-d)} - \varepsilon)}{\log(\lambda^{1/(1-d)})} \right) \right\rceil,$$

from which it follows that  $\lim_{\lambda \rightarrow 1^-} -\frac{N_{\varepsilon,\lambda}}{\log(1-\lambda)} = \frac{1}{\log(d)}$ , as expected.

In order to compute  $\lim_{\lambda \rightarrow 1^-} -\frac{N_{\varepsilon,\lambda}}{\log(1-\lambda)}$  in case we do not have an explicit expression for  $N_{\varepsilon,\lambda}$ , we define the sequence  $\tilde{u}_k = u_\lambda - u_k$ . Note that  $N_{\varepsilon,\lambda}$  is the largest value of  $k$  for which  $\tilde{u}_k$  remains below  $\varepsilon$ . In Figure 5.1b, we plotted  $\frac{\tilde{u}_{k+1}}{\tilde{u}_k}$  as a function of  $k$  for SQ( $d$ ). We observe that  $\frac{\tilde{u}_{k+1}}{\tilde{u}_k} \approx d$  (represented by the horizontal lines) for  $k$  bounded away from 0 and  $k \leq N_{\varepsilon,\lambda}$ . This in its turn entails that:

$$\varepsilon \approx \tilde{u}_{N_{\varepsilon,\lambda}} = \frac{\tilde{u}_{N_{\varepsilon,\lambda}}}{\tilde{u}_{N_{\varepsilon,\lambda}-1}} \cdots \frac{\tilde{u}_1}{\tilde{u}_0} \tilde{u}_0 \approx d^{N_{\varepsilon,\lambda}} \cdot (\lambda^{1/(1-d)} - 1).$$

Taking the log on both sides, dividing by  $-\log(1 - \lambda)$  and taking the limit of  $\lambda \rightarrow 1^-$ , allows us to recover that  $\lim_{\lambda \rightarrow 1^-} -\frac{N_{\varepsilon,\lambda}}{\log(1-\lambda)} \approx \frac{1}{\log(d)}$ , as

$$\lim_{\lambda \rightarrow 1^-} \frac{\log(\lambda^{1/(1-d)} - 1)}{\log(1 - \lambda)} = 1.$$

To establish Theorem 59 we also rely on the sequence  $\tilde{u}_k$  and introduce upper and lower bounds on  $\tilde{u}_{k+1}/\tilde{u}_k$  which converge to the same value as  $\lambda \rightarrow 1^-$  and  $\varepsilon \rightarrow 0^+$ . This allows us to derive an expression for

$$\lim_{\lambda \rightarrow 1^-} -\frac{W_\lambda}{\log(1 - \lambda)} = \lim_{\varepsilon \rightarrow 0^+} \lim_{\lambda \rightarrow 1^-} -\frac{N_{\varepsilon,\lambda}}{\log(1 - \lambda)}.$$

## 5.4 General result

### 5.4.1 The ordinary differential equation

One of our main results results can be applied to functions which are the solution of the ODE:

$$\bar{F}'(w) = T_\lambda(\bar{F}(w)) - \bar{F}(w), \quad (5.6)$$

with some boundary condition  $\bar{F}(0) = x$  with  $x \in [\lambda, 1]$ . We showed in Theorem 12 that the ccdf of the limiting stationary workload distribution for the LL( $d$ ) policy can be found as the solution of (5.6) with boundary condition  $\bar{F}(0) = \lambda$  and  $T_\lambda(u) = \lambda u^d$ . For the Red( $d$ ) policy with i.i.d. replicas it was proven in [40] that the ccdf of the response time distribution  $\bar{F}_R(w)$  satisfies the same ODE, that is  $\bar{F}_R(w)$  satisfies (5.6) with  $T_\lambda(u) = \lambda u^d$ , but with boundary condition  $\bar{F}_R(0) = 1$ .

For LL( $d, K$ ) (c.f. Section 5.6 we find that the ccdf of the workload distribution satisfies (5.6) with:

$$T_\lambda(u) = \frac{\lambda}{K} \sum_{j=0}^{K-1} (K-j) \binom{d}{j} u^{d-j} (1-u)^j, \quad (5.7)$$

and  $\bar{F}(0) = \lambda$ . For the LL( $d_1, \dots, d_n, p_1, \dots, p_n$ ) policy (c.f. Section 5.6.1) we find that the ccdf of the workload distribution is given by the solution of (5.6) with:

$$T_\lambda(u) = \sum_i p_i u^{d_i}, \quad (5.8)$$

and  $\bar{F}(0) = \lambda$ . For the memory dependent  $\text{LL}(d)$  policy, we assume that each arrival has a probability  $\pi_0(\lambda)$  to be routed using the  $\text{LL}(d)$  policy, while with the remaining probability  $1 - \pi_0(\lambda)$  it is routed to an empty queue. For this policy we show in Chapter 6 that the ccdf of the workload distribution  $\bar{F}(w)$  satisfies (5.6) with:

$$T_\lambda(u) = \lambda\pi_0(\lambda)u^d. \quad (5.9)$$

The fact that the solution to (5.6) captures the limit of the stationary workload distribution of a single queue, or the stationary response time distribution of a job, as the number of queues  $N$  tends to infinity for the policies under consideration, is due to the recent results found in [83] (except for the memory dependent case, (5.9)). In [83], the authors prove the independence ansatz introduced in [23] for a variety of workload dependent load balancing policies, their approach is based on the following three properties:

- (a) Monotonicity, which essentially states that as we increase the number of probes used per arrival, the delay a job experiences also reduces.
- (b) Work conservation, that is, executed work is never lost.
- (c) The property that, on average, *new arriving workload prefers to go to servers with lower workloads*.

More specifically, in [83] the authors prove the independence ansatz for any convex combination of  $\text{LL}(d, K)$  (with arbitrary job sizes) and  $\text{Red}(d, K)$  (with exponential job sizes and i.i.d. replicas). For the memory dependent load balancing policies proving the ansatz remains an open problem as one is faced with the additional problem of a time-scale separation as the memory content evolves on a different time-scale than the workload. For more details we refer the reader to Chapter 6 and [16].

### 5.4.2 The recurrence relation

In Section 5.4.1, we showed that for a variety of workload dependant load balancing policies, the ccdf of the equilibrium workload distribution can be written as the solution of the simple ODE (5.6). For queue length dependent load balancing policies, one denotes by  $u_k$  the probability that the cavity queue has  $k$  or more jobs in its queue. One finds that for the queue length

dependent variants of the policies which were introduced in Section 5.4.1 these  $u_k$  satisfy the simple recurrence relation:

$$u_{k+1} = T_\lambda(u_k), \quad (5.10)$$

where  $T_\lambda$  is the same function as in (5.6).

Indeed we have the following results:

- In [66] it is shown that for  $\text{SQ}(d)$ ,  $u_k$  satisfies (5.10) with  $T_\lambda(u) = u^d$  as for  $\text{LL}(d)$ .
- In [90], the authors prove that for  $\text{SQ}(d, K)$   $u_k$  is given as the solution to (5.10) with  $T_\lambda(u)$  defined as in (5.7).
- The fact that  $\text{SQ}(d_1, \dots, d_n, p_1, \dots, p_n)$  satisfies (5.10) with  $T_\lambda(u)$  as in (5.8) is not hard to show (the method is similar to the one used for  $\text{SQ}(d)$ ).
- The memory dependent  $\text{SQ}(d)$  policy is studied in detail in Chapter 6. In particular we show that  $u_k$  indeed satisfies (5.6) with  $T_\lambda$  defined as in (5.9).

### 5.4.3 Statement & application of the main result

We first introduce all assumptions which we require in order to obtain the limiting value of  $-\mathbb{E}[W_\lambda]/\log(1-\lambda)$  as  $\lambda \rightarrow 1^-$ . We illustrate the assumptions by showing that they hold for the  $\text{LL}(d)$ ,  $\text{SQ}(d)$  and  $\text{Red}(d)$  policies, that is for the choice  $T_\lambda(u) = \lambda u^d$ .

**Assumption 1.** *There exists a  $\bar{\lambda} \in (0, 1)$  such that:*

- *For  $\lambda \in (\bar{\lambda}, 1)$  there exists a  $u \in (1, \infty) : T_\lambda(u) = u$ . We define  $u_\lambda \in (1, \infty)$  as the minimal value for which  $T_\lambda(u_\lambda) = u_\lambda$ .*
- *The function  $u_\cdot : \lambda \rightarrow u_\lambda$  is continuous and  $\lim_{\lambda \rightarrow 1^-} u_\lambda = 1$ .*

For  $T_\lambda(u) = \lambda u^d$ , we can set  $\bar{\lambda} = 0$  and finding  $u_\lambda$  reduces to obtaining the smallest solution of  $u = \lambda u^d$  in  $(1, \infty)$ . We quickly find that  $u_\lambda = \lambda^{\frac{1}{1-d}}$ , which is obviously continuous in  $\lambda$  and converges to one as  $\lambda$  approaches 1.

**Assumption 2.** *For all  $u \in (0, 1]$ , we have:*

- $T_\lambda(0) = 0$ ,  $T_\lambda(u) < u$  and  $\lim_{\lambda \rightarrow 1^-} \frac{T_\lambda(u)}{u} < 1$ ,
- $\left(\frac{T_\lambda(u)}{u}\right)' \geq 0$ , which implies that  $T_\lambda$  is increasing on  $(0, 1)$ .

We have  $\frac{T_\lambda(u)}{u} = \lambda u^{d-1}$  from which this assumption trivially follows.

**Assumption 3.** For all  $\lambda \in (\bar{\lambda}, 1)$  we define:

$$h_\lambda(x) = \frac{u_\lambda - T_\lambda(u_\lambda - x)}{x}. \quad (5.11)$$

There is some  $b \in \mathbb{N}$  such that for all  $\lambda \in (\bar{\lambda}, 1)$  we have  $h_\lambda(x)$  is decreasing for  $x \in [u_\lambda - \lambda^b, 1]$ .

For  $T_\lambda(u) = \lambda u^d$ , we find (with  $h_\lambda(x)$  defined as in (5.11)):

$$h_\lambda(x) = \frac{\lambda^{\frac{1}{1-d}} - \lambda(\lambda^{\frac{1}{1-d}} - x)^d}{x}, \quad (5.12)$$

its derivative is given by:

$$h'_\lambda(x) = \frac{\lambda \left(\lambda^{\frac{1}{1-d}} - x\right)^{d-1} \left(\lambda^{\frac{1}{1-d}} + (d-1)x\right) - \lambda^{\frac{1}{1-d}}}{x^2}$$

differentiating  $x^2 h'_\lambda(x)$  once more yields:

$$(x^2 h'_\lambda(x))' = -\lambda(d-1)d \left(\lambda^{\frac{1}{1-d}} - x\right)^{d-2} x,$$

which is obviously negative for  $x \in [0, 1)$ . Hence, this assumption now follows with  $b = 0$  from the fact that  $(x^2 h'_\lambda(x))'$  equals 0 for  $x = 0$ . The next assumptions differs slightly for our two main results. For the result regarding the solution of an ODE (Theorem 60) we find:

**Assumption 4.** For any  $\lambda \in (\bar{\lambda}, 1)$  we let  $\bar{w}_\lambda \in [0, \infty)$  be the smallest value for which  $\bar{F}(\bar{w}_\lambda) \leq \lambda^b$ . There is some  $\bar{w}$  which can be chosen independently of  $\lambda$  such that  $\bar{w}_\lambda \leq \bar{w}$ .

For the result regarding the solution of a recurrence relation (Theorem 59) we find:

**Assumption 4'.** For any  $\lambda \in (\bar{\lambda}, 1)$  we let  $\bar{k}_\lambda \in \mathbb{N}$  be the smallest value for

which  $u_{\bar{k}_\lambda} \leq \lambda^b$ . There is some  $\bar{k}$  which can be chosen independently of  $\lambda$  such that  $\bar{k}_\lambda \leq \bar{k}$ .

As we showed assumption 3 with  $b = 0$ , we find that  $\bar{w}_\lambda = 0 = \bar{k}_\lambda$  for all  $\lambda \in [0, 1)$  from which these assumptions trivially follow with  $\bar{w} = 0 = \bar{k}$ .

**Remark 48.** For assumption 4 it suffices in general to show that  $\bar{F}(w) \leq \lambda e^{-(1-\lambda)w}$ . Indeed, to have  $\lambda e^{-(1-\lambda)w} \leq \lambda^b$  it suffices to have  $b - 1 \leq \bar{w}_\lambda$ . Therefore one may pick  $\bar{w} = b - 1$ . Note that  $\lambda e^{-(1-\lambda)w}$  is the probability that the workload of an M/M/1 queue is at least  $w$ , therefore it suffices that the policy is at least as good as random routing. One can make a similar remark for assumption 4'.

**Assumption 5.** There is some  $A \in (1, \infty)$  for which  $\lim_{\lambda \rightarrow 1^-} h_\lambda(u_\lambda - \lambda^b) = A$ .

For  $h_\lambda(x)$  given by (5.12) we note that:

$$h_\lambda(u_\lambda - 1) = \frac{\lambda^{\frac{1}{1-d}} - \lambda}{\lambda^{\frac{1}{1-d}} - 1} \xrightarrow[\lambda \rightarrow 1^-]{} d,$$

where the limit statement can be shown using l'Hopital's rule. Therefore this assumption holds for LL( $d$ ), SQ( $d$ ) and Red( $d$ ) with  $A = d$ .

**Assumption 6.** There is some  $B \in [0, \infty)$  for which  $\lim_{\lambda \rightarrow 1^-} \frac{\log(u_\lambda - \lambda^b)}{\log(1-\lambda)} = B$ .

Using  $u_\lambda = \lambda^{\frac{1}{1-d}}$  and  $b = 0$ , we find that  $B = 1$ , when  $T_\lambda(u) = \lambda u^d$ , by a simple application of l'Hopital's rule.

**Assumption 7.** We have  $\lim_{\varepsilon \rightarrow 0^+} \lim_{\lambda \rightarrow 1^-} h_\lambda(\varepsilon) = A$ .

We note that for  $T_\lambda(u) = \lambda u^d$ :

$$\lim_{\lambda \rightarrow 1^-} h_\lambda(\varepsilon) = \frac{1 - (1 - \varepsilon)^d}{\varepsilon} \xrightarrow[\varepsilon \rightarrow 0^+]{} d,$$

from which assumption 7 follows. We are now in a position to state our general result. We first state it for queue length dependent policies:

**Theorem 59.** For  $\lambda \in (0, 1)$ , let  $(u_k)_k$  denote the unique solution to the recurrence relation (5.10) with  $u_0 = 1$ . If assumptions 1-7 hold (with assumption 4 replaced by assumption 4'), it then follows that :

$$\lim_{\lambda \rightarrow 1^-} -\frac{\sum_{k=0}^{\infty} u_k}{\log(1 - \lambda)} = \frac{B}{\log(A)}. \quad (5.13)$$

**Remark 49.** Applying Little's law allows one to simply replace  $\sum_{k=0}^{\infty} u_k$  by  $\mathbb{E}[Q_\lambda]$ ,  $\mathbb{E}[R_\lambda]$  and  $\mathbb{E}[W_\lambda]$  when we apply the limiting result of Theorem 59 to queue length dependent load balancing policies.

We now consider the case of workload dependent load balancing policies.

**Theorem 60.** For any  $\lambda \in (0, 1)$  we let  $\bar{F} : [0, \infty) \rightarrow [0, 1]$  be a solution to (5.6) with  $\bar{F}(0) = x$  for some fixed  $x \in [\lambda, 1]$ , where we assume  $\bar{F}$  is the unique continuously differentiable solution to this ODE. Further, we assume that  $T_\lambda$  satisfies assumptions 1 - 7. We then have:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\int_0^\infty \bar{F}(w) dw}{\log(1 - \lambda)} = \frac{B}{A - 1}. \quad (5.14)$$

For most of our applications,  $\int_0^\infty \bar{F}(w) dw$  is equal to the expected workload. The next Corollary shows that the mean queue length is in fact equal to the mean workload for some of the policies considered in this chapter, allowing us to obtain the mean waiting time from the mean workload using Little's law.

**Corollary 61.** Consider  $LL(d)$  (with or without memory),  $LL(d, K)$  or  $LL(d_1, \dots, d_n, p_1, \dots, p_n)$  with job sizes that are exponential with mean one and assume the ccdf of the workload distribution  $\bar{F}(w)$  satisfies the requirements outlined in Theorem 60, then the mean queue length is equal to the mean workload. In particular:

$$\begin{aligned} \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1 - \lambda)} &= \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[R_\lambda]}{\log(1 - \lambda)} = \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[Q_\lambda]}{\log(1 - \lambda)} \\ &= \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[L_\lambda]}{\log(1 - \lambda)} = \frac{B}{A - 1}, \end{aligned} \quad (5.15)$$

where  $W_\lambda$ ,  $R_\lambda$ ,  $Q_\lambda$  and  $L_\lambda$  denote the waiting time, response time, queue length and workload distribution for the load balancing policy with load  $\lambda$ .

*Proof.* We first note that if  $\bar{F}'(w) = T_\lambda(\bar{F}(w)) - \bar{F}(w)$  and  $\bar{F}(0) = \lambda$ , then  $\bar{F}(w)$  also satisfies the following FPE:

$$\bar{F}(w) = \lambda - \lambda \int_0^w \left(1 - \frac{T_\lambda(\bar{F}(u))}{\lambda}\right) e^{u-w} du,$$

which can be seen by replacing  $T_\lambda(\bar{F}(u))$  by  $\bar{F}'(u) + F(u)$  and using integration by parts. This FPE can be further simplified to:

$$\bar{F}(w) = \lambda e^{-w} + \int_0^w T_\lambda(\bar{F}(u))e^{u-w} du.$$

Integrating both sides from 0 to infinity, we obtain (using Fubini):

$$\frac{1}{\lambda} \int_0^\infty \bar{F}(w) dw = 1 + \int_0^\infty \frac{T_\lambda(\bar{F}(w))}{\lambda} dw. \quad (5.16)$$

One can see that for the policies considered  $T_\lambda(\bar{F}(w))$  is the arrival rate to servers with  $w$  or more work, from this it follows that  $\frac{T_\lambda(\bar{F}(w))}{\lambda}$  is the probability an arbitrary arrival has a waiting time which exceeds  $w$ . We can thus write (5.16) as  $\frac{\mathbb{E}[L_\lambda]}{\lambda} = 1 + \mathbb{E}[W_\lambda] = \mathbb{E}[R_\lambda]$ .

From this observation, combining (5.16) and Little's law, it follows that the mean workload is indeed equal to the mean queue length. The equations given in (5.15) now easily follow, indeed the first equality follows from  $\mathbb{E}[R_\lambda] = \mathbb{E}[W_\lambda] + 1$ , the second equality is Little's Law, the third equality is what we just proved and the last equality follows by applying Theorem 60.  $\square$

As we already showed all assumptions for LL( $d$ ), it follows by applying Corollary 61 that:

**Corollary 62.** *Let  $W_\lambda$  denote the waiting time distribution for the LL( $d$ ) policy with arrival rate  $\lambda$  and exponential job sizes with mean one, we find:*

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1 - \lambda)} = \frac{1}{d - 1} \quad (5.17)$$

**Remark 50.** *The equality in (5.17) appears in Theorem 7.2 found in [47], however there is an incorrect use of the Moore-Osgood Theorem, as the limit function  $U$  is not necessarily continuous. In fact, its continuity is exactly what needs to be shown. Therefore we presented the first complete proof of this result here.*

For the Red( $d$ ) policy with i.i.d. replicas the waiting time is not clearly defined, therefore we state this result w.r.t. response time, we find from Theorem 60:

**Corollary 63.** Let  $R_\lambda$  denote the response time distribution for the Red( $d$ ) policy with i.i.d. replicas, arrival rate  $\lambda$  and exponential job sizes with mean one, we find:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[R_\lambda]}{\log(1 - \lambda)} = \frac{1}{d - 1}$$

For the memory scheme, in the particular case where servers probe the dispatcher when they become idle, we show in Chapter 6 that

$$\pi_0(\lambda) = \frac{1 - (1 - \lambda^d)^{\frac{1}{M+1}}}{\lambda^d}$$

(with  $M$  the memory size). From this we easily compute the values  $A = d$  and  $B = \frac{1}{M+1}$  and it follows that for the memory based LL( $d$ ) policy we have:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[L_\lambda]}{\log(1 - \lambda)} = \frac{1}{M + 1} \cdot \frac{1}{d - 1},$$

while for the memory based SQ( $d$ ) policy we have:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[L_\lambda]}{\log(1 - \lambda)} = \frac{1}{M + 1} \cdot \frac{1}{\log(d)}.$$

In the next section we show that computing the values of  $A$  and  $B$  is in general not hard. Thus our result allows one to quickly obtain an expression for the limiting value  $B/(A - 1)$  and  $B/\log(A)$ . Of course to formally prove that it is the correct limiting value, the assumptions must be verified.

#### 5.4.4 Computation of the limit

##### LL( $d, K$ ) and SQ( $d, K$ )

For these policies  $T_\lambda(u)$  is given by equation (5.7). First note that by differentiating  $u_\lambda = T_\lambda(u_\lambda)$  one obtains  $\lim_{\lambda \rightarrow 1^-} u'_\lambda = -K/(d - K)$  (see also (5.28)).

We now compute the value  $A$  of assumption 5, to this end we note that:

$$\lim_{\lambda \rightarrow 1^-} h_\lambda(u_\lambda - \lambda^b) = \lim_{\lambda \rightarrow 1^-} \frac{T_\lambda(u_\lambda) - T_\lambda(\lambda^b)}{u_\lambda - \lambda^b}.$$

Furthermore, one can see that in both  $\frac{T_\lambda(u_\lambda)}{u_\lambda - \lambda^b}$  and  $\frac{T_\lambda(\lambda^b)}{u_\lambda - \lambda^b}$  the terms with  $j \geq 2$  of  $T_\lambda$  disappear in the limit of  $\lambda \rightarrow 1^-$ . Therefore we find that:

$$\begin{aligned} \lim_{\lambda \rightarrow 1^-} h_\lambda(u_\lambda - \lambda^b) &= \lim_{\lambda \rightarrow 1^-} \frac{\lambda}{K} \left[ K \frac{u_\lambda^d - \lambda^{bd}}{u_\lambda - \lambda^b} \right. \\ &\quad \left. + (K-1)d \frac{u_\lambda^{d-1}(1-u_\lambda) - \lambda^{b(d-1)}(1-\lambda^b)}{u_\lambda - \lambda^b} \right]. \end{aligned}$$

The result now follows by applying l'Hopital's rule to conclude that:

$$\lim_{\lambda \rightarrow 1^-} h_\lambda(u_\lambda - \lambda^b) = \frac{1}{K} (Kd + (K-1)d(-1)) = \frac{d}{K},$$

completing the proof of assumption 5 with  $A = \frac{d}{K}$ .

For assumption 6 it follows by applying l'Hopital's rule that  $B = 1$ . From this one may conjecture that the limiting value:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = \frac{1}{\frac{d}{K} - 1} = \frac{K}{d-K} \quad (5.18)$$

holds. In Section 5.5 we verify the other assumptions (proving the above limit). Assumption 3 is particularly difficult to show for LL( $d, K$ ).

**LL**( $d_1, \dots, d_n, p_1, \dots, p_n$ ) / **SQ**( $d_1, \dots, d_n, p_1, \dots, p_n$ )

For this policy we have  $T_\lambda(u)$  given by (5.8), from this we find that

$$u'_\lambda = \frac{u_\lambda}{\lambda} + \lambda \sum_{i=1}^n p_i d_i u_\lambda^{d_i-1} u'_\lambda.$$

Taking the limit  $\lambda \rightarrow 1^-$  one finds that  $\lim_{\lambda \rightarrow 1^-} u'_\lambda = \frac{-1}{\sum_i p_i d_i - 1}$ . From this one can show that assumption 5 holds with  $A = \sum_i p_i d_i$  and assumption 6 with  $B = 1$ . We verify the remaining assumptions in Section 5.6.1.

### 5.4.5 Proof of Theorem 59

*Proof.* Throughout the proof we let  $\lambda \in (\bar{\lambda}, 1)$  and we define  $\tilde{u}_k = u_\lambda - u_k$  for all  $k$ . By definition of  $\bar{k}_\lambda$  in assumption 4' we have:

$$\tilde{u}_{\bar{k}_\lambda-1} = u_\lambda - u_{\bar{k}_\lambda-1} \leq u_\lambda - \lambda^b \leq u_\lambda - u_{\bar{k}_\lambda} = \tilde{u}_{\bar{k}_\lambda}.$$

The sequence  $u_k$  decreases to zero as  $\lim_{k \rightarrow \infty} u_k = \lim_{k \rightarrow \infty} T_\lambda(u_k)$  and by the continuity of  $T_\lambda$  we have  $\lim_{k \rightarrow \infty} T_\lambda(u_k) = T_\lambda(\lim_{k \rightarrow \infty} u_k)$ . Hence  $\lim_{k \rightarrow \infty} u_k = 0$  due to assumption 2 as it is a fixed point on  $[0, 1]$  of  $T_\lambda$ . We thus find that  $\tilde{u}_k$  increases to  $u_\lambda \geq 1$  as  $k$  tends to infinity.

We have the following recurrence relation for  $\tilde{u}_k$ :

$$\tilde{u}_{k+1} = u_\lambda - u_{k+1} = u_\lambda - T_\lambda(u_k) = u_\lambda - T_\lambda(u_\lambda - \tilde{u}_k).$$

This allows us to obtain the equality  $\frac{\tilde{u}_{k+1}}{\tilde{u}_k} = h_\lambda(\tilde{u}_k)$  (with  $h_\lambda(x)$  defined as in (5.11)). Furthermore we find from assumptions 3 and 4' that for any  $k \geq \bar{k}_\lambda$ :

$$\frac{\tilde{u}_{k+1}}{\tilde{u}_k} = h_\lambda(\tilde{u}_k) \leq h_\lambda(\tilde{u}_{\bar{k}_\lambda}) \leq h_\lambda(u_\lambda - \lambda^b).$$

Denote  $h_\lambda(u_\lambda - \lambda^b)$  as  $A_\lambda$ . Let  $0 < \varepsilon < 1$  be arbitrarily small and define  $N_{\varepsilon, \lambda} = \max\{k \in \mathbb{N} \mid \tilde{u}_k \leq \varepsilon\}$ . In our proof, we will always take  $\lim_{\lambda \rightarrow 1^-}$  prior to  $\lim_{\varepsilon \rightarrow 0^+}$  therefore we may assume w.l.o.g. that  $\lambda$  is sufficiently close to one such that  $u_\lambda - \lambda^b \leq \varepsilon$ . This in turn implies that  $\bar{k}_\lambda \leq N_{\varepsilon, \lambda}$ , allowing us to write

$$\begin{aligned} \varepsilon \leq \tilde{u}_{N_{\varepsilon, \lambda}+1} &= \frac{\tilde{u}_{N_{\varepsilon, \lambda}+1}}{\tilde{u}_{N_{\varepsilon, \lambda}}} \cdot \dots \cdot \frac{\tilde{u}_{\bar{k}_\lambda}}{\tilde{u}_{\bar{k}_\lambda-1}} \cdot \tilde{u}_{\bar{k}_\lambda-1} \\ &\leq A_\lambda^{N_{\varepsilon, \lambda} - \bar{k}_\lambda + 1} h_\lambda(\tilde{u}_{\bar{k}_\lambda-1}) \tilde{u}_{\bar{k}_\lambda-1} \leq A_\lambda^{N_{\varepsilon, \lambda} - \bar{k}_\lambda + 2} (u_\lambda - \lambda^b), \end{aligned}$$

as  $h_\lambda(x)x$  is increasing in  $x$  on  $(u_\lambda - 1, u_\lambda)$  and  $\tilde{u}_{\bar{k}_\lambda-1} \leq u_\lambda - \lambda^b$ . Taking the logarithm on both sides and rearranging terms, we find the following inequality:

$$\frac{\log(\varepsilon) - \log(u_\lambda - \lambda^b)}{\log(A_\lambda)} \leq N_{\varepsilon, \lambda} - \bar{k}_\lambda + 2.$$

As  $-\log(u_\lambda - \lambda^b)$  tends to infinity when  $\lambda$  tends to one and  $\bar{k}_\lambda$  is bounded by  $\bar{k}$ ,  $N_{\varepsilon, \lambda}$  must tend to infinity as well.

Dividing both sides by  $-\log(1 - \lambda)$  and taking the limit  $\lambda \rightarrow 1^-$  we find from assumptions 4', 5 and 6 that:

$$\frac{B}{\log(A)} \leq \lim_{\lambda \rightarrow 1^-} -\frac{N_{\varepsilon, \lambda}}{\log(1 - \lambda)}.$$

For  $k \leq N_{\varepsilon, \lambda}$  we have  $u_\lambda - u_k = \tilde{u}_k \leq \varepsilon$  and therefore  $1 - \varepsilon \leq u_k$ . From this we find:

$$\frac{(1 - \varepsilon)B}{\log(A)} \leq \lim_{\lambda \rightarrow 1^-} -\frac{(1 - \varepsilon)N_{\varepsilon, \lambda}}{\log(1 - \lambda)} \leq \lim_{\lambda \rightarrow 1^-} -\frac{\sum_{k=0}^{\infty} u_k}{\log(1 - \lambda)}.$$

Letting  $\varepsilon \rightarrow 0^+$  we find the first inequality. For the other inequality we let  $\bar{k}_\lambda \leq k \leq N_{\varepsilon,\lambda}$  be arbitrary. We find that  $(u_\lambda - \lambda^b) \leq \tilde{u}_k \leq \varepsilon$  and therefore we have  $\frac{\tilde{u}_{k+1}}{\tilde{u}_k} = h_\lambda(\tilde{u}_k) \geq h_\lambda(\varepsilon)$  which implies:

$$\varepsilon \geq \tilde{u}_{N_{\varepsilon,\lambda}} = \frac{\tilde{u}_{N_{\varepsilon,\lambda}}}{\tilde{u}_{N_{\varepsilon,\lambda}-1}} \cdot \dots \cdot \frac{\tilde{u}_{\bar{k}_\lambda+1}}{\tilde{u}_{\bar{k}_\lambda}} \cdot \tilde{u}_{\bar{k}_\lambda} \geq (h_\lambda(\varepsilon))^{N_{\varepsilon,\lambda}-\bar{k}_\lambda} (u_\lambda - \lambda^b)$$

Taking the logarithm on both sides and rearranging terms yields:

$$N_{\varepsilon,\lambda} - \bar{k}_\lambda \leq \frac{\log(\varepsilon) - \log(u_\lambda - \lambda^b)}{\log(h_\lambda(\varepsilon))}.$$

Dividing by  $-\log(1 - \lambda)$  and taking the limit  $\lim_{\lambda \rightarrow 1^-}$  on both sides allows us to find from assumption 6:

$$\lim_{\lambda \rightarrow 1^-} -\frac{N_{\varepsilon,\lambda}}{\log(1 - \lambda)} \leq \lim_{\lambda \rightarrow 1^-} -\frac{B}{\log(h_\lambda(\varepsilon))}. \quad (5.19)$$

Note that for any  $k \geq N_{\varepsilon,\lambda} + 1$  we have  $\tilde{u}_k \geq \varepsilon$  and therefore  $u_k \leq u_\lambda - \varepsilon < 1$  for  $\lambda$  large enough. It thus follows from assumption 2 that:

$$\frac{u_{k+1}}{u_k} = \frac{T_\lambda(u_k)}{u_k} \leq \frac{T_\lambda(u_\lambda - \varepsilon)}{u_\lambda - \varepsilon} = C_{\lambda,\varepsilon} < 1.$$

It follows that:

$$\begin{aligned} \sum_{k=N_{\varepsilon,\lambda}+1}^{\infty} u_k &= \sum_{k=N_{\varepsilon,\lambda}+1}^{\infty} u_{N_{\varepsilon,\lambda}} \cdot \frac{u_{N_{\varepsilon,\lambda}+1}}{u_{N_{\varepsilon,\lambda}}} \cdot \dots \cdot \frac{u_k}{u_{k-1}} \\ &\leq \sum_{k=N_{\varepsilon,\lambda}+1}^{\infty} u_\lambda C_{\lambda,\varepsilon}^{k-N_{\varepsilon,\lambda}-1} \\ &= \frac{u_\lambda}{1 - C_{\lambda,\varepsilon}}. \end{aligned}$$

Note that

$$C_{1,\varepsilon} = \lim_{\lambda \rightarrow 1^-} C_{\lambda,\varepsilon} = \lim_{\lambda \rightarrow 1^-} \frac{T_\lambda(u_\lambda - \varepsilon)}{u_\lambda - \varepsilon} = \lim_{\lambda \rightarrow 1^-} \frac{T_\lambda(1 - \varepsilon)}{1 - \varepsilon} < 1$$

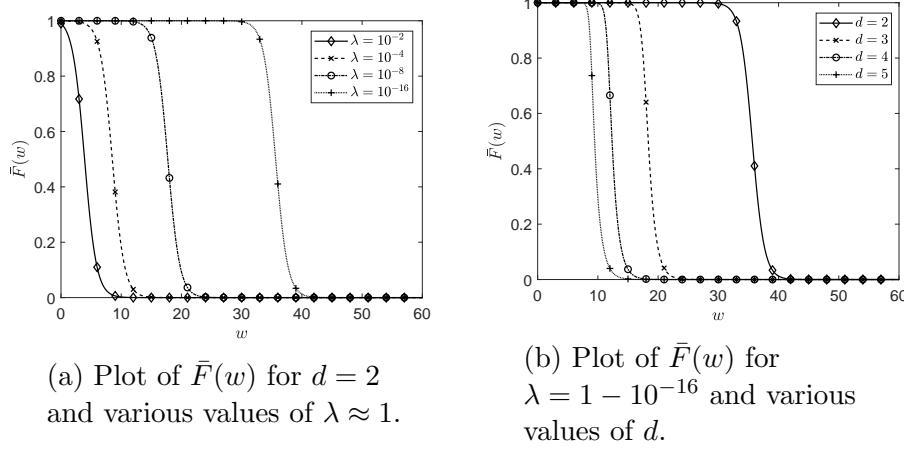


Figure 5.2: This figure illustrates that the tail behaviour of  $\bar{F}(w)$  is the same for all values of  $\lambda$ , this is the main idea used in Theorem 60.

due to the continuity of  $T_\lambda$  and assumption 2. Taking the limit  $\lambda \rightarrow 1^-$  we find that:

$$\begin{aligned} \lim_{\lambda \rightarrow 1^-} -\frac{\sum_{k=0}^{\infty} u_k}{\log(1-\lambda)} &= \lim_{\lambda \rightarrow 1^-} -\frac{\sum_{k=0}^{N_{\varepsilon,\lambda}} u_k}{\log(1-\lambda)} + \lim_{\lambda \rightarrow 1^-} -\frac{\sum_{k=N_{\varepsilon,\lambda}+1}^{\infty} u_k}{\log(1-\lambda)} \\ &\leq \lim_{\lambda \rightarrow 1^-} -\frac{N_{\varepsilon,\lambda} + 1}{\log(1-\lambda)} + \underbrace{\lim_{\lambda \rightarrow 1^-} \left( \frac{1}{1 - C_{1,\varepsilon}} \cdot \frac{1}{-\log(1-\lambda)} \right)}_{=0} \\ &\leq \lim_{\lambda \rightarrow 1^-} \frac{B}{\log(h_\lambda(\varepsilon))}, \end{aligned}$$

by (5.19). Taking the limit  $\varepsilon \rightarrow 0^+$  and applying assumption 7 we obtain the other inequality. This completes the proof.  $\square$

#### 5.4.6 Proof of Theorem 60

The main idea used in the proof of Theorem 60 is the fact that the tail behaviour of  $\bar{F}(w)$  is identical for all values of  $\lambda$ , while the point at which the tail *initiates its descend* moves further to the right as the value of  $\lambda$  approaches 1. This can be seen in Figure 5.2a, where we plot  $\bar{F}(w)$  for  $d = 2$  and  $\lambda = 1 - 10^{-2}, 1 - 10^{-4}, 1 - 10^{-8}$  and  $1 - 10^{-16}$ . Moreover, we observe in Figure 5.2b that increasing the value of  $d$ , moves the tail of the function

$\bar{F}(w)$  to the left which corresponds to having a smaller expected waiting time when  $\lambda \approx 1$ . Our proof boils down to formalizing the idea that there exists some  $w_\lambda$  such that  $\bar{F}(w) \approx 1$  for  $w \leq w_\lambda$ , while the integral  $\int_{w_\lambda}^\infty \bar{F}(w) dw$  remains bounded for all  $\lambda$ .

*Proof.* Our strategy exists in showing that  $\bar{F}(w)$  stays close to one for a long enough time and then decays sufficiently fast to zero. Throughout the proof, we assume that  $\lambda \in (\bar{\lambda}, 1)$ . Due to assumption 2 and  $\bar{F}(w) \in [0, 1]$ , we find that  $\bar{F}(w)$  is decreasing on  $[0, \infty)$  and therefore  $\lim_{w \rightarrow \infty} \bar{F}(w) = z$  exists. As  $0 = \lim_{w \rightarrow \infty} \bar{F}'(w) = \lim_{w \rightarrow \infty} T_\lambda(\bar{F}(w)) - z$  and  $T_\lambda$  is continuous, we have  $0 = T_\lambda(z) - z$ . Hence, assumption 2 yields that  $z = 0$ .

Define  $u_\lambda$  as in assumption 1 and let  $H(w) = u_\lambda - \bar{F}(w)$ . We find:

$$H'(w) = u_\lambda - T_\lambda(u_\lambda - H(w)) - H(w),$$

therefore we have  $\frac{H'(w)}{H(w)} = h_\lambda(H(w)) - 1$ . For any  $w \geq \bar{w}_\lambda$ , we have  $H(w) \geq u_\lambda - \lambda^b$  and due to assumption 3 this yields :

$$\frac{H'(w)}{H(w)} = h_\lambda(H(w)) - 1 \leq h_\lambda(u_\lambda - \lambda^b) - 1. \quad (5.20)$$

Now let  $0 < \varepsilon < 1$  be arbitrary. As  $H(w)$  increases (from  $u_\lambda - \bar{F}(0)$  to  $u_\lambda$ ), we can define  $w_{\varepsilon, \lambda}$  such that  $H(w_{\varepsilon, \lambda}) = \varepsilon$  for  $\lambda$  large enough due to assumption 1 which implies that  $u_\lambda - \bar{F}(0)$  and  $u_\lambda$  tends to 0 and 1, respectively. In fact we assume w.l.o.g. that  $\lambda$  is sufficiently close to one such that  $u_\lambda - \lambda^b \leq \varepsilon$ . Therefore  $\bar{w}_\lambda \leq w_{\varepsilon, \lambda}$  as  $H(w)$  is increasing and  $H(\bar{w}_\lambda) = u_\lambda - \lambda^b$ . By integrating (5.20) from  $\bar{w}_\lambda$  to  $w_{\varepsilon, \lambda}$  we find:

$$\begin{aligned} \log\left(\frac{H(w_{\varepsilon, \lambda})}{H(\bar{w}_\lambda)}\right) &= \int_{\bar{w}_\lambda}^{w_{\varepsilon, \lambda}} \frac{H'(u)}{H(u)} du \\ &\leq (w_{\varepsilon, \lambda} - \bar{w}_\lambda) \cdot (h_\lambda(u_\lambda - \lambda^b) - 1). \end{aligned}$$

Dividing both sides by  $-\log(1 - \lambda)$  and taking the limit  $\lambda \rightarrow 1^-$  we obtain:

$$\begin{aligned} \lim_{\lambda \rightarrow 1^-} -\frac{\log(\varepsilon) - \log(u_\lambda - \lambda^b)}{\log(1 - \lambda)} &= \lim_{\lambda \rightarrow 1^-} -\frac{\log(H(w_{\varepsilon, \lambda})) - \log(u_\lambda - \lambda^b)}{\log(1 - \lambda)} \\ &\leq \lim_{\lambda \rightarrow 1^-} -\left(\frac{w_{\varepsilon, \lambda}}{\log(1 - \lambda)} \cdot (h_\lambda(u_\lambda - \lambda^b) - 1)\right), \end{aligned}$$

as  $\bar{w}_\lambda$  is bounded by  $\bar{w}$  due to assumption 4. Applying assumptions 5 and 6 we obtain:

$$\frac{B}{A-1} \leq \lim_{\lambda \rightarrow 1^-} -\frac{w_{\varepsilon,\lambda}}{\log(1-\lambda)}.$$

For any  $w \leq w_{\varepsilon,\lambda}$  we have  $1-\varepsilon \leq u_\lambda - \varepsilon = \bar{F}(w_{\varepsilon,\lambda}) \leq \bar{F}(w)$ . It follows that:

$$(1-\varepsilon) \frac{B}{A-1} \leq \lim_{\lambda \rightarrow 1^-} -\frac{\int_0^{w_{\varepsilon,\lambda}} (1-\varepsilon) du}{\log(1-\lambda)} \leq \lim_{\lambda \rightarrow 1^-} -\frac{\int_0^\infty \bar{F}(w) dw}{\log(1-\lambda)}.$$

This shows one inequality by letting  $\varepsilon \rightarrow 0^+$ . To show the other we first note that for any  $w \in (\bar{w}_\lambda, w_{\varepsilon,\lambda})$  we have  $u_\lambda - \lambda^b \leq H(w) \leq \varepsilon$  and therefore also:

$$\frac{H'(w)}{H(w)} = h_\lambda(H(w)) - 1 \geq h_\lambda(\varepsilon) - 1.$$

Integrating both sides from  $\bar{w}_\lambda$  to  $w_{\varepsilon,\lambda}$  we find:

$$\log\left(\frac{H(w_{\varepsilon,\lambda})}{H(\bar{w}_\lambda)}\right) \geq (w_{\varepsilon,\lambda} - \bar{w}_\lambda)(h_\lambda(\varepsilon) - 1).$$

Dividing both sides by  $-\log(1-\lambda)$  and taking the limit of  $\lambda \rightarrow 1^-$ , this implies that we have (also use assumption 6):

$$\lim_{\lambda \rightarrow 1^-} -\frac{w_{\varepsilon,\lambda}}{\log(1-\lambda)}(h_\lambda(\varepsilon) - 1) \leq \lim_{\lambda \rightarrow 1^-} -\left(\frac{\log(\varepsilon) - \log(u_\lambda - \lambda^b)}{\log(1-\lambda)}\right) = B.$$

Note that we have:

$$\int_{w_{\varepsilon,\lambda}}^\infty \bar{F}(u) du = \int_{w_{\varepsilon,\lambda}}^\infty \frac{\bar{F}(u)}{\bar{F}'(u)} d\bar{F}(u) = \int_0^{u_\lambda - \varepsilon} \frac{1}{1 - \frac{T_\lambda(x)}{x}} dx, \quad (5.21)$$

assuming that  $\lambda$  is sufficiently close to one, we find from assumption 2 that (5.21) is bounded by

$$(u_\lambda - \varepsilon)^2 / ((u_\lambda - \varepsilon) - T_\lambda(u_\lambda - \varepsilon)),$$

which can be bounded uniformly in  $\lambda$ . This allows us to obtain:

$$\begin{aligned} \lim_{\lambda \rightarrow 1^-} -\frac{\int_0^\infty \bar{F}(w) dw}{\log(1-\lambda)} &\leq \lim_{\lambda \rightarrow 1^-} -\frac{\int_0^{w_{\varepsilon,\lambda}} 1 du}{\log(1-\lambda)} \\ &= \lim_{\lambda \rightarrow 1^-} -\frac{w_{\varepsilon,\lambda}}{\log(1-\lambda)} \\ &\leq \lim_{\lambda \rightarrow 1^-} \frac{B}{h_\lambda(\varepsilon) - 1}. \end{aligned}$$

Taking the limit  $\varepsilon \rightarrow 0^+$  and applying assumption 7, this completes the proof.  $\square$

## 5.5 LL( $d, K$ ) and SQ( $d, K$ )

We consider the LL( $d, K$ ) and SQ( $d, K$ ) models, that is, with rate  $\lambda/K$  a group of  $K$  i.i.d. jobs which have an exponential size with mean 1 arrive to the  $K$  least loaded resp. shortest queues amongst  $d$  randomly selected servers. In this section we give the detailed proof that (5.18) is indeed valid for LL( $d, K$ ). It is shown in Chapter 4 that the ccdf of the equilibrium workload distribution  $\bar{F}(w)$  satisfies the ODE (5.6) with  $T_\lambda(u)$  given by (5.7).

The fact that  $\bar{F}(w)$  is decreasing and  $\int_0^\infty \bar{F}(w) dw < \infty$  is a consequence of the following result:

**Proposition 64.** *For any  $\lambda, u \in (0, 1)$  with  $T_\lambda$  defined as in (5.7) we have  $T_\lambda(u) \leq \lambda u$ . In particular it follows that assumption 4 and 4' hold with  $\bar{w} = b - 1$  and  $\bar{k} = b$ .*

*Proof.* We may compute:

$$\begin{aligned} T_\lambda(u) &= \lambda \sum_{j=0}^{K-1} \frac{K-j}{K} \binom{d}{j} u^{d-j} (1-u)^j \leq \lambda \sum_{j=0}^{K-1} \frac{d-j}{d} \binom{d}{j} u^{d-j} (1-u)^j \\ &= \lambda \sum_{j=0}^{K-1} \frac{d-j}{d} \binom{d}{d-j} u^{d-j} (1-u)^j = \lambda \sum_{j=0}^{K-1} \binom{d-1}{d-j-1} u^{d-j} (1-u)^j \\ &= \lambda u \sum_{j=0}^{K-1} \binom{d-1}{j} u^{d-1-j} (1-u)^j, \end{aligned}$$

and this last sum is bounded by one, from which the result follows.  $\square$

We show that assumption 2 holds, note that the first bullet is immediate from the previous result.

**Lemma 65.** *Let  $T_\lambda$  be defined as in (5.7) and let  $u \in (0, 1)$ , the following inequality holds:*

$$\left( \frac{T_\lambda(u)}{u} \right)' > 0. \quad (5.22)$$

*Proof.* We may divide both sides in (5.22) by  $u^{d-2}$ , we compute:

$$\begin{aligned} \frac{1}{u^{d-2}} \left( \frac{T_\lambda(u)}{u} \right)' &= \frac{\lambda}{K} \sum_{j=0}^{K-1} (K-j) \binom{d}{j} (d-j-1) \left( \frac{1-u}{u} \right)^j \\ &\quad - \frac{\lambda}{K} \sum_{j=1}^{K-1} (K-j) \binom{d}{j} j \left( \frac{1-u}{u} \right)^{j-1}. \end{aligned}$$

Now let  $\xi = \frac{1-u}{u}$  and note that  $\xi \in (0, \infty)$  for  $u \in (0, 1)$ , we find that  $\frac{1}{u^{d-2}} \left( \frac{T_\lambda(u)}{u} \right)'$  can be further simplified as:

$$\begin{aligned} &\frac{\lambda}{K} \sum_{j=0}^{K-2} \left[ (K-j) \binom{d}{j} (d-j-1) - (K-j-1) \binom{d}{j+1} (j+1) \right] \xi^j \\ &+ \frac{\lambda}{K} \binom{d}{K-1} (d-K) \xi^{K-1} = (d-K) \frac{\lambda}{K} \sum_{j=0}^{K-1} \binom{d}{j} \xi^j > 0. \end{aligned}$$

This shows that (5.22) holds.  $\square$

We have the following elementary Lemma:

**Lemma 66.** *Let  $f : [0, 1] \rightarrow \mathbb{R}$  and  $g : [0, 1] \rightarrow [0, \infty)$  be continuous differentiable functions and let  $h_a(x) = f(x) + ag(x)$  for  $a \in [0, 1]$ . If  $f(0) = 0$  and  $f'(0) < 0$  then there exists a value  $a_0 > 0$  such that for all  $a \in [0, a_0]$  the function  $h_a(x)$  has a root in  $[0, 1]$ . Moreover if we let  $x_a = \min\{x \in [0, 1] \mid h_a(x) = 0\}$  we have  $\lim_{a \rightarrow 0^+} x_a = 0$ .*

*Proof.* As  $[0, 1]$  is compact and  $g$  is continuous we have  $A = \max_{x \in [0, 1]} \{g(x)\} < \infty$ . Moreover as  $f$  is continuous and decreasing in 0, we find a  $\delta, \gamma > 0$  such that  $f(\delta) = -\gamma$  and  $f'(x) < 0$  for all  $x \in [0, \delta]$ . If we now let  $a_0 = \frac{\gamma}{A}$ , the result easily follows as  $h_a(0) \geq 0$  and  $h_a(\delta) \leq 0$ .  $\square$

The most difficult assumption to verify for the LL( $d, K$ ) and SQ( $d, K$ ) policies is assumption 3, therefore we first validate the other remaining assumptions (note that we already verified assumptions 2, 4, 5 and 6). We have:

**Lemma 67.** *Let  $1 \leq K < d$  be fixed. There exists a  $\bar{\lambda} < 1$  such that for all  $\lambda \in (\bar{\lambda}, 1)$ , the equation  $T_\lambda(u) = u$  with  $T_\lambda$  defined as in (5.7) has a solution on  $(1, \infty)$ . Moreover, if we let  $u_\lambda$  denote the minimal solution in  $[1, \infty)$  for  $\lambda \in (\bar{\lambda}, 1)$  we have:*

(a)  $\lim_{\lambda \rightarrow 1^-} u_\lambda = 1$ , therefore assumption 1 holds.

(b) We have:  $\lim_{\varepsilon \rightarrow 0^+} \lim_{\lambda \rightarrow 1^-} h_\lambda(\varepsilon) = A$ . Therefore assumption 7 holds.

*Proof.* Dividing both sides of  $u - T_\lambda(u) = 0$  by  $\lambda \cdot u^d$  we find this equation to be equivalent to:

$$\frac{1}{\lambda} \frac{1}{u^{d-1}} - \frac{1}{K} \sum_{j=0}^{K-1} (K-j) \binom{d}{j} \left( \frac{1-u}{u} \right)^j = 0.$$

Let  $\xi = \frac{u-1}{u} \in [0, 1]$  for  $u \geq 1$ , we obtain:

$$\frac{(1-\xi)^{d-1}}{\lambda} - \frac{1}{K} \sum_{j=0}^{K-1} (K-j) \binom{d}{j} (-1)^j \xi^j = 0$$

adding and subtracting  $(1-\xi)^{d-1}$ , we further find this to be equivalent to

$$\left( \frac{1}{\lambda} - 1 \right) (1-\xi)^{d-1} + (1-\xi)^{d-1} - \frac{1}{K} \sum_{j=0}^{K-1} (K-j) \binom{d}{j} (-1)^j \xi^j = 0.$$

If we let  $a = \frac{1}{\lambda} - 1$  we find a value  $a_0 > 0$  from Lemma 66 such that there exists a root at  $\xi_a$  for all  $a \in [0, a_0)$  (as  $f'(0) = 1 - d/K < 0$ ). It now suffices to take  $\bar{\lambda} \geq \frac{1}{1+a_0}$  from which the existence of a root for  $T_\lambda(u) = u$  follows, moreover (a) trivially follows from Lemma 66 as we define  $u_\lambda = \min\{u \in [1, \infty) \mid T_\lambda(u_\lambda) = u_\lambda\}$ .

To show (b) we note that:

$$\lim_{\lambda \rightarrow 1^-} h_\lambda(\varepsilon) = \left( \frac{1 - (1-\varepsilon)^d}{\varepsilon} \right) - \frac{1}{K} \sum_{j=1}^{K-1} (K-j) \binom{d}{j} (1-\varepsilon)^{d-j} \varepsilon^{j-1}.$$

Taking the limit  $\varepsilon \rightarrow 0^+$  of this expression yields the sought result.  $\square$

To show assumption 3 we first need to do some additional work, in particular we compute  $\lim_{\lambda \rightarrow 1^-} u_\lambda^{(n)}$  for  $n = 1, \dots, K+1$  (c.f. Lemma 68). To show this result, we employ the Faà di Bruno formula which states that for functions  $f$  and  $g$  we have:

$$(f \circ g)^{(n)}(x) = \sum_{k=1}^n f^{(k)}(g(x)) \cdot B_{n,k}(g'(x), g''(x), \dots, g^{(n-k+1)}(x)),$$

where  $B_{n,k}$  denotes the exponential Bell polynomial defined as:

$$B_{n,k}(x_1, \dots, x_{n-k+1}) = \sum \frac{n!}{j_1! j_2! \cdots j_{n-k+1}!} \cdot \\ \left( \frac{x_1}{1!} \right)^{j_1} \cdots \left( \frac{x_{n-k+1}}{(n-k+1)!} \right)^{j_{n-k+1}}.$$

Here the sum is taken over all non-negative integers  $j_1, \dots, j_{n-k+1}$  which satisfy:

$$k = j_1 + \cdots + j_{n-k+1} \quad \text{and} \quad n = j_1 + 2j_2 + \cdots + (n-k+1)j_{n-k+1}.$$

Furthermore we employ the fact that:

$$B_{n,k}(1!, \dots, (n-k+1)!) = \frac{n!}{k!} \binom{n-1}{k-1}, \quad (5.23)$$

which are known as the Lah numbers. We are now able to show:

**Lemma 68.** *For any  $d, K$  we have:*

$$\lim_{\lambda \rightarrow 1^-} u_\lambda^{(n)} = (-1)^n n! \frac{d^{n-1} K}{(d-K)^n} \quad (5.24)$$

for  $1 \leq n \leq K$  and

$$\lim_{\lambda \rightarrow 1^-} u_\lambda^{(K+1)} = (-1)^{K+1} (K+1)! \frac{d^K K}{(d-K)^{K+1}} - \frac{d!}{(d-K)!} \left( \frac{K}{d-K} \right)^{K+1} \quad (5.25)$$

*Proof.* As this proof is quite lengthy and technical, we start by giving a sketch of the proof before we delve into the details.

## Sketch of the proof

We define  $\Theta(u) = \frac{1}{\lambda} \frac{T_\lambda(u)}{u}$ , one can compute  $\Theta^{(n)}(u)$  by induction and taking the limit of  $u \rightarrow 1^-$ , it is possible to compute:

$$\Theta^{(n)}(1) = (-1)^{n+1} n! \frac{d-K}{K} \quad \text{for } 1 \leq n \leq K, \quad (5.26)$$

$$\Theta^{(K+1)}(1) = (-1)^{K+1} \frac{d! - (d-K)!(K+1)!}{(d-K)!} \frac{d-K}{K}. \quad (5.27)$$

We now continue by induction to show (5.24-5.25). For the case  $n = 1$  we note that from  $u_\lambda = T_\lambda(u_\lambda)$  it follows that:

$$\begin{aligned} u'_\lambda &= \frac{1}{K} \sum_{j=0}^{K-1} (K-j) \binom{d}{j} u_\lambda^{d-j} (1-u_\lambda)^j + \frac{\lambda}{K} \sum_{j=0}^{K-1} (K-j)(d-j) \binom{d}{j} \\ &u_\lambda^{d-j-1} (1-u_\lambda)^j u'_\lambda - \frac{\lambda}{K} \sum_{j=1}^{K-1} (K-j) j \binom{d}{j} u_\lambda^{d-j} (1-u_\lambda)^{j-1} u'_\lambda. \end{aligned} \quad (5.28)$$

Taking the limit  $\lambda \rightarrow 1^-$  we obtain  $\lim_{\lambda \rightarrow 1^-} u'_\lambda = 1 + \frac{d}{K} \lim_{\lambda \rightarrow 1^-} u'_\lambda$  yielding (5.24) with  $n = 1$ . Let  $2 \leq n \leq K+1$  and note that  $u_\lambda = T_\lambda(u_\lambda)$  may be reformulated as  $1 = \lambda \Theta(u_\lambda)$ . By differentiating both sides  $n \geq 2$  times, it follows that we have:

$$0 = n \left( \frac{\partial}{\partial \lambda} \right)^{n-1} \Theta(u_\lambda) + \lambda \left( \frac{\partial}{\partial \lambda} \right)^n \Theta(u_\lambda). \quad (5.29)$$

It follows from the Faà di Bruno formula that:

$$\left( \frac{\partial}{\partial \lambda} \right)^n \Theta(u_\lambda) = \sum_{k=1}^n \Theta^{(k)}(u_\lambda) B_{n,k}(u'_\lambda, \dots, u_\lambda^{(n-k+1)}) \quad (5.30)$$

where  $B_{n,k}$  denotes the exponential Bell polynomial. We have:

$$B_{n,1}(u'_\lambda, \dots, u_\lambda^{(n)}) = u_\lambda^{(n)}$$

(as  $j_1 = \dots = j_{n-1} = 0$  and  $j_n = 1$ ) and for  $k > 1$  using induction and (5.23) one can show that:

$$\begin{aligned} \lim_{\lambda \rightarrow 1^-} B_{n,k}(u'_\lambda, \dots, u_\lambda^{(n-k+1)}) &= \frac{n!}{k!} \binom{n-1}{k-1} (-1)^n \frac{d^{n-k} K^k}{(d-K)^n}. \\ \lim_{\lambda \rightarrow 1^-} B_{n-1,k}(u'_\lambda, \dots, u_\lambda^{(n-k)}) &= \frac{(n-1)!}{k!} \binom{n-2}{k-1} (-1)^{n-1} \frac{d^{n-k-1} K^k}{(d-K)^{n-1}}. \end{aligned}$$

Therefore, by combining (5.26), (5.29), (5.30) and using Pascal's triangle one can compute that for  $n \leq K+1$ :

$$\lim_{\lambda \rightarrow 1^-} u_\lambda^{(n)} = (-1)^{n+1} \left( \frac{K}{d-K} \right)^{n+1} \Theta^{(n)}(1) - n! \left( \frac{K}{d-K} \right)^n + n!(-1)^n \frac{d^{n-1} K}{(d-K)^n},$$

for  $n \leq K+1$ . Plugging in (5.26) and (5.27) yields (5.24) and (5.25), respectively.

We now repeat the above proof filling with all the details.

## Complete proof

We first show that for  $\Theta(u) = \frac{1}{\lambda} \frac{T_\lambda(u)}{u}$ :

$$\Theta^{(n)}(1) = (-1)^{n+1} n! \frac{d-K}{K}, \quad (5.31)$$

for  $1 \leq n \leq K$  and

$$\Theta^{(K+1)}(1) = (-1)^{K+1} \frac{d! - (d-K)!(K+1)!}{(d-K)!} \frac{d-K}{K}. \quad (5.32)$$

We showed in the proof of Lemma 65 that:

$$\frac{K}{d-K} \Theta'(u) = \sum_{j=0}^{K-1} \binom{d}{j} u^{d-j-2} (1-u)^j.$$

By induction on  $n$  we now show for  $n \leq K$  that

$$\frac{K}{d-K} \Theta^{(n)}(u) = (-1)^{n+1} n! \sum_{j=0}^{K-n} \binom{d}{j} u^{d-j-n-1} (1-u)^j + \sum_{j=1}^{n-1} E_j^{(n-j-1)}, \quad (5.33)$$

where we denote:

$$E_j = (-1)^{j+1} j! \binom{d}{K-j} (d-K-1) u^{d-K-2} (1-u)^{K-j}. \quad (5.34)$$

Indeed, one finds:

$$\begin{aligned} \frac{K}{d-K} \Theta^{(n)}(u) &= \frac{K}{d-K} \frac{\partial}{\partial u} \left[ (-1)^n (n-1)! \sum_{j=0}^{K-n+1} \binom{d}{j} u^{d-j-n} (1-u)^j \right. \\ &\quad \left. + \sum_{j=1}^{n-2} E_j^{(n-j-2)} \right] \\ &= \frac{K}{d-K} \left[ (-1)^{n+1} n! \sum_{j=0}^{K-n} \frac{\binom{d}{j+1} (j+1) - \binom{d}{j} (d-j-n)}{n} \right. \\ &\quad \left. u^{d-j-n-1} (1-u)^j + (-1)^n (n-1)! \binom{d}{K-n+1} (d-K-1) \cdot \right. \\ &\quad \left. u^{d-K-2} (1-u)^{K-n+1} + \sum_{j=1}^{n-2} E_j^{(n-j-1)} \right]. \end{aligned}$$

The result then follows by induction by applying the equality:

$$\frac{\binom{d}{j+1}(j+1) - \binom{d}{j}(d-j-n)}{n} = \binom{d}{j}.$$

Noting that for any  $n \leq K$  we have  $\lim_{u \rightarrow 1} \sum_{j=1}^{n-2} E_j^{(n-j-1)} = 0$ , we find that (5.31) indeed holds. Furthermore we have:

$$\frac{K}{d-K} \Theta^{(K+1)}(u) = (-1)^{K+1} K! (d-K-1) u^{d-K-2} + \sum_{j=1}^{K-1} E_j^{(K-j)}.$$

Moreover, it is not hard to see that:

$$\lim_{u \rightarrow 1} E_j^{(K-j)} = (-1)^{K+1} \binom{d}{K-j} (d-K-1) j! (K-j)!.$$

This allows us to compute:

$$\begin{aligned} \frac{K}{d-K} \Theta^{(K+1)}(1) &= (-1)^{K+1} \left[ K! + \sum_{j=1}^{K-1} \binom{d}{K-j} j! (K-j)! \right] (d-K-1) \\ &= (-1)^{K+1} K! \left[ \sum_{j=0}^{K-1} \frac{\binom{d}{j}}{\binom{K}{j}} \right] (d-K-1) \\ &= (-1)^{K+1} K! \left( \binom{d}{K} - (K+1) \right), \end{aligned}$$

where we used identity (4.1) in [96, p46] with  $j = 0, n = K-1, z = d$  and  $x = K$ . This shows that (5.32) indeed holds.

We now continue by induction to show (5.24-5.25). For the case  $n = 1$  we note that from  $u_\lambda = T_\lambda(u_\lambda)$  it follows that:

$$u'_\lambda = \frac{1}{K} \sum_{j=0}^{K-1} (K-j) \binom{d}{j} u_\lambda^{d-j} (1-u_\lambda)^j \quad (5.35)$$

$$\begin{aligned} &+ \frac{\lambda}{K} \sum_{j=0}^{K-1} (K-j)(d-j) \binom{d}{j} u_\lambda^{d-j-1} (1-u_\lambda)^j u'_\lambda \\ &- \frac{\lambda}{K} \sum_{j=1}^{K-1} (K-j)j \binom{d}{j} u_\lambda^{d-j} (1-u_\lambda)^{j-1} u'_\lambda. \end{aligned} \quad (5.36)$$

Taking the limit  $\lambda \rightarrow 1^-$  we obtain  $\lim_{\lambda \rightarrow 1^-} u'_\lambda = 1 + \frac{d}{K} \lim_{\lambda \rightarrow 1^-} u'_\lambda$  yielding (5.24) with  $n = 1$ . Let  $2 \leq n \leq K + 1$  note that  $u_\lambda = T_\lambda(u_\lambda)$  and therefore also  $1 = \lambda \Theta(u_\lambda)$ . By differentiating both sides  $n \geq 2$  times, it follows that we have:

$$0 = n \left( \frac{\partial}{\partial \lambda} \right)^{n-1} \Theta(u_\lambda) + \lambda \left( \frac{\partial}{\partial \lambda} \right)^n \Theta(u_\lambda). \quad (5.37)$$

It follows from the Faà di Bruno formula that:

$$\left( \frac{\partial}{\partial \lambda} \right)^n \Theta(u_\lambda) = \sum_{k=1}^n \Theta^{(k)}(u_\lambda) B_{n,k}(u'_\lambda, \dots, u_\lambda^{(n-k+1)}) \quad (5.38)$$

where  $B_{n,k}$  denotes the exponential Bell polynomial. We have:

$$B_{n,1}(u'_\lambda, \dots, u_\lambda^{(n)}) = u_\lambda^{(n)}$$

and for  $k > 1$  induction allows us to state for  $n \leq K + 1$  that:

$$\begin{aligned} & \lim_{\lambda \rightarrow 1^-} B_{n,k}(u'_\lambda, \dots, u_\lambda^{(n-k+1)}) \\ &= B_{n,k} \left( (-1) \frac{K}{d-K}, \dots, (-1)^{n-k+1} (n-k+1)! \frac{d^{n-k} K}{(d-K)^{n-k+1}} \right) \\ &= B_{n,k}(1!, \dots, (n-k+1)!) \cdot (-1)^n \frac{d^{n-k} K^k}{(d-K)^n}, \end{aligned}$$

where we used the simple identities

$$\begin{aligned} B_{n,k}(x_1 y, x_2 y, \dots, x_{n-k+1} y) &= B_{n,k}(x_1, \dots, x_{n-k+1}) y^k, \\ B_{n,k}(x_1 z, x_2 z^2, \dots, x_{n-k+1} z^{n-k+1}) &= B_{n,k}(x_1, \dots, x_{n-k+1}) z^n, \end{aligned}$$

with  $y = K/d$  and  $z = -d/(d-K)$ . Using (5.23) we have:

$$\lim_{\lambda \rightarrow 1^-} B_{n,k}(u'_\lambda, \dots, u_\lambda^{(n-k+1)}) = \frac{n!}{k!} \binom{n-1}{k-1} (-1)^n \frac{d^{n-k} K^k}{(d-K)^n}.$$

Analogously, one may compute:

$$\lim_{\lambda \rightarrow 1^-} B_{n-1,k}(u'_\lambda, \dots, u_\lambda^{(n-k)}) = \frac{(n-1)!}{k!} \binom{n-2}{k-1} (-1)^{n-1} \frac{d^{n-k-1} K^k}{(d-K)^{n-1}}.$$

Therefore (5.37), (5.38) and (5.31) imply for  $n \leq K + 1$ :

$$\begin{aligned}
0 &= n \sum_{k=1}^{n-1} \frac{(n-1)!}{k!} \binom{n-2}{k-1} (-1)^{n-1} \frac{d^{n-k-1} K^k}{(d-K)^{n-1}} \Theta^{(k)}(1) \\
&\quad + \sum_{k=2}^n \frac{n!}{k!} \binom{n-1}{k-1} (-1)^n \frac{d^{n-k} K^k}{(d-K)^n} \Theta^{(k)}(1) + \Theta^{(1)}(1) \lim_{\lambda \rightarrow 1^-} u_\lambda^{(n)} \\
&= n! \sum_{k=1}^{n-1} \binom{n-2}{k-1} (-1)^{n+k} \frac{d^{n-k} K^{k-1}}{(d-K)^{n-1}} \\
&\quad + n! \sum_{k=1}^{n-1} \binom{n-2}{k-1} (-1)^{n+k+1} \frac{d^{n-k-1} K^k}{(d-K)^{n-1}} \\
&\quad + n! \sum_{k=2}^{n-1} \binom{n-1}{k-1} (-1)^{n+k+1} \frac{d^{n-k} K^{k-1}}{(d-K)^{n-1}} \\
&\quad + (-1)^n \left( \frac{K}{d-K} \right)^n \Theta^{(n)}(1) + \frac{d-K}{K} \lim_{\lambda \rightarrow 1^-} u_\lambda^{(n)} \\
&= n!(-1)^{n+1} \left( \frac{d}{d-K} \right)^{n-1} + n! \left( \frac{K}{d-K} \right)^{n-1} \\
&\quad + n! \sum_{k=2}^{n-1} \left[ \binom{n-2}{k-1} + \binom{n-2}{k-2} - \binom{n-1}{k-1} \right] \cdot \\
&\quad (-1)^{n+k} \frac{d^{n-k} K^{k-1}}{(d-K)^{n-1}} + (-1)^n \left( \frac{K}{d-K} \right)^n \Theta^{(n)}(1) + \frac{d-K}{K} \lim_{\lambda \rightarrow 1^-} u_\lambda^{(n)}.
\end{aligned}$$

From Pascal's triangle we find that:

$$\lim_{\lambda \rightarrow 1^-} u_\lambda^{(n)} = (-1)^{n+1} \left( \frac{K}{d-K} \right)^{n+1} \Theta^{(n)}(1) - n! \left( \frac{K}{d-K} \right)^n + n!(-1)^n \frac{d^{n-1} K}{(d-K)^n},$$

for  $n \leq K + 1$ . Plugging in (5.31) and (5.32) yields (5.24) and (5.25), respectively.  $\square$

We are now able to show that assumption 3 indeed holds:

**Lemma 69.** *Let  $1 \leq K < d$  be fixed, there exists a  $\tilde{\lambda} < 1$  and  $b \in \mathbb{N}$  (independent of  $\lambda$ ) such that the function  $h_\lambda(x)$  defined as in (5.11) with  $T_\lambda$  as in (5.7) is decreasing as a function of  $x \in [u_\lambda - \lambda^b, u_\lambda]$  for all  $\lambda \in (\tilde{\lambda}, 1)$ .*

*Proof.* As this proof is quite lengthy and technical, we start by giving a sketch of the proof before we delve into the details.

## Sketch of the proof

Throughout, we assume that  $\bar{\lambda} < \lambda < 1$  with  $\bar{\lambda}$  as in Lemma 67. We show there is some  $\tilde{\lambda} \geq \bar{\lambda}$  and  $b \in \mathbb{N}$  which does not depend on the value of  $\lambda$  for which  $h_\lambda(x)$  is decreasing on  $[u_\lambda - \lambda^b, u_\lambda]$  for all  $\lambda \in (\tilde{\lambda}, 1)$ .

We define  $\zeta_\lambda(x) = Kx^2 h'_\lambda(x)$  and note that it suffices to show that  $\zeta_\lambda(x) \leq 0$  for  $\lambda$  sufficiently close to one. To this end one can show that:

$$\zeta'_\lambda(x) = -\lambda(d-K)\binom{d}{K-1}(d-K+1)(1-u_\lambda+x)^{K-1}(u_\lambda-x)^{d-K-1}x. \quad (5.39)$$

This is obviously negative for all  $x \in [u_\lambda - 1, u_\lambda]$ . It thus suffices to show that we can find a value  $b \in \mathbb{N}$  such that  $\zeta_\lambda(u_\lambda - \lambda^b) \leq 0$ . Let us denote  $\Theta(u) = \frac{1}{\lambda} \frac{T_\lambda(u)}{u}$ , we can show that:

$$\lambda^b \zeta_\lambda(u_\lambda - \lambda^b) = -\lambda K u_\lambda \lambda^b (\Theta(u_\lambda) - \Theta(\lambda^b)) \quad (5.40)$$

$$+ \lambda(d-K)(u_\lambda - \lambda^b) \sum_{j=0}^{K-1} \binom{d}{j} \lambda^{b(d-j)} (1-\lambda^b)^j. \quad (5.41)$$

For now let us focus on the case  $K = d - 1$ . For this case we have:

$$\Theta^{(n)}(u) = (-1)^{n+1} n! \frac{1}{d-1} \sum_{j=0}^{d-n-1} \binom{d}{j} u^{d-j-n-1} (1-u)^j, \quad (5.42)$$

employing the Taylor expansion of  $\Theta(u_\lambda)$  at  $\lambda^b$ , we find that (5.41) can be written as:

$$\lambda^b \zeta_\lambda(u_\lambda - \lambda^b) = \lambda \left[ (u_\lambda - \lambda^b) \sum_{j=0}^{d-2} \binom{d}{j} \lambda^{b(d-j)} (1-\lambda^b)^j \right] \quad (5.43)$$

$$- u_\lambda \lambda^b \sum_{n=1}^{d-1} (d-1) \Theta^{(n)}(\lambda^b) \frac{(u_\lambda - \lambda^b)^n}{n!}. \quad (5.44)$$

Combining (5.42) and (5.44) we are able to compute:

$$\begin{aligned} \lim_{\lambda \rightarrow 1^-} \frac{\lambda^b}{(u_\lambda - \lambda^b)^d} \zeta_\lambda(u_\lambda - \lambda^b) &= -d \left( \frac{b}{b+d-1} \right)^{d-1} \\ &+ \left( \frac{b}{b+d-1} \right)^d + (-1)^{1+d} \left( \frac{d-1}{b+d-1} \right)^d, \end{aligned}$$

which converges to  $1 - d \leq 0$  as  $b$  tends to infinity. This proves Lemma 69 for  $K = d - 1$ .

Fix  $K$  and let  $d \geq K + 1$  be variable. Let  $(K_1, d_1)$  and  $(K_2, d_2)$  be arbitrary (with  $K_i < d_i$ ), denote by  $_i u_\lambda$  the fixed point associated to  $(K_i, d_i)$  and  $_i \zeta_\lambda$  the associated  $\zeta_\lambda$  function. We can show the following inequalities:

$${}_2 \zeta'_\lambda(x) \leq {}_1 \zeta'_\lambda(x + {}_1 u_\lambda - {}_2 u_\lambda) \quad \text{for } x \in [{}_2 u_\lambda - 1, {}_2 u_\lambda - \lambda^b] \quad (5.45)$$

$${}_2 \zeta_\lambda({}_2 u_\lambda - 1) \leq {}_1 \zeta_\lambda({}_1 u_\lambda - 1), \quad (5.46)$$

in case we have:

(i)  $K$  is even,  $(K_1, d_1) = (K, d)$  and  $(K_2, d_2) = (K, d + 1)$  or

(ii)  $K$  is odd,  $(K_1, d_1) = (K + 1, d + 1)$  and  $(K_2, d_2) = (K, d)$ .

If (5.45-5.46) hold, we find that:

$$\begin{aligned} {}_2 \zeta_\lambda({}_2 u_\lambda - \lambda^b) &= {}_2 \zeta_\lambda({}_2 u_\lambda - 1) + \int_{{}_2 u_\lambda - 1}^{{}_2 u_\lambda - \lambda^b} {}_2 \zeta'_\lambda(x) dx \\ &\leq {}_1 \zeta_\lambda({}_1 u_\lambda - 1) + \int_{{}_2 u_\lambda - 1}^{{}_2 u_\lambda - \lambda^b} {}_1 \zeta'_\lambda(x + {}_1 u_\lambda - {}_2 u_\lambda) dx \\ &= {}_1 \zeta_\lambda({}_1 u_\lambda - \lambda^b) \end{aligned}$$

This shows that if  ${}_1 \zeta_\lambda({}_1 u_\lambda - \lambda^b) \leq 0$ , then also  ${}_2 \zeta_\lambda({}_2 u_\lambda - \lambda^b) \leq 0$ . Applying (i) then concludes the proof for  $K$  even as we already established the result for  $K = d - 1$ . Having shown the result for  $K$  even then implies that the result also holds for  $K$  odd by applying (ii).

We now repeat the proof filling in all details.

## Complete proof

Throughout, we assume that  $\bar{\lambda} < \lambda < 1$  with  $\bar{\lambda}$  as in Lemma 67. We show there is some  $\tilde{\lambda} \geq \bar{\lambda}$  and  $b \in \mathbb{N}$  which does not depend on the value of  $\lambda$  for which  $h_\lambda(x)$  is decreasing on  $[u_\lambda - \lambda^b, u_\lambda]$  for all  $\lambda \in (\tilde{\lambda}, 1)$ .

First we note that the derivative of  $T_\lambda(u)$  as a function of  $u$  is:

$$\begin{aligned} T'_\lambda(u) &= \frac{\lambda}{K} \sum_{j=0}^{K-1} (K-j)(d-j) \binom{d}{j} u^{d-j-1} (1-u)^j \\ &\quad - \frac{\lambda}{K} \sum_{j=0}^{K-2} (K-j-1)(j+1) \binom{d}{j+1} u^{d-j-1} (1-u)^j \\ &= \frac{\lambda}{K} \sum_{j=0}^{K-1} (d-j) \binom{d}{j} u^{d-j-1} (1-u)^j. \end{aligned}$$

We thus find:

$$\begin{aligned} Kx^2 h'_\lambda(x) &= \left[ -\frac{1}{x^2} (u_\lambda - T_\lambda(u_\lambda - x)) + \frac{1}{x} T'_\lambda(u_\lambda - x) \right] \cdot Kx^2 \\ &= -Ku_\lambda + \lambda \sum_{j=0}^{K-1} (K-j) \binom{d}{j} (u_\lambda - x)^{d-j} (1 - u_\lambda + x)^j \\ &\quad + \lambda x \sum_{j=0}^{K-1} (d-j) \binom{d}{j} (u_\lambda - x)^{d-j-1} (1 - u_\lambda + x)^j \\ &= -Ku_\lambda + \lambda \sum_{j=0}^{K-1} (K-j) \binom{d}{j} (u_\lambda - x)^{d-j} (1 - u_\lambda + x)^j \\ &\quad - \lambda \sum_{j=0}^{K-1} (d-j) \binom{d}{j} (u_\lambda - x)^{d-j} (1 - u_\lambda + x)^j \\ &\quad + \lambda u_\lambda \sum_{j=0}^{K-1} (d-j) \binom{d}{j} (u_\lambda - x)^{d-j-1} (1 - u_\lambda + x)^j. \end{aligned}$$

If we now define  $\zeta_\lambda(x) = Kx^2 h'_\lambda(x)$  we obtain:

$$\begin{aligned} \zeta_\lambda(x) &= -Ku_\lambda - \lambda(d-K) \sum_{j=0}^{K-1} \binom{d}{j} (u_\lambda - x)^{d-j} (1 - u_\lambda + x)^j \\ &\quad + \lambda u_\lambda \sum_{j=0}^{K-1} (d-j) \binom{d}{j} (u_\lambda - x)^{d-j-1} (1 - u_\lambda + x)^j. \end{aligned} \tag{5.47}$$

It therefore suffices to show that  $\zeta_\lambda(x) \leq 0$  for  $\lambda$  sufficiently close to one. To this end we compute:

$$\begin{aligned}
\zeta'_\lambda(x) &= \lambda(d-K)\binom{d}{K-1}(d-K+1)(u_\lambda-x)^{d-K}(1-u_\lambda+x)^{K-1} \\
&\quad + \lambda(d-K)\sum_{j=0}^{K-2}\binom{d}{j}(d-j)(u_\lambda-x)^{d-j-1}(1-u_\lambda+x)^j \\
&\quad - \lambda(d-K)\sum_{j=0}^{K-2}\binom{d}{j+1}(j+1)(u_\lambda-x)^{d-j-1}(1-u_\lambda+x)^j \\
&\quad - \lambda u_\lambda(d-K)(d-K+1)\binom{d}{K-1}(u_\lambda-x)^{d-K-1}(1-u_\lambda+x)^{K-1} \\
&\quad - \lambda u_\lambda\sum_{j=0}^{K-2}(d-j)(d-j-1)\binom{d}{j}(u_\lambda-x)^{d-j-2}(1-u_\lambda+x)^j \\
&\quad + \lambda u_\lambda\sum_{j=0}^{K-2}\binom{d}{j+1}(d-j-1)(j+1)(u_\lambda-x)^{d-j-2}(1-u_\lambda+x)^j,
\end{aligned}$$

which simplifies to:

$$\zeta'_\lambda(x) = -\lambda(d-K)\binom{d}{K-1}(d-K+1)(1-u_\lambda+x)^{K-1}(u_\lambda-x)^{d-K-1}x. \quad (5.48)$$

This is obviously negative for all  $x \in [u_\lambda - 1, u_\lambda]$ . It thus suffices to show that we can find a value  $b \in \mathbb{N}$  such that  $\zeta_\lambda(u_\lambda - \lambda^b) \leq 0$ . To this end, we

find:

$$\begin{aligned}
\zeta_\lambda(u_\lambda - \lambda^b) &= -Ku_\lambda - \lambda(d - K) \sum_{j=0}^{K-1} \binom{d}{j} \lambda^{b(d-j)} (1 - \lambda^b)^j \\
&\quad + \lambda u_\lambda \sum_{j=0}^{K-1} (d - j) \binom{d}{j} \lambda^{b(d-j-1)} (1 - \lambda^b)^j \\
&= -Ku_\lambda - \lambda(d - K) \sum_{j=0}^{K-1} \binom{d}{j} \lambda^{b(d-j)} (1 - \lambda^b)^j \\
&\quad + \frac{\lambda u_\lambda}{\lambda^b} (d - K) \sum_{j=0}^{K-1} \binom{d}{j} \lambda^{b(d-j)} (1 - \lambda^b)^j \\
&\quad + \frac{\lambda u_\lambda}{\lambda^b} \sum_{j=0}^{K-1} (K - j) \binom{d}{j} \lambda^{b(d-j)} (1 - \lambda^b)^j \\
&= -\frac{K\lambda}{\lambda^b} \left( \lambda^b \frac{T_\lambda(u_\lambda)}{\lambda} - u_\lambda \frac{T_\lambda(\lambda^b)}{\lambda} \right) \\
&\quad + \frac{\lambda}{\lambda^b} (d - K) (u_\lambda - \lambda^b) \sum_{j=0}^{K-1} \binom{d}{j} \lambda^{b(d-j)} (1 - \lambda^b)^j.
\end{aligned}$$

Now let us denote  $\Theta(u) = \frac{1}{\lambda} \frac{T_\lambda(u)}{u}$ , we find that:

$$\lambda^b \zeta_\lambda(u_\lambda - \lambda^b) = -\lambda K u_\lambda \lambda^b (\Theta(u_\lambda) - \Theta(\lambda^b)) \quad (5.49)$$

$$+ \lambda(d - K)(u_\lambda - \lambda^b) \sum_{j=0}^{K-1} \binom{d}{j} \lambda^{b(d-j)} (1 - \lambda^b)^j. \quad (5.50)$$

For now let us focus on the case  $K = d - 1$ . By (5.33) and (5.34) we have for  $n \leq K = d - 1$  that

$$\Theta^{(n)}(u) = (-1)^{n+1} n! \frac{1}{d-1} \sum_{j=0}^{d-n-1} \binom{d}{j} u^{d-j-n-1} (1-u)^j, \quad (5.51)$$

as  $E_j = 0$  when  $K = d - 1$ . Note that  $\Theta^{(d-1)}(u) = (-1)^d (d-2)!$  is constant and therefore  $\Theta^{(n)}(u) = 0$  for  $n > K = d - 1$ .

Employing the Taylor expansion of  $\Theta(u_\lambda)$  at  $\lambda^b$ , we find that (5.50) can be written as:

$$\lambda^b \zeta_\lambda(u_\lambda - \lambda^b) = \lambda \left[ (u_\lambda - \lambda^b) \sum_{j=0}^{d-2} \binom{d}{j} \lambda^{b(d-j)} (1 - \lambda^b)^j \right] \quad (5.52)$$

$$- u_\lambda \lambda^b \sum_{n=1}^{d-1} (d-1) \Theta^{(n)}(\lambda^b) \frac{(u_\lambda - \lambda^b)^n}{n!}. \quad (5.53)$$

Due to (5.51)  $u_\lambda \lambda^b \sum_{n=1}^{d-1} (d-1) \Theta^{(n)}(\lambda^b) \frac{(u_\lambda - \lambda^b)^n}{n!}$  can be simplified as:

$$\begin{aligned} &= u_\lambda \sum_{n=1}^{d-1} (-1)^{n+1} (u_\lambda - \lambda^b)^n \sum_{j=0}^{d-n-1} \binom{d}{j} \lambda^{b(d-j-n)} (1 - \lambda^b)^j \\ &= -u_\lambda \sum_{j=0}^{d-2} \binom{d}{j} \lambda^{b(d-j)} (1 - \lambda^b)^j \sum_{n=1}^{d-j-1} (1 - u_\lambda/\lambda^b)^n \\ &= -u_\lambda \sum_{j=0}^{d-2} \binom{d}{j} \lambda^{b(d-j)} (1 - \lambda^b)^j \left( \frac{1 - (1 - u_\lambda/\lambda^b)^{d-j}}{u_\lambda/\lambda^b} - 1 \right) \\ &= \sum_{j=0}^{d-2} \binom{d}{j} (1 - \lambda^b)^j (\lambda^{b(d-j)} (u_\lambda - \lambda^b) + \lambda^b (\lambda^b - u_\lambda)^{d-j}). \end{aligned}$$

Combined with (5.53) this yields:

$$\lambda^b \zeta_\lambda(u_\lambda - \lambda^b) = \lambda^{b+1} \sum_{j=0}^{d-2} (-1)^{d-j+1} \binom{d}{j} (u_\lambda - \lambda^b)^{d-j} (1 - \lambda^b)^j.$$

Dividing by  $(u_\lambda - \lambda^b)^d$  we find that:

$$\frac{\lambda^b}{(u_\lambda - \lambda^b)^d} \zeta_\lambda(u_\lambda - \lambda^b) = \lambda^{b+1} \sum_{j=0}^{d-2} (-1)^{d-j+1} \binom{d}{j} \left( \frac{1 - \lambda^b}{u_\lambda - \lambda^b} \right)^j.$$

It is easy to show by applying l'Hopital's rule and using the fact that  $\lim_{\lambda \rightarrow 1^-} u'_\lambda = -d + 1$  for  $K = d - 1$  that:

$$\lim_{\lambda \rightarrow 1^-} \left( \frac{1 - \lambda^b}{u_\lambda - \lambda^b} \right) = \frac{b}{b + d - 1}.$$

Therefore we find

$$\begin{aligned} \lim_{\lambda \rightarrow 1^-} \frac{\lambda^b}{(u_\lambda - \lambda^b)^d} \zeta_\lambda(u_\lambda - \lambda^b) &= \sum_{j=0}^{d-2} (-1)^{d-j+1} \binom{d}{j} \left( \frac{b}{b+d-1} \right)^j \\ &= -d \left( \frac{b}{b+d-1} \right)^{d-1} + \left( \frac{b}{b+d-1} \right)^d + (-1)^{1+d} \left( \frac{d-1}{b+d-1} \right)^d, \end{aligned}$$

which converges to  $1-d \leq 0$  as  $b$  tends to infinity. This proves Lemma 69 for  $K = d-1$ .

Fix  $K$  and let  $d \geq K+1$  be variable, we find (apply (5.47-5.48) that:

$$\zeta'_\lambda(x) = -\lambda(d-K) \binom{d}{K-1} (d-K+1)(1-u_\lambda+x)^{K-1} (u_\lambda-x)^{d-K-1} x,$$

and

$$\zeta_\lambda(u_\lambda - 1) = (\lambda d - K) u_\lambda - \lambda(d-K). \quad (5.54)$$

Now let  $(K_1, d_1)$  and  $(K_2, d_2)$  be arbitrary (with  $K_i < d_i$ ), denote by  $_i u_\lambda$  the fixed point associated to  $(K_i, d_i)$  and  $_i \zeta_\lambda$  the associated  $\zeta_\lambda$  function. We show the following inequalities :

$${}_2 \zeta'_\lambda(x) \leq {}_1 \zeta'_\lambda(x + {}_1 u_\lambda - {}_2 u_\lambda) \quad (5.55)$$

for  $x \in [{}_2 u_\lambda - 1, {}_2 u_\lambda - \lambda^b]$  and

$${}_2 \zeta_\lambda({}_2 u_\lambda - 1) \leq {}_1 \zeta_\lambda({}_1 u_\lambda - 1), \quad (5.56)$$

in case we have:

- (i)  $K$  is even,  $(K_1, d_1) = (K, d)$  and  $(K_2, d_2) = (K, d+1)$ ,
  - (ii)  $K$  is odd,  $(K_1, d_1) = (K+1, d+1)$  and  $(K_2, d_2) = (K, d)$ .
- If (5.55-5.56) hold, we find that:

$$\begin{aligned} {}_2 \zeta_\lambda({}_2 u_\lambda - \lambda^b) &= {}_2 \zeta_\lambda({}_2 u_\lambda - 1) + \int_{{}_2 u_\lambda - 1}^{{}_2 u_\lambda - \lambda^b} {}_2 \zeta'_\lambda(x) dx \\ &\leq {}_1 \zeta_\lambda({}_1 u_\lambda - 1) + \int_{{}_2 u_\lambda - 1}^{{}_2 u_\lambda - \lambda^b} {}_1 \zeta'_\lambda(x + {}_1 u_\lambda - {}_2 u_\lambda) dx \\ &= {}_1 \zeta_\lambda({}_1 u_\lambda - \lambda^b) \end{aligned}$$

This shows that if  ${}_1\zeta_\lambda({}_1u_\lambda - \lambda^b) \leq 0$ , then also  ${}_2\zeta_\lambda({}_2u_\lambda - \lambda^b) \leq 0$ . Applying (i) would then conclude the proof for  $K$  even as we already established the result for  $K = d - 1$ . Having shown the result for  $K$  even then implies that the result also holds for  $K$  odd by applying (ii).

First, we show (5.55) for (i). To this end we let  $x \in [{}_2u_\lambda - 1, {}_2u_\lambda - \lambda^d]$  be arbitrary, we find that  ${}_2\zeta'_\lambda(x) \leq {}_1\zeta'_\lambda(x + {}_1u_\lambda - {}_2u_\lambda)$  is equivalent to:

$$\left(1 + \frac{{}_1u_\lambda - {}_2u_\lambda}{x}\right) \frac{1}{{}_2u_\lambda - x} \leq \frac{(d_2 - K) \binom{d_2}{K-1} (d_2 - K + 1)}{(d_1 - K) \binom{d_1}{K-1} (d_1 - K + 1)}.$$

This can be shown to hold for  $\lambda$  sufficiently close to 1 by noting that for  $x \in [{}_2u_\lambda - 1, {}_2u_\lambda - \lambda^b]$  we have

$$\begin{aligned} & \lim_{\lambda \rightarrow 1^-} \left(1 + \frac{{}_1u_\lambda - {}_2u_\lambda}{x}\right) \frac{1}{{}_2u_\lambda - x} \leq \lim_{\lambda \rightarrow 1^-} \left(1 + \frac{{}_1u_\lambda - {}_2u_\lambda}{{}_2u_\lambda - 1}\right) \frac{1}{\lambda^b} \\ &= \frac{d_2 - K}{d_1 - K} \\ &\leq \frac{(d_2 - K) \binom{d_2}{K-1} (d_2 - K + 1)}{(d_1 - K) \binom{d_1}{K-1} (d_1 - K + 1)}, \end{aligned}$$

from this we find that (5.55) indeed holds in case (i) for any  $K$  and thus certainly for  $K$  even.

We now consider (5.56) for case (i). Due to (5.54) one finds for any  $n \geq 1$  that:

$$\left(\frac{\partial}{\partial \lambda}\right)^n \zeta_\lambda(u_\lambda - 1) = n du_\lambda^{(n-1)} + (\lambda d - K) u_\lambda^{(n)} - \delta_{\{n=1\}}(d - K). \quad (5.57)$$

We employ (5.24), to conclude that for  $n \leq K$ :

$$\lim_{\lambda \rightarrow 1^-} \left(\frac{\partial}{\partial \lambda}\right)^n \zeta_\lambda(u_\lambda - 1) = 0, \quad (5.58)$$

while for  $n = K + 1$  we find from (5.24-5.25):

$$\lim_{\lambda \rightarrow 1^-} \left(\frac{\partial}{\partial \lambda}\right)^{K+1} \zeta_\lambda(u_\lambda - 1) = -\frac{K \cdot d!}{(d - K)!} \cdot \left(\frac{K}{d - K}\right)^K. \quad (5.59)$$

We now denote

$$H_n = \lim_{\lambda \rightarrow 1^-} \left(\frac{\partial}{\partial \lambda}\right)^n ({}_2\zeta_\lambda({}_2u_\lambda - 1) - {}_1\zeta_\lambda({}_1u_\lambda - 1)), \quad (5.60)$$

From (5.58) we clearly have  $H_n = 0$  for  $0 \leq n \leq K$ . For  $n = K + 1$  we find:

$$H_{K+1} = \frac{d!}{(d-K)!} \cdot \left( K \left( \frac{K}{d-K} \right)^K - (d+1) \left( \frac{K}{d+1-K} \right)^{K+1} \right).$$

which is positive if and only if:

$$(1+d-K)^{K+1} - (d+1)(d-K)^K > 0$$

Letting  $d = K + y$  (for  $y \geq 1$ ) we find that this is equivalent to:

$$(1+y)^{K+1} - (1+K+y)y^K > 0.$$

As  $(1+y)^{K+1} - (1+K+y)y^K = \sum_{j=0}^{K-1} \binom{K+1}{j} y^j$ , which is positive for  $K \geq 2$  and  $y \geq 0$ , we conclude that  $H_{K+1}$  is positive.

By looking at the Taylor series expansion of  ${}_2\zeta_\lambda({}_2u_\lambda - 1) - {}_1\zeta_\lambda({}_1u_\lambda - 1)$  in  $\lambda = 1$  and noting that  $H_0 = \dots = H_K = 0$ , we note that for  $\lambda$  sufficiently close to one:

$${}_2\zeta_\lambda({}_2u_\lambda - 1) - {}_1\zeta_\lambda({}_1u_\lambda - 1) \approx H_{K+1}(\lambda - 1)^{K+1}/(K+1)!,$$

which is negative for  $K$  even. This shows that (5.56) indeed holds for case (i).

We now consider (5.55) for (ii), using simple computations we find that this is equivalent to

$$\binom{d+1}{K} (1 - {}_2u_\lambda + x)(x + {}_1u_\lambda - {}_2u_\lambda) \leq \binom{d}{K-1} x,$$

for  $x \in [{}_2u_\lambda - 1, {}_2u_\lambda - \lambda^b]$ . It therefore suffices to show that for  $\lambda$  close to one, we have:

$$(1 - {}_2u_\lambda + x) \left( 1 + \frac{{}_1u_\lambda - {}_2u_\lambda}{{}_2u_\lambda - 1} \right) \leq \frac{K}{d+1}.$$

This holds as  $(1 - {}_2u_\lambda + x)$  converges to zero as  $\lambda \rightarrow 1^-$  and  $\lim_{\lambda \rightarrow 1^-} 1 + ({}_1u_\lambda - {}_2u_\lambda)/({}_2u_\lambda - 1) = (K+1)/K$ .

The final step is to show (5.56) for (ii). If we define  $H_n$  as in (5.60) and make use of (5.58) and (5.59), we find that  $H_n = 0$  for  $n \leq K$  while for  $n = K + 1$  we have:

$$H_{K+1} = \lim_{\lambda \rightarrow 1^-} \left( \frac{\partial}{\partial \lambda} \right)^{K+1} {}_2\zeta_\lambda({}_2u_\lambda - 1) = -K \left( \frac{K}{d-K} \right)^K \frac{d!}{(d-K)!} < 0.$$

As  $K$  is odd,  $H_{K+1}(\lambda - 1)^{K+1}/(K + 1)!$  is negative, which completes the proof.  $\square$

It follows that assumption 3 indeed holds, therefore we may apply Corollary 61 to show the limiting result for  $\text{LL}(d, K)$  and Theorem 59 for  $\text{SQ}(d, K)$ .

## 5.6 $\text{LL}(d_1, \dots, d_n, p_1, \dots, p_n)$ and $\text{SQ}(d_1, \dots, d_n, p_1, \dots, p_n)$

In this section, we consider a policy which, with probability  $p_i$ , sends an incoming job to the queue with the least amount of work left out of  $d_i$  randomly selected servers. Throughout, we assume that  $\sum_i p_i = 1$ ,  $d_i \geq 1$  and  $\sum_i p_i d_i > 1$ . We assume jobs are exponentially distributed with mean one and the arrival rate is equal to  $\lambda \in (0, 1)$ . Let  $\bar{F}(w)$  denote the probability that a queue in the mean field limit has  $w$  or more work. We find that  $\bar{F}(w)$  can be found as the solution of a simple ODE. The proof of Proposition 70 combines the main idea of the proofs of Theorems 9 and 12 and the result of Theorem 40.

**Proposition 70.** *The ccdf of the workload distribution for  $\text{LL}(d_1, \dots, d_n, p_1, \dots, p_n)$  satisfies ODE (5.6) with  $T_\lambda(u)$  defined as in (5.8).*

*Proof.* The most direct method to show that (5.8) indeed holds is to set  $d = \max_{i=1}^n \{d_i\}$  and note that from Theorem 40 it follows that (for any  $w \in [0, \infty)$ ):

$$\bar{F}'(w) = -\lambda \sum_{i=1}^n p_i d_i \mathbb{P}\{U \leq w, \mathcal{Q}_i(U) > w\}, \quad (5.61)$$

where  $\mathcal{Q}_i(U)$  represents the workload at an arbitrary queue with workload  $U$  after it was one of  $d_i$  selected servers for a job arrival. Note that we have:

$$\mathbb{P}\{U \leq w, \mathcal{Q}_i(U) > w\} = \mathbb{P}\{0 < U \leq w, \mathcal{Q}_i(U) > w\} + \mathbb{P}\{U = 0, \mathcal{Q}_i(U) > w\}.$$

$\mathbb{P}\{0 < U \leq w, \mathcal{Q}_i(U) > w\}$  is equal to (apply integration by parts):

$$\int_0^w f(u) \bar{F}(u)^{d_i-1} e^{u-w} du = \frac{1}{d_i} \left( \lambda^{d_i} e^{-w} - \bar{F}(w)^{d_i} + \int_0^w \bar{F}(u)^{d_i} e^{u-w} du \right)$$

with  $f(u)$  the density of the workload distribution. For  $\mathbb{P}\{U = 0, \mathcal{Q}_i(U) > w\}$  we compute:

$$\begin{aligned} & e^{-w}(1 - \bar{F}(0)) \cdot \sum_{j=0}^{d_i-1} \binom{d_i-1}{j} \frac{(1 - \bar{F}(0))^j \bar{F}(0)^{d_i-1-j}}{j+1} \\ &= e^{-w} \frac{1 - \bar{F}(0)^{d_i}}{d_i} = e^{-w} \frac{1 - \lambda^{d_i}}{d_i}. \end{aligned}$$

This allows us to conclude, using (5.61) that:

$$\bar{F}'(w) = -\lambda \sum_{i=1}^n p_i \left( e^{-w} - \bar{F}(w)^{d_i} + \int_0^w \bar{F}(u)^{d_i} e^{u-w} du \right). \quad (5.62)$$

Integrating both sides of (5.62) we obtain:

$$\begin{aligned} \bar{F}(w) - \bar{F}(0) &= -\lambda \sum_{i=1}^n p_i \left( 1 - e^{-w} - \int_0^w \bar{F}(u)^{d_i} du + \int_0^w \int_0^u \bar{F}(v)^{d_i} e^{v-u} dv du \right) \\ &= -\lambda \sum_{i=1}^n p_i \left( 1 - e^{-w} - \int_0^w e^{u-w} \bar{F}(u)^{d_i} du \right). \end{aligned}$$

We therefore find that:

$$\lambda \sum_{i=1}^n p_i \left( e^{-w} + \int_0^w e^{u-w} \bar{F}(u)^{d_i} du \right) = \bar{F}(w).$$

Using this to further simplify (5.62) allows us to conclude that (5.8) indeed holds.  $\square$

In Section 5.6.1 we show that one may apply Corollary 61. In Section 5.6.2 we answer the question whether a certain choice of  $p_i$  and  $d_i$  is better than another, moreover we show which choice of  $p_i$  and  $d_i$  is optimal given a fixed number of probes which are allowed per arrival (that is  $\sum_i p_i d_i$  fixed). Lastly in Section 5.6.3 we show that when some jobs use the LL( $d$ ) policy while other jobs are assigned arbitrarily, the ODE can be solved explicitly and we use this result to give an alternative proof of our limit result.

### 5.6.1 The limit result

We first show that the requirements to apply Theorem 60 (and therefore also Corollary 61) are satisfied with  $\bar{\lambda} = 0$  in assumption 1,  $b = 0$  in assumption 3 and  $\bar{w} = 0$  resp.  $\bar{k} = 0$  in assumption 4 resp. 4'.

**Lemma 71.** *For any  $\lambda \in (0, 1)$  the equation  $u = T_\lambda(u)$  with  $T_\lambda(u) = \sum_i p_i u^{d_i}$  as in Proposition 70 has exactly one solution  $u_\lambda$  on  $(1, \infty)$  moreover this solution satisfies:*

- (a)  $\lim_{\lambda \rightarrow 1^-} u_\lambda = 1$ , in particular assumption 1 holds.
- (b)  $T_\lambda(u) < u$ ,  $(T_\lambda(u)/u)' \geq 0$  and  $\lim_{\lambda \rightarrow 1^-} T_\lambda(u)/u \leq \lambda < 1$  in particular assumption 2 holds.
- (c) The function  $\xi_i(x) = \frac{u_\lambda^{d_i} - (u_\lambda - x)^{d_i}}{x}$  is decreasing on  $[u_\lambda - 1, u_\lambda]$ . In particular assumption 3 holds with  $b = 0$  as  $h_\lambda(x) = \lambda \sum_i p_i \xi_i(x)$ . Moreover assumptions 4 and 4' hold with  $\bar{w} = 0$  and  $\bar{k} = 0$ .
- (d) We have  $\lim_{\varepsilon \rightarrow 0^+} \lim_{\lambda \rightarrow 1^-} h_\lambda(\varepsilon) = \sum_{i=1}^n p_i d_i$ , in particular assumption 7 holds.

*Proof.* Define the function  $r(u) = \lambda \sum_{i=1}^n p_i u^{d_i} - u$ . We find that  $r(1) = \lambda - 1 < 0$  and it is obvious that  $r(u)$  tends to infinity as  $u$  tends to infinity. This shows that there certainly is a  $u \in (1, \infty)$  for which  $r(u) = 0$ . Now let:

$$u_\lambda = \min\{u \in (1, \infty) \mid r(u) = 0\}.$$

We find that for all  $u > u_\lambda$ :

$$\begin{aligned} r'(u) &= \lambda \sum_{i=1}^n p_i d_i u^{d_i-1} - 1 > \left( \lambda \sum_{i=1}^n p_i d_i u_\lambda^{d_i} - u_\lambda \right) / u_\lambda \\ &= \lambda \sum_{i=1}^n p_i (d_i - 1) u_\lambda^{d_i-1} \geq 0, \end{aligned}$$

this shows that  $r(u) > 0$  for all  $u > u_\lambda$  and hence uniqueness follows. For the other claims we have:

- (a) This follows from  $\lim_{\lambda \rightarrow 1^-} r(1) = 0$ .

- (b) This trivially follows from the fact that  $u^{d_i} \leq u$  for all  $d_i \geq 1$  and  $u \in [0, 1]$ .
- (c) For the function  $\xi_i(x)$  defined in 71(c) we have

$$x^2 \cdot \xi'_i(x) = -(u_\lambda)^{d_i} + (u_\lambda - x)^{d_i-1}(u_\lambda + (d_i - 1)x).$$

Computing the derivative of this, we find:

$$(x^2 \cdot \xi'_i(x))' = -(d_i - 1)d_i(u_\lambda - x)^{d_i-2}x \leq 0,$$

for  $x \in [u_\lambda - 1, u_\lambda]$ . From the fact that  $(u_\lambda - 1)^2 \xi'_i(u_\lambda - 1) = 1 + (u_\lambda - 1)d_i - (u_\lambda)^{d_i} \leq 0$ , we can now incur that  $x^2 \cdot \xi'_i(x)$  is negative on  $[u_\lambda - 1, u_\lambda]$  and therefore  $\xi_i(x)$  is indeed decreasing on  $[u_\lambda - 1, u_\lambda]$ . This suffices to show assumption 3 with  $b = 0$  and assumption 4 with  $\bar{k} = 0$  as we have:

$$h_\lambda(x) = \frac{T(u_\lambda) - \lambda \sum_{i=1}^n p_i (u_\lambda - x)^{d_i}}{x} = \lambda \sum_{i=1}^n p_i \xi_i(x).$$

- (d) First one may compute the limit:

$$\lim_{\lambda \rightarrow 1^-} h_\lambda(\varepsilon) = \frac{1 - \sum_{i=1}^n p_i (1 - \varepsilon)^{d_i}}{\varepsilon},$$

taking the limit of  $\varepsilon \rightarrow 0^+$  we obtain:

$$\lim_{\varepsilon \rightarrow 0^+} \lim_{\lambda \rightarrow 1^-} h_\lambda(\varepsilon) = \sum_{i=1}^n p_i \lim_{\varepsilon \rightarrow 0^+} \frac{(1 - (1 - \varepsilon)^{d_i})}{\varepsilon} = \sum_{i=1}^n p_i d_i.$$

□

Let  $W_\lambda^{(i)}$  and  $R_\lambda^{(i)}$  denote the waiting and response time for a job which is assigned to the server with the Least Loaded (or shortest) queue amongst  $d_i$  randomly selected servers and  $W_\lambda = \sum_i p_i W_\lambda^{(i)}$  and  $R_\lambda = \sum_i p_i R_\lambda^{(i)}$  the waiting and response time for  $LL(d_1, \dots, d_n, p_1, \dots, p_n)$  (or  $SQ(\dots)$ ). We obtain the following limits:

**Corollary 72.** *We have for all  $i$ :*

$$\begin{aligned} \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[R_\lambda^{(i)}]}{\log(1-\lambda)} &= \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[R_\lambda]}{\log(1-\lambda)} = \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda^{(i)}]}{\log(1-\lambda)} \\ &= \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = \frac{1}{\sum_{i=1}^n p_i d_i - 1}. \end{aligned}$$

*Proof.* The last equality follows from Corollary 61 as we verified all assumptions in Lemma 71 and Section 5.4.4. Moreover we have for any  $i$ :

$$\mathbb{E}[W_\lambda^{(i)}] = \int_0^\infty \bar{F}(w)^{d_i} dw \leq \int_0^\infty \bar{F}(w) dw.$$

Therefore we have in the limit  $\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda^{(i)}]}{\log(1-\lambda)} \leq \frac{1}{\sum_i p_i d_i - 1}$  and

$$\begin{aligned} \frac{1}{\sum_i p_i d_i - 1} &= \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} \\ &= \lim_{\lambda \rightarrow 1^-} -\frac{\sum_i p_i \mathbb{E}[W_\lambda^{(i)}]}{\log(1-\lambda)} \\ &= \sum_i p_i \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda^{(i)}]}{\log(1-\lambda)}. \end{aligned}$$

This concludes the proof as  $\sum_i p_i = 1$  and  $p_i \geq 0$  for all  $i$ .  $\square$

### 5.6.2 The impact of $d_i$ and $p_i$

Let  $p_1, \dots, p_n$  and  $d_1, \dots, d_n$  be arbitrary. As the function  $\varphi_u(x) = u^x$  is a convex function for any  $u \in [0, 1]$ , we find that for all  $u \in [0, 1]$  we have  $u^{\sum_{i=1}^n p_i d_i} \leq \sum_{i=1}^n p_i u^{d_i}$ . From Corollary 72 it therefore follows that for any arrival rate  $\lambda \in (0, 1)$  and fixed  $\sum_{i=1}^n p_i d_i \in \mathbb{N}$  the optimal policy is  $\text{LL}(\sum_{i=1}^n p_i d_i)$ , resp.  $\text{SQ}(\sum_{i=1}^n p_i d_i)$ . On the other hand it follows from Theorem 72 that as  $\lambda$  tends to one, the choice of  $p_i$  and  $d_i$  does not affect the mean waiting time. In this section we take a closer look at which choices of  $p_i$  and  $d_i$  yield lower waiting times.

More specifically, one may wonder whether  $\text{LL}(a_1, \dots, a_n, p_1, \dots, p_n)$  (resp.  $\text{SQ}(\dots)$ ) outperforms another policy  $\text{LL}(b_1, \dots, b_n, q_1, \dots, q_n)$  (resp.  $\text{SQ}(\dots)$ ). Moreover, given a maximal amount of average choice  $\sum_{i=1}^n p_i d_i$  on job arrival,

what is the optimal choice of  $p_i \in [0, 1]$  and  $d_i \in \mathbb{N}$ ? To answer these questions we introduce the concept of Majorization with weights which is presented in [61] (Chapter IV, Section 14 A). Specifically the following result is shown (originally introduced in [17], but a more comprehensive proof can be found in [18]):

**Proposition 73.** *Let  $p = (p_1, \dots, p_n)$  and  $q = (q_1, \dots, q_m)$  be fixed vectors with nonnegative components such that  $\sum_{i=1}^n p_i = \sum_{i=1}^m q_i = 1$ . For  $x \in \mathbb{R}^n, y \in \mathbb{R}^m$  the following are equivalent*

- (1) *For all convex functions  $\varphi : \mathbb{R} \rightarrow \mathbb{R}$  we have  $\sum_{i=1}^n p_i \varphi(x_i) \leq \sum_{i=1}^m q_i \varphi(y_i)$ .*
- (2) *There exists an  $m \times n$  matrix  $A = (a_{ij})$  which satisfies  $a_{ij} \geq 0, eA = e$  (with  $e = (1, \dots, 1)$ ),  $Ap^T = q^T$  (with  $p^T$  the transpose of  $p$ ) and  $x = yA$ .*

As a consequence of Proposition 73 we say that  $(p, a)$  is majorized by  $(q, b)$  and write  $(p, a) \preceq (q, b)$  if and only if (1) or (2) in Proposition 73 holds. The interpretation is that  $(q, b)$  is more scattered than  $(p, a)$ . This yields a method for comparing policies as  $(p, a) \preceq (q, b)$  also implies that the workload distribution  $\bar{F}(w)$  of  $\text{LL}(p, a)$  (resp. queue length distribution of  $\text{SQ}(d, p)$ ) is upper bounded by the workload distribution of  $\text{LL}(q, b)$  (resp. queue length distribution of  $\text{SQ}(q, b)$ ).

Despite the fact that given a budget  $\bar{d} = \sum_i p_i d_i$ , the optimal policy is simply  $\text{LL}(\bar{d})$ , we may have  $\bar{d} \notin \mathbb{N}$ . In this case we simply use  $\text{LL}((p, 1-p), (\lfloor \bar{d} \rfloor, \lceil \bar{d} \rceil))$  for an appropriate  $p \in [0, 1]$ . We show that this is indeed the optimal choice (here  $\lfloor \bar{d} \rfloor$  denotes the floor and  $\lceil \bar{d} \rceil$  denotes the ceil of  $\bar{d}$ ).

**Theorem 74.** *Let  $p = (p_1, \dots, p_n)$  with  $\sum_{i=1}^n p_i = 1, p_i \geq 0$  and  $d = (d_1, \dots, d_n)$  with  $d_i \in \mathbb{N}$ . If we let  $\bar{d} = \sum_{i=1}^n p_i d_i$  and  $q = (q_1, q_2)$  s.t.  $q_1 + q_2 = 1$  and  $q_1 \lfloor \bar{d} \rfloor + q_2 \lceil \bar{d} \rceil = \bar{d}$  then  $(q, (\lfloor \bar{d} \rfloor, \lceil \bar{d} \rceil)) \preceq (p, d)$ .*

*Proof.* We show that Proposition 73, ((2)) holds, to this end we let  $A = (a_{ij}) \in \mathbb{R}^{n,2}$ . From  $Aq^T = p$  it follows that for all  $j$ :

$$a_{1j} = \frac{p_j - q_2 a_{2j}}{q_1}. \quad (5.63)$$

It is not hard to see that one can indeed choose  $0 \leq a_{2j} \leq \frac{p_j}{q_2}$  such that  $\sum_{j=1}^n a_{2j} d_j = \lceil \bar{d} \rceil$  and  $\sum_j a_{2j} = 1$ , it then automatically follows from (5.63)

that also  $\sum_j a_{1j} = 1$ . Moreover it follows that:

$$\sum_{j=1}^n a_{1j} d_j = \frac{\bar{d}}{q_1} - \frac{q_2}{q_1} \lceil \bar{d} \rceil = \lfloor \bar{d} \rfloor.$$

This completes the proof.  $\square$

### 5.6.3 LL( $d, p$ )

We take a closer look at the particular case where  $n = 2$  and  $d_2 = 1$ , we denote  $p = p_1$  and thus  $p_2 = 1 - p$ . We write LL( $d, p$ ) as a shorthand for LL( $d_1, d_2, p_1, p_2$ ). In practice, this policy can be viewed as having two arrival streams : one at each server individually, at rate  $\lambda(1 - p)$  for which there is no load balancing and a second at rate  $\lambda p N$  which is distributed using the LL( $d$ ) load balancing policy. It turns out that (as for LL( $d$ ) in Chapter 2), this policy has a closed form solution for the ccdf of the workload distribution:

**Proposition 75.** *The equilibrium workload distribution for the LL( $d, p$ ) policy with exponential job sizes of mean one is given by (5.5).*

*Proof.* In this case, the ODE defined in (5.8) reduces to:

$$\bar{F}'(w) = \lambda \left( p\bar{F}(w)^d + (1 - p)\bar{F}(w) - \frac{\bar{F}(w)}{\lambda} \right). \quad (5.64)$$

This is an autonomous ODE, we find that it can be solved explicitly simply by writing it as:

$$\frac{d\bar{F}(w)}{p\bar{F}(w)^d + (1 - p)\bar{F}(w) - \frac{\bar{F}(w)}{\lambda}} = \lambda dw,$$

integrating and rewriting in function of  $\bar{F}(w)$  yields (5.5).  $\square$

From this we find:

**Proposition 76.** *The mean queue length for the LL( $d, p$ ) policy with exponential job sizes of mean one is given by:*

$$\mathbb{E}[Q_\lambda] = \frac{\lambda}{1 - (1 - p)\lambda} \sum_{n=0}^{\infty} \frac{1}{1 + n(d - 1)} \left( \frac{p\lambda^d}{1 - (1 - p)\lambda} \right)^n. \quad (5.65)$$

In particular for  $d = 2$  we have:

$$\mathbb{E}[Q_\lambda] = -\frac{\log \left(1 - \frac{p\lambda^2}{1-(1-p)\lambda}\right)}{p\lambda}.$$

*Proof.* Recall from Corollary 61 that the average queue length equals the average workload. The remaining proof goes along the same lines as the proof of Theorem 13 and relies on the Hypergeometric function  ${}_2F_1(a, b, c; z)$  for which the following two properties hold:

$${}_2F_1(a, b, c; z) = (1 - z)^{-a} \cdot {}_2F_1\left(a, c - b, c; \frac{z}{z - 1}\right) \quad (5.66)$$

$${}_2F_1(a, b, c; z) = \sum_{n=0}^{\infty} \frac{(a)_n (b)_n}{(c)_n} \frac{z^n}{n!} \quad \text{if } |z| < 1. \quad (5.67)$$

Here  $(\cdot)_n$  is the Pochhammer symbol (or falling factorial) we have  $(q)_n = \prod_{k=0}^{n-1} (q + k)$ . We apply (5.66) to ensure that  $z \in (0, 1)$  which in turn allows us to apply the sum formula (5.67).

The mean workload is given by  $\int_0^\infty \bar{F}(w) dw$ . Using  $y = e^{-w}$  we find that it equals:

$$-\lambda \int_0^1 \left( \frac{b}{p\lambda^d + (b - p\lambda^d)y^{b(d-1)}} \right)^{\frac{1}{d-1}} \frac{1}{y} dy, \quad (5.68)$$

with  $b = 1 - (1 - p)\lambda$ . By definition of the Hypergeometric function (5.68) is equal to

$$\frac{\lambda}{b} \left(1 + \frac{p\lambda^d}{b - p\lambda^d}\right)^{\frac{1}{d-1}} {}_2F_1\left(\frac{1}{d-1}, \frac{1}{d-1}, 1 + \frac{1}{d-1}; \frac{-p\lambda^d}{b - p\lambda^d}\right).$$

Equality (5.66) allows us to rewrite the mean workload as

$$\frac{\lambda}{b} {}_2F_1\left(\frac{1}{d-1}, 1, 1 + \frac{1}{d-1}; \frac{p\lambda^d}{b}\right).$$

As  $p\lambda^d/b \in (0, 1)$ , (5.67) implies that the mean workload is given by:

$$\frac{\lambda}{b} \sum_{n=0}^{\infty} \frac{\left(\frac{1}{d-1}\right)_n (1)_n}{\left(1 + \frac{1}{d-1}\right)_n} \left(\frac{p\lambda^d}{b}\right)^n.$$

Using this and the fact that  $(1)_n = n!$ , we obtain the result.  $\square$

The mean waiting time is now given by  $\mathbb{E}[W_\lambda] = \frac{\mathbb{E}[Q_\lambda]}{\lambda} - 1$ , using this, all results involving mean queue length can easily be adapted to mean waiting time. We find a simple lower and upper bound for the mean queue length:

**Proposition 77.** *We have:*

$$\tilde{Q}_\lambda - \frac{\lambda^{d+1}}{p(d-1)^2(1-(1-p)\lambda)^2} \frac{\pi^2}{6} \leq \mathbb{E}[Q_\lambda] \leq \tilde{Q}_\lambda,$$

with:

$$\tilde{Q}_\lambda = \frac{\lambda}{1-(1-p)\lambda} \cdot \left( 1 + \frac{1}{d-1} \log \left( \frac{1-(1-p)\lambda}{1-(1-p)\lambda - p\lambda^d} \right) \right). \quad (5.69)$$

*Proof.* Throughout this proof we denote  $z = \frac{p\lambda^d}{1-(1-p)\lambda}$ , where  $z < 1$  for  $\lambda < 1$ . We first note that as  $\log(1/(1-z)) = \sum_{n \geq 1} z^n/n$ ,  $\tilde{Q}_\lambda$  can be written as

$$\tilde{Q}_\lambda = \frac{\lambda}{1-(1-p)\lambda} \left( 1 + \sum_{n=1}^{\infty} \frac{z^n}{n(d-1)} \right).$$

From this it is obvious that  $\mathbb{E}[Q_\lambda] \leq \tilde{Q}_\lambda$ . Furthermore we find:

$$\begin{aligned} \tilde{Q}_\lambda - \mathbb{E}[Q_\lambda] &= \frac{\lambda}{(d-1)(1-(1-p)\lambda)} \sum_{n=1}^{\infty} \frac{z^n}{(1+n(d-1))n} \\ &\leq \frac{\lambda}{(d-1)^2(1-(1-p)\lambda)} \sum_{n=1}^{\infty} \frac{z^n}{n^2} \leq \frac{\lambda z \pi^2}{6(d-1)^2(1-(1-p)\lambda)}, \end{aligned}$$

as  $\sum_{n \geq 1} 1/n^2 = \pi^2/6$ . This concludes the proof.  $\square$

Similar bounds for  $\text{LL}(d)$ , i.e. when  $p = 1$  were not presented in Chapter 2. Using these bounds we obtain an alternative proof for the limiting result in Theorem 60 for the special case where  $n = 2$  and  $d_2 = 1$ . Indeed, a simple application of l'Hopital's rule yields that:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[\tilde{Q}_\lambda]}{\log(1-\lambda)} = \frac{1}{p(d-1)}.$$

## 5.7 Low load limit

From the result  $\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = x$ , one could argue that for a sufficiently high value of  $\lambda$ , we have  $\mathbb{E}[W_\lambda] \approx -x \cdot \log(1-\lambda)$ . Instead of taking the limit  $\lambda \rightarrow 1^-$  in (5.15) (and (5.13)), one could also consider the limit  $\lambda \rightarrow 0^+$ . However, it is not hard to see that for all policies we considered this limit is simply equal to zero, yielding  $\mathbb{E}[W_\lambda] \approx 0$  for sufficiently small values of  $\lambda$  which is not all too useful as an approximation.

We therefore introduce the value  $p_\lambda$  which denotes the probability that a job is assigned to an idle queue. For the LL( $d$ )/SQ( $d$ ) policy, it is not hard to see that any job is assigned to an idle queue with probability  $p_\lambda = 1 - \lambda^d$ .

We suggest to use  $-\log(p_\lambda)$  as the scaling factor instead of  $-\log(1-\lambda)$  such that we also get a non-zero limit when  $\lambda$  tends to zero, that is, we consider the fraction  $-\frac{\mathbb{E}[W_\lambda]}{\log(p_\lambda)}$ . We show that the limit for  $\lambda$  tending to one is not altered with the new scaling factor. In Section 5.8 we show that this new scaling can be used as a useful approximation of the expected waiting time when  $\lambda$  is close to 0 or 1.

We now explain how to compute  $p_\lambda$  and the associated limits for the policies considered in this work.

### 5.7.1 LL( $d$ )

For the LL( $d$ ) policy we have  $p_\lambda = 1 - \lambda^d$ . It follows that:

**Proposition 78.** *For the LL( $d$ ) policy with exponential job sizes of mean one we have:*

$$\lim_{\lambda \rightarrow 0^+} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda^d)} = \frac{1}{d} \quad \text{and} \quad \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda^d)} = \frac{1}{d-1}. \quad (5.70)$$

*Proof.* It is easy to see that  $\lim_{\lambda \rightarrow 1^-} \frac{\log(1-\lambda)}{\log(1-\lambda^d)} = 1$ , which shows that the limit  $\frac{1}{d-1}$  when  $\lambda$  tends to one remains valid. For the low load limit we only need to consider the case where the dispatcher selects  $d$  servers which all have exactly one job in their queue. The LL( $d$ ) policy assigns the incoming job to the server which finishes its job first. Therefore we find that the mean waiting time for the LL( $d$ ) policy (for  $\lambda \approx 0$ ) can be approximated by  $(1-p_\lambda)/d = \frac{\lambda^d}{d}$ . Therefore the result follows from the fact that  $\lim_{\lambda \rightarrow 0^+} -\frac{\lambda^d}{\log(1-\lambda^d)} = 1$ .  $\square$

### 5.7.2 LL( $d, K$ )

For the LL( $d, K$ ) policy we can compute  $p_\lambda$  using the same ideas and we obtain the following result:

**Proposition 79.** *For the LL( $d, K$ ) policy with exponential job sizes with mean one, we find that:*

$$\lim_{\lambda \rightarrow 0^+} -\frac{\mathbb{E}[W_\lambda]}{\log(p_\lambda)} = \frac{1}{d - K + 1} \quad \text{and} \quad \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(p_\lambda)} = \frac{K}{d - K} \quad (5.71)$$

$$\text{with } p_\lambda = 1 - \sum_{j=0}^{K-1} \frac{K-j}{K} \binom{d}{j} (1-\lambda)^j \lambda^{d-j}.$$

*Proof.* We first compute the probability  $p_\lambda$  that an arbitrary job is assigned to an idle server for the LL( $d, K$ ) policy. It is not hard to see that  $p_\lambda$  is given by:

$$\begin{aligned} p_\lambda &= \frac{\mathbb{E}[\min\{\text{number of idle servers selected}, K\}]}{K} \\ &= \frac{\sum_{j=0}^d \min\{j, K\} \binom{d}{j} (1-\lambda)^j \lambda^{d-j}}{K}. \end{aligned}$$

This simplifies to the given formula for  $p_\lambda$  by using  $1 = \sum_{j=0}^d \binom{d}{j} (1-\lambda)^j \lambda^{d-j}$ .

The low load limit can be computed by noting that the only situation we need to look at is a probe which finds  $K - 1$  idle servers and  $d - K + 1$  servers which are processing a single job. For the job with non-zero waiting time, we find that the expected waiting time is given by the minimum of  $d - K + 1$  exponential jobs. One finds that the expected waiting time (for  $\lambda \approx 0$ ) is given by  $\lambda^{d-K+1} \frac{\binom{d}{K-1}}{K \cdot (d-K+1)}$ . From this the limit for  $\lambda$  tending to 0 follows after computing:

$$\begin{aligned} &\lim_{\lambda \rightarrow 1^-} -\frac{\lambda^{d-K+1} \frac{\binom{d}{K-1}}{K \cdot (d-K+1)}}{\log \left( 1 - \sum_{j=0}^{K-1} \frac{K-j}{K} \binom{d}{j} (1-\lambda)^j \lambda^{d-j} \right)} \\ &= \lim_{\lambda \rightarrow 1^-} -\frac{\lambda^{d-K+1} \frac{\binom{d}{K-1}}{K \cdot (d-K+1)}}{\log \left( 1 - \frac{1}{K} \binom{d}{K-1} (1-\lambda)^{K-1} \lambda^{d-K+1} \right)} \\ &= \frac{1}{d - K + 1}. \end{aligned}$$

For the limit of  $\lambda$  tending to 1, one simply needs to show that  $\frac{\log(p_\lambda)}{\log(1-\lambda)}$  converges to one as  $\lambda \rightarrow 1^-$ , which follows by applying l'Hopital's rule.  $\square$

### 5.7.3 LL( $d_1, \dots, d_n, p_1, \dots, p_n$ )

For the LL( $d_1, \dots, d_n, p_1, \dots, p_n$ ) policy we find that the probability of assigning a job to an empty server is equal to  $p_\lambda = \sum_i p_i \cdot (1 - \lambda^{d_i}) = 1 - \sum_i p_i \lambda^{d_i}$ . We have the following result:

**Proposition 80.** *For the LL( $d_1, \dots, d_n, p_1, \dots, p_n$ ) policy with exponential job sizes with mean one we have:*

$$\lim_{\lambda \rightarrow 0^+} -\frac{\mathbb{E}[W_\lambda]}{\log(p_\lambda)} = \frac{1}{d_j} \quad \text{and} \quad \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(p_\lambda)} = \frac{1}{\sum_i p_i d_i - 1} \quad (5.72)$$

with  $p_\lambda = 1 - \sum_i p_i \lambda^{d_i}$  and  $j = \min\{i \mid p_i > 0\}$ .

*Proof.* The proof follows the same lines as the proof of Proposition 78 noting that as  $\lambda$  gets close to zero, one will only find exclusively busy servers when probing  $d_j$  servers.  $\square$

### 5.7.4 Queue length dependent load balancing policies

We observed in the previous subsections that the obtained low load limit for variants of the LL( $d$ ) policy are always smaller than one. This is a consequence of the fact that when we select  $d$  queues which all have exactly one job, the LL( $d$ ) policy still performs better than random routing as we can assign the job to the queue which will finish its job first. In contrast, any queue length dependent load balancing policy will just assign the job arbitrarily when all the selected servers have equal loads. From this (or analytical methods similar to those used in the previous sections), one finds that for the same definition of  $p_\lambda$  the limits for the SQ-variants are given by:

$$\lim_{\lambda \rightarrow 0^+} -\frac{\mathbb{E}[W_\lambda]}{\log(p_\lambda)} = 1 \quad \text{and} \quad \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(p_\lambda)} = \frac{B}{\log(A)}. \quad (5.73)$$

## 5.8 Numerical experiments

In Figure 5.3a, we show the evolution of  $-\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)}$  as a function of  $\lambda$  for the LL( $d$ ) policy. We observe that (as stated before) the high load limit is given as  $\frac{1}{d-1}$ , while the low load limit is simply zero. However in Figure 5.3b, we

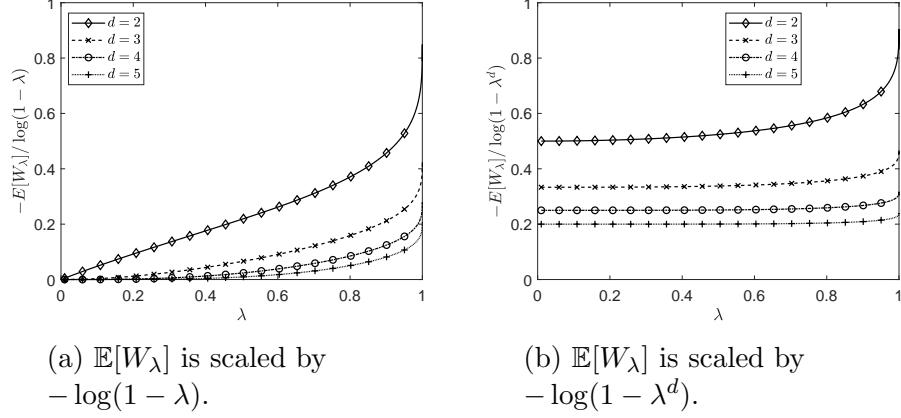


Figure 5.3:  $\mathbb{E}[W_\lambda]$  as a function of  $\lambda$  for the  $\text{LL}(d)$  policy with different scalings.

used the alternate scaling  $-\log(1 - \lambda^d)$  as argued in Section 5.7. Figure 5.3b also shows that the alternate scaling flattens the curve, which shows that  $-\log(1 - \lambda^d)/d$  resp.  $-\log(1 - \lambda^d)/(d - 1)$  are better approximations of  $\mathbb{E}[W_\lambda]$  (which become exact as  $\lambda \approx 0$  resp.  $\lambda \approx 1$ ).

In Figure 5.4 we repeat the experiment conducted in Figure 5.3 for the  $\text{LL}(d, K)$  policy instead of the  $\text{LL}(d)$  policy. We fix  $K = 2$  and vary the value of  $d$ , we again observe that using the alternate scaling  $-\log(p_\lambda)$  yields a flatter curve and thus also a better approximation for  $\lambda$  close to zero or one.

## 5.9 Conclusions and extensions

### Conclusions

In this chapter we studied the behaviour of the expected waiting time  $\mathbb{E}[W_\lambda]$  when  $\lambda \approx 1$  for a variety of load balancing policies in the mean field regime. We present a set of sufficient assumptions such that the limit

$$\lim_{\lambda \rightarrow 1^-} -\mathbb{E}[W_\lambda]/\log(1 - \lambda)$$

can be derived without much effort. For some load balancing policies (such as  $\text{LL}(d)/\text{SQ}(d)$ ) these assumptions are easy to verify, while for other policies

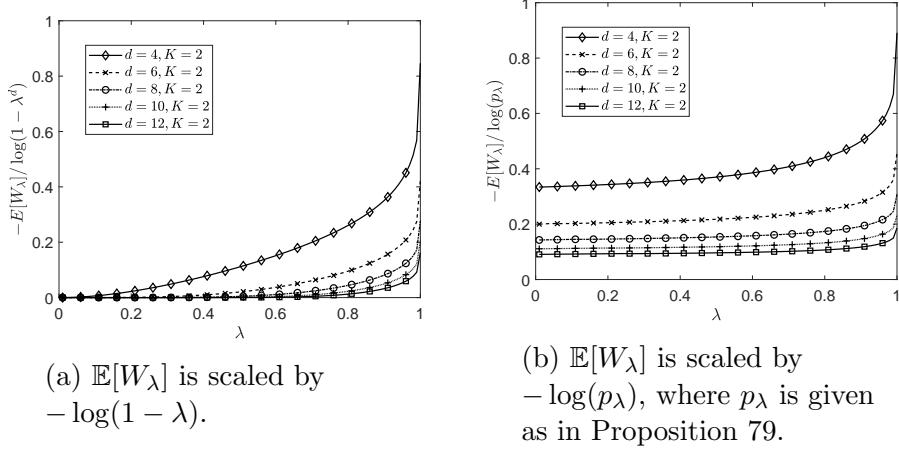


Figure 5.4:  $\mathbb{E}[W_\lambda]$  as a function of  $\lambda$  for the  $\text{LL}(d, K)$  policy with different scalings.

(such as  $\text{LL}(d, K)/\text{SQ}(d, K)$ ) this turned out to be much more challenging. Even if it is unclear how to verify these assumptions, our result yields a natural conjecture on the limiting value. The resulting limiting value is also surprisingly elegant for the policies studied in this chapter. Adjusting the denominator allows one to retain the limiting value as  $\lambda \rightarrow 1^-$  while also obtaining an elegant low load limit. As our main theorem applies to any ODE/recurrence relation for which  $T_\lambda$  satisfies the sufficient assumptions, our main results may also find applications outside the area of load balancing.

## Extensions

Numerical experiments (not reported in this chapter) suggest that the observations made in Figure 5.1 also hold for non-exponential job size distributions, which suggests that the main ideas presented in this chapter may also be applicable to other job size distributions. We have devoted Section 8.4 to open problems related to the theory presented in this Chapter. In particular we already have some partial results when considering batch arrivals, Erlang- $k$  job sizes and deterministic job sizes.

# Chapter 6

## Performance analysis of load balancing policies with memory

He says: "Son can you play me  
a memory?"

I'm not really sure how it goes  
But it's sad and it's sweet and I  
knew it complete  
When I wore a younger man's  
clothes.

---

Billy Joel

### Abstract

Joining the shortest or least loaded queue among  $d$  randomly selected queues are two fundamental load balancing policies. Under both policies the dispatcher does not maintain any information on the queue length or load of the servers. In this chapter we analyse the performance of these policies when the dispatcher has some memory available to store the ids of some of the idle servers. We consider methods where the dispatcher discovers idle servers as well as methods where idle servers inform the dispatcher about their state.

We focus on large-scale systems and our analysis uses the cavity method. The main insight provided is that the performance measures obtained via the cavity method for a load balancing policy *with* memory reduce to the performance measures for the same policy *without* memory provided that

the arrival rate is properly scaled. Thus, we can study the performance of load balancers with memory in the same manner as load balancers without memory. In particular this entails closed form solutions for joining the shortest or least loaded queue among  $d$  randomly selected queues with memory in case of exponential job sizes. Moreover, we obtain a simple closed form expression for the (scaled) expected waiting time as the system tends towards instability.

We present simulation results that support our belief that the approximation obtained by the cavity method becomes exact as the number of servers tends to infinity.

## 6.1 Introduction

Load balancing is often used in large-scale clusters to reduce latency. A simple algorithm, denoted by  $SQ(d)$ , exists in assigning incoming jobs to a server that currently holds the least number of jobs out of  $d$  randomly selected queues. This is referred to as the *power-of-d-choices* algorithm [5, 67, 95]. Another popular algorithm which has received quite some attention recently exists in assigning an incoming job to the server which is the least loaded amongst  $d$  randomly selected queues, i.e. the server which is able to start working on the job first receives the job. This policy is referred to as  $LL(d)$  and has been studied in e.g. Chapter 2 and [21, 74, 68].

The main objective of Chapter 6 is to generalize the analysis of the  $SQ(d)$  and  $LL(d)$  policy to the case where the dispatcher has some (finite or infinite) memory available to store the ids of idle servers. These ids may be discovered by either probing servers to check whether they are idle or servers may inform the dispatcher that they became idle. We focus on the performance of large scale systems and as such make use of the cavity method introduced in [20]. The cavity method relies on the assumption that the queue length (or load) of any finite set of queues becomes independent as the number of servers tends to infinity, called the *ansatz*.

The ansatz was proven in some particular cases, in [21] it was shown for  $LL(d)$  with general job sizes and  $SQ(d)$  with DHR job sizes. Recently, the ansatz was also proven a variety of load balancing policies which are similar to  $LL(d)$  (see [83]). Our objective is not to prove the ansatz for load balancers with memory, but to study these policies using the cavity method. To demonstrate the usefulness of our analysis we present simulation results

which suggest that the cavity method captures the system behavior as the number of servers tends to infinity.

A few papers have previously studied the use of some (bounded) memory at the dispatcher in combination with a power-of- $d$  policy. In [70] the authors study a policy with a memory of size  $m$ , where at every job arrival  $d$  servers are probed and the job is assigned to the server with the smallest number of pending jobs amongst the  $d$  probed servers and  $m$  servers in memory. The  $m$  servers with the shortest queue amongst the remaining  $d+m-1$  servers form the memory for the next job arrival. In [36] the authors study the amount of memory resp. probes required in order to obtain vanishing queueing delay. In [86, 8] policies are studied where the dispatcher maintains an upper bound on the queue length of each server and dispatches jobs based on these upper bounds.

The main insight obtained in this chapter is that studying a load balancing policy with memory using the cavity method, corresponds to studying the same load balancing policy without memory if we scale down the arrival rate in a proper manner (see also Theorem 84, 85, 86 and 92). For the LL( $d$ ) policy, we do not impose any restrictions on the job size distribution. For the SQ( $d$ ) policy, we initially restrict our attention to exponential job sizes and then generalize our main result to PH and general job size distributions.

As a by-product, our results allow us to study the Join-Idle-Queue policy (denoted by JIQ) with finite memory. JIQ exists in keeping track of the ids of the idle queues and assigning incoming jobs to an idle queue whenever there is an idle server in memory and simply assigning it to a random server otherwise. This policy has vanishing delays when the number of servers tends to infinity in case of infinite memory [60, 84, 35, 24].

Apart from the cavity method analysis, we additionally present explicit results for a critically loaded system by relying on the framework in Chapter 5 that allows one to compute the limit  $\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[R_\lambda]}{\log(1-\lambda)}$  for load balancing policies with exponential job sizes. We show that (unsurprisingly) for most memory schemes, the limiting value for  $\lambda \rightarrow 1^-$  remains unchanged when we add memory at the dispatcher. However, when the dispatcher has a memory of size  $A$  and servers inform the dispatcher when they become idle, the limiting value (for  $\lambda \rightarrow 1^-$ ) is multiplied by  $\frac{1}{A+1}$  for both the LL( $d$ ) and SQ( $d$ ) policy. In particular with a memory size of 1, the waiting time for a critically loaded system is halved compared to having no memory at the dispatcher.

Finally, we analyse the low load limit in case of exponential job sizes. In particular, we take a closer look at the ratio of the mean waiting time for two different load balancing policies as the load tends to zero, for which we find a simple closed form solution.

## 6.2 Structure of the chapter

This chapter is structured as follows. We start by giving an informal intuitive description of the analysis carried out in this chapter in Section 6.3. In Section 6.4, the model is introduced and we shortly review previously obtained results for  $SQ(d)$  and  $LL(d)$ . In Section 6.5 we present four approaches which make use of memory at the dispatcher and show how to obtain the probability that the memory is empty for each of these methods. Next we present our major analytical tool, the queue at the cavity in Section 7.6.1 and we describe how it is defined for the memory dependent  $LL(d)$  and  $SQ(d)$  policy. We carry out the formal analysis of the queue at the cavity in Section 6.7. Our analysis is verified by means of simulation in Section 6.8. In Section 6.9 we show how our results may be used for numerical experimentation by studying one specific setting. In Section 6.10 we study this system for a critically loaded system, while in Section 6.11 the a system with a small load is considered. We conclude the chapter in Section 6.12 and discuss possible future work.

## 6.3 Informal intuition

In this chapter, we assume there is one central dispatcher which assigns jobs to servers using a power of  $d$  choices algorithm. In this section, we restrict our attention to a memory based  $SQ(d)$  policy with exponential job sizes and a memory size equal to  $d - 1$ . There are many methods in which we can *fill* our memory, but the most straightforward method is to note that when we probe  $d$  servers and  $k \geq 2$  probed servers are idle, it would make sense to also remember the ids of the  $k - 1$  idle queues to which we do not assign a job.

The first observation, which is important to make, is the fact that memory size of the dispatcher and the queue length of the cavity queue evolve on a different timescale. We have arrivals to the dispatcher at a rate equal to

$\lambda \cdot N$  and each arrival has an impact on the memory. On the other hand, the potential arrival rate is some fixed value that does not depend on  $N$  and is bounded by  $\lambda \cdot d < \infty$ . As  $N \rightarrow \infty$ , it follows that between any two potential arrivals to the cavity queue, the memory state of the dispatcher has had an *unbounded* number of state transitions. Therefore, for each potential arrival, we may assume that the memory process is in its stationary state and its previous state does not impact the current state. All we really care about when a potential arrival occurs is whether or not the memory is empty. The analysis now breaks down into 2 parts, we need to analyse the Markov chain associated to the memory process (given the distribution of the queue length distribution) and we need to analyse the cavity queue (given the stationary distribution of the memory process). We introduce the following notation.

- We denote by  $\pi_k$ ,  $k = 0, \dots, d - 1$  the probability that there are 0, 1, 2 up to  $d - 1$  ids of idle queues in memory.
- We denote by  $u_k$  the probability that the cavity queue has  $k$  or more jobs in its queue.

Let us first consider the transition matrix associated to the memory process. Note that all we really need to know about the probed queues is whether or not they are empty. Due to the conservation of work, we find that (with  $\rho = \lambda \cdot \mathbb{E}[X]$ ) the probability that a queue is empty is given by  $1 - \rho$ . We therefore find that the probability that exactly  $\ell$  of the  $d$  probed servers are empty, is given by

$$\zeta_\ell = \binom{d}{\ell} \rho^{d-\ell} (1 - \rho)^\ell.$$

We therefore find that the transition matrix of the memory process is given by

$$M(\rho) = \begin{pmatrix} \zeta_0 + \zeta_1 & \zeta_2 & \zeta_3 & \dots & \zeta_{\ell-2} & \zeta_{\ell-1} & \zeta_\ell \\ 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & \ddots & 0 & 0 & 0 \\ 0 & 0 & 0 & \dots & 1 & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 & 0 \end{pmatrix}.$$

For this transition matrix one easily compute the stationary distribution  $\pi$  which satisfies  $\pi M(\rho) = \pi$ , in particular we find that (see Section 6.5 for

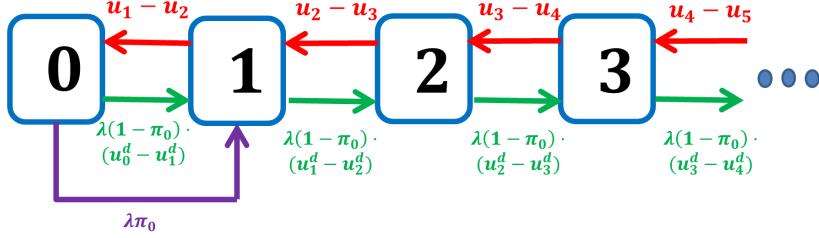


Figure 6.1: Graphical depiction of the transition rates for the memory based  $\text{SQ}(d)$  policy.

more details):

$$\pi_0 = \frac{1}{\rho^d + (1 - \rho)^d}.$$

This solves the first part of the problem, it remains to analyse the cavity queue, given the value  $\pi_0$ , which is the probability that an arrival is assigned to an empty queue (without making use of the  $\text{SQ}(d)$  policy).

In Figure 6.1, the squares represent the queue lengths for the cavity queue while the arrows between them represent the rate at which the cavity queue changes state. This may be compared to Figure 1.4 (where we make the same figure for  $\text{SQ}(d)$  without memory). We observe that we make a distinction between two types of arrivals:

- $\text{SQ}(d)$  arrivals, these transition rates are the same as the ones for the *normal*  $\text{SQ}(d)$  policy, except for the fact that we have to scale down the potential arrival rate from  $\lambda \cdot d$  to  $\lambda \cdot d \cdot \pi_0$ . Indeed, only the arrivals which find the memory to be empty cause  $\text{SQ}(d)$  arrivals.
- The arrivals which join an idle queue from memory. These only occur when the cavity queue is empty. The arrival rate for these type of arrivals is equal to  $\lambda \cdot (1 - \pi_0)$  as they only occur when the memory is non-empty. We then scale-up this arrival rate to  $\frac{\lambda \cdot (1 - \pi_0)}{1 - u_1}$  as these arrivals are always routed to empty servers. Lastly, we require the cavity queue to be idle in order to experience an arrival from memory, therefore we multiply by  $(1 - u_1)$ . In total this gives us the up-crossing rate depicted in purple in Figure 6.1.

We can now again employ a level-crossing argument to obtain the values of  $u_k$ , we find:

- The level crossing argument between queue length 0 and 1 yields:

$$u_1 - u_2 = \lambda(1 - \pi_0)(u_0^d - u_1^d) + \lambda\pi_0.$$

- The level crossing argument between  $k$  and  $k + 1$  for  $k \geq 1$  yields:

$$u_{k+1} - u_k = \lambda(1 - \pi_0)(u_k^2 - u_{k+1}^2).$$

Taking the sum for  $\ell \geq k$ , we find from the above equations:

$$\begin{aligned} u_1 &= \lambda(1 - \pi_0) + \lambda\pi_0 = \lambda \\ u_{k+1} &= \lambda(1 - \pi_0)u_k^d \quad \text{for } k \geq 1. \end{aligned}$$

From this, we find that:

$$u_k = \frac{(\rho\pi_0^{1/d})^{\frac{d^k-1}{d-1}}}{\pi_0^{1/d}}.$$

Note that if we want to compute the response time distribution, we find:

$$\begin{aligned} \bar{F}_R(w) &= (1 - \pi_0) \cdot \mathbb{P}\{E_1 \geq w\} + \pi_0 \sum_{k=0}^{\infty} (u_k^d - u_{k+1}^d) \mathbb{P}\{E_{k+1} \geq w\} \\ &= (1 - \lambda^d \pi_0) \mathbb{P}\{E_1 \geq w\} \\ &\quad + \sum_{k=1}^{\infty} \left( \left( (\rho\pi_0^{1/d})^{\frac{d^k-1}{d-1}} \right)^d - \left( (\rho\pi_0^{1/d})^{\frac{d^{k+1}-1}{d-1}} \right)^d \right) \cdot \mathbb{P}\{E_{k+1} \geq w\}, \end{aligned}$$

with  $E_k$  the sum of  $k$  i.i.d. exponential distributions. Plugging in the  $\pi_0$  which we computed previously allows us to conclude the analysis of the SQ( $d$ ) policy with Interrupted Probing. The fact that we get such a simple expression is pretty cool, but if we would compute the response time distribution for the SQ( $d$ ) policy with an arrival rate set equal to  $\lambda \cdot \pi_0^{1/d}$ , we would find exactly the same expression! That is: *The response time distribution for SQ( $d$ ) with a memory at the dispatcher is equal to the response time distribution for SQ( $d$ ) with an arrival rate equal to  $\lambda \cdot \pi_0^{1/d}$ .* Throughout this work we mainly generalize this result in the following directions:

- We consider many practical memory generation algorithms for which we can compute  $\pi_0$  in closed form.
- We consider generally distributed job sizes.
- Besides the SQ( $d$ ) policy, we also consider the LL( $d$ ) policy.

## 6.4 Model description

We consider a system consisting of  $N$  identical servers (with  $N$  large). There is some central dispatcher to which jobs arrive according to a  $\text{Poisson}(\lambda N)$  process. The dispatcher has some (finite or infinite) memory available to store ids of idle servers. When a job arrives and the dispatcher has the id(s) of some idle server(s) in its memory, the job is dispatched to a random server, the id of which is in memory. If the dispatcher's memory is empty,  $d$  servers are chosen at random and the job is either send to the server with the shortest queue ( $\text{SQ}(d)$ , see Section 6.4.1) or to the server with the least amount of work ( $\text{LL}(d)$ , see Section 6.4.2). Setting  $d = 1$  yields the JIQ policy where the job is simply routed arbitrarily whenever there are no idle servers known by the dispatcher. Before we proceed we provide some further details on the classic  $\text{SQ}(d)$  and  $\text{LL}(d)$  policy.

### 6.4.1 Classic $\text{SQ}(d)$

The  $\text{SQ}(d)$  policy was first introduced in [67, 95] for a system with  $\text{Poisson}(\lambda N)$  arrivals and exponential job sizes (with mean  $1/\mu$ ). Whenever a job arrives,  $d$  servers are probed at random and the incoming job is routed to the probed server with the least number of jobs in its queue. It was shown (see Section 1.5) that in the limit as  $N \rightarrow \infty$  the system behavior converges to the solution of the following set of ODEs:

$$\frac{d}{dt}u_k(t) = \lambda(u_{k-1}(t)^d - u_k(t)^d) - \mu(u_k(t) - u_{k+1}(t)),$$

where we denote by  $u_k(t)$  the probability that, at time  $t$ , an arbitrary server has at least  $k$  jobs in its queue and  $u_0(t) = 1$ . This set of ODEs also corresponds to applying the cavity method to the  $\text{SQ}(d)$  policy. The fixed point of this set of ODEs obeys a simple recursive formula:

$$\mu u_{k+1} = \lambda u_k^d, \tag{6.1}$$

which has the closed form solution  $u_k = \rho^{\frac{d^k-1}{d-1}}$  with  $\rho = \lambda/\mu$ . In particular one obtains from Little's Law the closed form solution of the expected response time:

$$\mathbb{E}[R] = \frac{1}{\lambda} \sum_{k=1}^{\infty} \rho^{\frac{d^k-1}{d-1}}. \tag{6.2}$$

### 6.4.2 Classic LL( $d$ )

The LL( $d$ ) policy was analysed in Chapter 2 for a system with arbitrary job sizes with mean  $\mathbb{E}[G]$  using the cavity method. Whenever a job arrives,  $d$  queues are probed and the job is sent to the queue which has the least amount of work left. This means that the job joins the queue at which its service can start the soonest. In practice this can be implemented through late binding, see also [74]. Let  $\bar{F}(w)$  denote the equilibrium probability that an arbitrary queue has at least  $w$  work left using the cavity method. It is shown in Chapter 2 that  $\bar{F}(w)$  satisfies the FPE:

$$\bar{F}(w) = \rho - \lambda \int_0^w (1 - \bar{F}(u)^d) \bar{G}(w-u) du, \quad (6.3)$$

with  $\rho = \lambda \mathbb{E}[G]$  and  $\bar{G}(w-u)$  the probability that a job has a size greater than  $w-u$ . This FPE can alternatively be written as the following DIDE:

$$\bar{F}'(w) = -\lambda \left[ \bar{G}(w) - \bar{F}(w)^d + \int_0^w \bar{F}(u)^d g(w-u) du \right],$$

with  $g$  the density function of the job size distribution. Both have the boundary condition  $\bar{F}(0) = \rho$ . Moreover, this equation has a closed form solution in case of exponential job sizes (with mean  $1/\mu$ ):

$$\bar{F}(w) = (\rho + (\rho^{1-d} - \rho) e^{(d-1)w})^{\frac{1}{1-d}}. \quad (6.4)$$

In particular, one obtains a closed form solution for the expected response time:

$$\mathbb{E}[R] = \frac{1}{\lambda} \sum_{n=0}^{\infty} \frac{\rho^{dn+1}}{1+n \cdot (d-1)}. \quad (6.5)$$

## 6.5 Discovering idle servers

We now discuss a number of approaches for the dispatcher to discover ids of idle servers. In the first few approaches the dispatcher discovers idle servers by probing, while in the last approach the idle servers identify themselves to the dispatcher. Note that as the amount of incoming work per server per unit of time is equal to  $\rho < 1$ , no work is replicated, and all servers are identical,

it follows that the steady state probability that a server is busy is given by  $\rho$ .<sup>1</sup>

### 6.5.1 Interrupted probing (IP)

In the first approach, called *interrupted* probing (IP), the dispatcher probes  $d$  servers when its memory is empty upon a job arrival. If there are  $k \geq 1$  idle servers among the  $d$  probed servers, it sends the incoming job to one of the idle servers and stores ids of the  $k - 1$  other servers in memory. These  $k - 1$  ids are then used for the subsequent  $k - 1$  arrivals. Thus for these  $k - 1$  arrivals, the dispatcher does not probe any servers. As  $\rho$  is the steady state probability that a server is busy, we can find the probability  $\pi_0$  of having no ids in memory when a new job arrives by looking at the Markov chain with state space  $0, \dots, d - 1$  and transition probability matrix  $M(\rho)$ :

$$\begin{aligned} M(\rho)_{0,0} &= \rho^d + \binom{d}{1} \rho^{d-1} (1-\rho), \\ M(\rho)_{0,\ell} &= \binom{d}{\ell+1} \rho^{d-1-\ell} (1-\rho)^{\ell+1}, \\ M(\rho)_{k,k-1} &= 1, \end{aligned}$$

and  $M(\rho)_{k,\ell} = 0$  otherwise.

As only the first row is non-trivial, it is not hard to check that  $\pi = (\pi_0, \dots, \pi_{d-1})$  given by:

$$\pi_k = \pi_0 \left[ 1 - \sum_{j=0}^k \binom{d}{j} \rho^{d-j} (1-\rho)^j \right],$$

for  $k \geq 1$  is an invariant vector of  $M(\rho)$ . From the requirement  $\sum_{k=0}^{d-1} \pi_k = 1$  it then follows that

$$\pi_0 = \frac{1}{\rho^d + (1-\rho)d}. \quad (6.6)$$

---

<sup>1</sup>For SQ( $d$ ) with exponential job sizes this is shown explicitly in the proof of Theorem 83, while for SQ( $d$ ) with general job sizes the proof is carried out in Proposition 87. For LL( $d$ ) this easily follows from integrating both sides of (6.32).

The number of probes used per arrival is clearly given by  $\pi_0 d$  which equals

$$\frac{1}{1 - \rho + \frac{\rho^d}{d}}.$$

The main advantage of the IP approach is that it uses far less than  $d$  probes per arrival when  $\rho$  is not too large (see also Figure 6.2b).

### 6.5.2 Continuous probing (CP)

This approach is similar to the IP approach, except that whenever we use a server id from memory for a job arrival, the dispatcher still probes  $d$  random servers. The ids of the  $d$  servers that are idle are subsequently added to memory. We assume that the available memory is unlimited.

In order to determine the probability  $\pi_0$  of having a server id in memory, we need to study a Markov chain on an infinite state space. Its transition probability matrix  $M(\rho)$  has the following form:

$$M(\rho)_{0,0} = \rho^d + \binom{d}{1} \rho^{d-1} (1-\rho),$$

$$M(\rho)_{0,\ell} = \binom{d}{\ell+1} \rho^{d-1-\ell} (1-\rho)^{\ell+1},$$

for  $1 \leq \ell \leq d-1$ . For  $k \geq 1$ , we have

$$M(\rho)_{k,k-1+\ell} = \binom{d}{\ell} \rho^{d-\ell} (1-\rho)^\ell,$$

for  $k-1 \leq \ell < d+k$ , and  $M(\rho)_{k,\ell} = 0$  otherwise. First note that if  $d > \frac{1}{1-\rho}$ , this Markov chain is transient as the drift in state  $k > 0$  is given by  $d(1-\rho) - 1$ , meaning after some point in time the chain never returns to state zero and all incoming arrivals can be assigned to an idle server. When  $d < \frac{1}{1-\rho}$ , the chain is positive recurrent and we need to determine  $\pi_0 < 1$ . A similar observation was made in [86, 8].

The average time the memory remains empty is equal to:

$$\frac{1}{1 - M(\rho)_{0,0}} = \frac{1}{1 - \rho^d - d\rho^{d-1}(1-\rho)}.$$

Furthermore, when the memory becomes non-empty, the length that it remains non-empty depends on the number of server ids that are placed into

memory. More specifically let  $\mathbb{E}[T_{k,0}]$  denote the expected first return time to 0 given that the chain starts in state  $k > 0$ , then:

$$\mathbb{E}[T_{k,0}] = k\mathbb{E}[T_{1,0}],$$

and the mean time that the Markov chain stays away from state 0 given that it just made a jump from state 0 to some state  $k > 0$  is given by  $\mathbb{E}[X_0]\mathbb{E}[T_{1,0}]$ , where  $\mathbb{E}[X_0]$  is one less than the mean number of idle servers among  $d$  servers given that at least 2 are idle. It is not hard to see that

$$\mathbb{E}[X_0] = \frac{d(1-\rho) - (1-\rho^d)}{1-\rho^d - d\rho^{d-1}(1-\rho)}.$$

Further,  $\mathbb{E}[T_{1,0}] = \frac{1}{1-d(1-\rho)}$  as  $\mathbb{E}[T_{1,0}] = 1 + d(1-\rho)\mathbb{E}[T_{1,0}]$ . Putting this together we obtain

$$\pi_0 = \frac{1 - d(1-\rho)}{\rho^d}, \quad (6.7)$$

when  $d < \frac{1}{1-\rho}$ . Note that the CP approach uses  $d$  probes per arrival.

### 6.5.3 Bounded continuous probing (BCP)

This approach is identical to the CP approach, but with finite memory size  $A$ . Hence the transition matrix  $M(\rho)$  is of size  $A + 1$  and its transitions are the same as in Section 6.5.2, except that any transition from a state  $k \leq A$  to a state  $\ell > A$  becomes a transition to state  $A$ . In particular for  $k > A - d + 1$  we have:

$$M(\rho)_{k,A} = \sum_{j=A-k+1}^d \binom{d}{j} \rho^{d-j} (1-\rho)^j,$$

and for all other values,  $M(\rho)$  coincides with the expressions given in Section 6.5.2. This Markov chain does not appear to have a simple closed form solution for arbitrary values of  $d$ , however for  $d = 2$  one finds:

$$\pi_0 = \frac{1 - \left(\frac{1-\rho}{\rho}\right)^2}{1 - \left(\frac{1-\rho}{\rho}\right)^{2(A+1)}}.$$

For  $d > 2$  a simple numerical scheme can be used to compute  $\pi_0$ . Note that this approach uses  $d$  probes per arrival unless the dispatcher sends the probes one at a time and stops probing when the memory is full.

### 6.5.4 Other probing schemes

In this section we present a result that applies to any scheme where the dispatcher discovers idle servers by probing and any idle server that is discovered is stored in memory. Thus the result only applies to BCP if the probes are sent one at a time.

**Proposition 81.** *Assume all discovered idle servers are stored in memory. Then for any  $LL(d)/SQ(d)$  memory based policy, the average number of probes used per arrival is given by:*

$$\frac{1 - \pi_0 \rho^d}{1 - \rho}. \quad (6.8)$$

*Proof.* If we think of the probes being transmitted one at a time and assigning the job as soon as an idle server is discovered, the dispatcher uses on average  $\sum_{k=0}^{d-1} \rho^k$  probes for any job arrival that occurs when the memory is empty. Further, for any arrival that uses an id in memory, an average of  $1/(1 - \rho)$  probes was used to discover that id. Hence, the average number of probes transmitted can be written as:

$$\pi_0 \frac{1 - \rho^d}{1 - \rho} + (1 - \pi_0) \frac{1}{1 - \rho}.$$

□

The above result indicates that for any such policy for which we either know the average number of probed queues (as for CP) or can express this using  $\pi_0$  (as for IP), we immediately obtain  $\pi_0$ . As the CP policy sends  $d$  probes per arrival and the IP policy  $\pi_0 d$ , Proposition 81 yields (6.6) and (6.7) without the need to analyse a Markov chain.

### 6.5.5 Idle server messaging (ISM)

In this scheme the dispatcher does not probe to discover idle servers, instead a server notifies the dispatcher whenever it becomes idle. In case of infinite memory, the dispatcher knows all idle server ids at all times and the system reduces to the JIQ policy when  $d = 1$ . Our interest lies mostly in knowing what happens when the memory size is finite and the job is assigned to the shortest of  $d$  queues whenever the memory is empty when a job arrives.

If we denote  $A$  as the number of ids that can be stored in memory, we show that  $\pi_0$  is given by

$$\pi_0 = \frac{1 - (1 - \rho^d)^{\frac{1}{A+1}}}{\rho^d}. \quad (6.9)$$

For  $SQ(d)$  this is shown in Proposition 88, while for  $LL(d)$  this is presented in Proposition 93. In particular, this result entails that  $\pi_0$  is insensitive to the job size distribution for  $SQ(d)$  and  $LL(d)$ .

If we assume that the  $d$  probes are transmitted one at a time when memory is empty and the dispatcher stops probing as soon as an idle server is discovered, the number of probes and messages transmitted by the dispatchers and servers per job arrival can be expressed as:

$$\pi_0 \frac{1 - \rho^d}{1 - \rho} + (1 - \pi_0 \rho^d),$$

where the first term corresponds to the number of probes send per arrival by the dispatcher and the second correspond to the number of server messages per arrival (which is equal to the probability that a job is assigned to an idle server).

## 6.6 Description of the queue at the cavity

Our analysis is based on the *queue at the cavity* method which was introduced in [20] to analyse load balancing systems. The key idea is to focus on the evolution of a single tagged queue, referred to as the queue at the cavity, and to assume that all other queues have the same queue length (or workload) distribution at any time  $t$ . Moreover the queue length (or workload) of any finite set of queues is assumed to be independent at any time  $t$ . We first explain the approach in a system without memory and then indicate how to adapt it to incorporate memory.

In a system without memory, the queue at the cavity experiences potential arrivals at rate  $\lambda d$  as this is the rate at which a tagged queue is selected as one of the  $d$  randomly selected queues. If a potential arrival occurs at time  $t$ ,  $d - 1$  i.i.d. random variables are initialized which have the same queue length (or workload) distribution as the queue at the cavity at time  $t$ . The potential arrival becomes an actual arrival if the queue at the cavity has the shortest

queue (or smallest workload) amongst these  $d$  values (where ties are broken at random). For  $SQ(d)$  with exponential job sizes with mean  $1/\mu$  the queue length of the queue at the cavity decreases at a constant rate equal to  $\mu$  in between potential arrivals, while for  $LL(d)$  the workload decreases linearly at rate 1 when larger than zero. For PH distributed job sizes, one needs to include the phase of the job at the head of the queue, while for general job sizes we need to include the work left for the job at the head of the queue.

To incorporate memory into the cavity method we note that the state of the memory (that is, the number of ids that it contains) evolves at a faster time scale than the fraction of queues with a certain queue length (or workload). As such the state of the memory at time  $t$  is given by the steady state  $\pi(t)$  of the discrete time Markov chain with transition matrix  $M(\rho(t))$  that captures the evolution of the memory, where  $\rho(t)$  is the fraction of busy servers at time  $t$  (see Section 6.5 for some examples with  $\rho(t) = \rho$ ). For more details on the concept of the time-scale separation we employ, we refer the reader to [16].

Let  $\pi_0(t)$ , the first entry of  $\pi(t)$ , represent the probability that the memory is empty at time  $t$ . We modify the queue at the cavity by decreasing the potential arrival rate to the queue at the cavity to  $\lambda d\pi_0(t)$ , i.e. potential arrivals only occur when there is no empty queue to join in memory. These potential arrivals are then dealt with in the exact same manner as in the setting without memory. When the queue at the cavity is empty, we assume that on top of the potential arrival rate of  $\lambda d\pi_0(t)$ , we have an effective arrival rate of  $\lambda \frac{1-\pi_0(t)}{1-\rho(t)}$ . The latter arrival rate can be interpreted as follows: jobs arrive at rate  $\lambda N$ , with probability  $(1 - \pi_0(t))$  such a job is assigned to a queue in memory and with probability  $1/((1 - \rho(t))N)$  the queue at the cavity gets the job as it is one of the  $(1 - \rho(t))N$  idle servers at time  $t$ .

In the next section we study the cavity process of  $SQ(d)$  and  $LL(d)$  with memory in detail. We assume job sizes have some general distribution with probability density function (pdf)  $g$ , cumulative distribution function (cdf)  $G$  and complementary cdf (ccdf)  $\bar{G}$ . For a random variable with cdf  $H$  we let  $\mathbb{E}[H]$  denote its mean. Let  $\mu = \frac{1}{\mathbb{E}[G]}$  denote the mean service rate and note that we have for the system load:  $\rho = \lambda \cdot \mathbb{E}[G]$ . Furthermore we let  $\mathcal{G}$  denote a generic random variable with distribution  $G$ . We will sometimes assume that  $\mathcal{G}$  is an exponential random variable. Furthermore, for  $LL(d)$  we denote by  $f, F$  and  $\bar{F}$  the pdf, cdf and ccdf of the workload distribution of the queue at the cavity in equilibrium (note that we have  $\bar{F}(0) = \rho$ ). For  $SQ(d)$  with

exponential job sizes we denote by  $u_k$  the equilibrium probability that the queue at the cavity has  $k$  or more jobs (with  $u_0 = 1$  and  $u_1 = \rho$ ).

## 6.7 Analysis of the queue at the cavity

We now analyse the queue at the cavity described in the previous section for  $SQ(d)$  and  $LL(d)$ . Note that the results presented in this section apply to any of the memory schemes discussed in Section 6.5. To obtain results for a specific memory scheme one simply replaces  $\pi_0$  by the appropriate expression. We show that the equilibrium queue length and workload distribution of  $SQ(d)$  and  $LL(d)$  with memory, respectively, have exactly the same form as in the same setting without memory if we replace  $\lambda$  by  $\lambda\pi_0^{1/d}$  and divide by  $\pi_0^{1/d}$ . With respect to the response time distribution, we show that the system with memory and arrival rate  $\lambda$  has the same response time distribution as the system without memory and arrival rate  $\lambda\pi_0^{1/d}$ .

### 6.7.1 $SQ(d)$

In this section we develop the analysis of the queue at the cavity for the  $SQ(d)$  policy, we start by assuming job sizes are exponential and subsequently we consider Phase Type and general job sizes.

#### Exponential Job Sizes

We start by describing the transient behaviour of the queue at the cavity for  $SQ(d)$ :

**Proposition 82.** *Consider the  $SQ(d)$  policy with memory, exponential job sizes with mean  $1/\mu$  and arrival rate  $\lambda < \mu$ . Let  $u_k(t)$  be the probability that the queue at the cavity has  $k$  or more jobs at time  $t$ , then*

$$\frac{d}{dt}u_k(t) = \lambda\pi_0(t)(u_{k-1}(t)^d - u_k(t)^d) - \mu(u_k(t) - u_{k+1}(t)), \quad (6.10)$$

$$\frac{d}{dt}u_1(t) = \lambda\pi_0(t)(u_0(t)^d - u_1(t)^d) + \lambda(1 - \pi_0(t)) - \mu(u_1(t) - u_2(t)), \quad (6.11)$$

for  $k \geq 2$  and  $u_0(t) = 1$ .

*Proof.* Let  $\Delta > 0$  be arbitrary, we first assume that  $k \geq 2$  and consider the cases in which the queue at the cavity may have  $k$  or more jobs at time  $t + \Delta$ . First, it may have exactly  $k$  jobs at time  $t$  and no departures occur in  $[t, t + \Delta]$ , this occurs with probability:

$$Q_{1,k} = (1 - \mu\Delta)(u_k(t) - u_{k+1}(t)) + o(\Delta). \quad (6.12)$$

It may also have  $k + 1$  or more jobs at time  $t$ , and at most 1 departure occurs in  $[t, t + \Delta]$ :

$$Q_{2,k} = u_{k+1}(t) + o(\Delta). \quad (6.13)$$

A third possibility is that it had exactly  $k - 1$  jobs at time  $t$  and exactly one arrival occurs in  $[t, t + \Delta]$  which joined the queue at the cavity, this occurs with probability:

$$Q_{3,k} = \lambda d \left( \int_0^\Delta \pi_0(t + \delta) d\delta \right) (u_{k-1}(t) - u_k(t)) \quad (6.14)$$

$$\begin{aligned} & \sum_{j=0}^{d-1} \frac{1}{j+1} \binom{d-1}{j} (u_{k-1}(t) - u_k(t))^j \cdot u_k(t)^{d-j} + o(\Delta) \\ &= \lambda \left( \int_0^\Delta \pi_0(t + \delta) d\delta \right) (u_{k-1}(t)^d - u_k(t)^d) + o(\Delta). \end{aligned} \quad (6.15)$$

We now obtain:

$$u_k(t + \Delta) = Q_{1,k} + Q_{2,k} + Q_{3,k},$$

subtracting  $u_k(t)$  on both sides, dividing by  $\Delta$  and taking the limit  $\Delta \rightarrow 0$  yields (6.10). For (6.11), one needs to consider the same  $Q_{1,k}$ ,  $Q_{2,k}$  and  $Q_{3,k}$  as for  $k \geq 2$  for the case of potential arrivals. There is however an additional term for the case where the queue at the cavity is empty at time  $t$  and it experiences an arrival due to the memory induced arrival rate, this yields:

$$Q_{4,1} = \lambda \left( \int_0^\Delta \frac{1 - \pi_0(t + \delta)}{u_0(t + \delta) - u_1(t + \delta)} d\delta \right) (u_0(t) - u_1(t)) + o(\Delta),$$

one then obtains  $u_1(t + \Delta) = Q_{1,1} + Q_{2,1} + Q_{3,1} + Q_{4,1}$ , subtracting  $u_1(t)$ , dividing both sides by  $\Delta$  and taking the limit  $\Delta \rightarrow 0$  yields (6.11). Finally the last equation  $u_0(t) = 1$  is trivial by the definition of  $u_0(t)$ .  $\square$

From the transient regime, we are able to deduce the equilibrium workload distribution:

**Theorem 83.** Consider the  $SQ(d)$  policy with memory, exponential job sizes with mean  $1/\mu$  and arrival rate  $\lambda < \mu$ . Let  $u_k$  be the equilibrium probability that the queue at the cavity has  $k$  or more jobs, then

$$u_k = \rho^{\frac{d^k - 1}{d-1}} \cdot \pi_0^{\frac{d^{k-1} - 1}{d-1}} = (\rho \pi_0^{1/d})^{\frac{d^k - 1}{d-1}} / \pi_0^{1/d}, \quad (6.16)$$

for  $k \geq 1$  and  $\rho = \lambda/\mu$ .

*Proof.* Taking the limit of  $t \rightarrow \infty$  in (6.10-6.11) we find that the following holds:

$$\begin{aligned} 0 &= \lambda \pi_0 (u_0^d - u_1^d) + \frac{\lambda(1 - \pi_0)}{1 - \rho} \cdot (u_0 - u_1) - \mu \cdot (u_1 - u_2), \\ 0 &= \lambda \pi_0 (u_{k-1}^d - u_k^d) - \mu \cdot (u_k - u_{k+1}), \end{aligned}$$

for  $k \geq 2$ . Summing all of these equations yields  $u_1 = \rho$ , while taking the sum for  $k \geq j$  implies that  $u_j = \lambda \pi_0 u_{j-1}^d$  for  $j \geq 2$ . This simple recurrence relation has (6.16) as its unique solution.  $\square$

Comparing (6.16) with the solution of (6.1), we see that  $u_k$  is identical as in the setting without memory if we replace  $\rho$  by  $\rho \pi_0^{1/d}$  and divide by  $\pi_0^{1/d}$  (even for  $k = 1$ ).

**Theorem 84.** Let  $0 < \lambda < \mu$  be arbitrary and  $R$  the response time of the  $SQ(d)$  policy with memory, exponential job sizes with mean  $1/\mu$  and arrival rate  $\lambda$ . Further, let  $\tilde{R}$  denote the response time for the same system without memory, but with arrival rate  $\lambda \pi_0^{1/d}$ , then  $\tilde{R}$  and  $R$  have the same distribution.

*Proof.* Let us denote by  $u_k$  and  $v_k$  the probability that the queue at the cavity has at least  $k$  jobs for the system with and without memory, respectively. We have  $u_k = \pi_0^{-1/d} \cdot v_k$ , for  $k \geq 1$  and  $u_0 = v_0 = 1$ . Let  $\bar{F}_X$  be the ccdf of  $X$ , then

$$\bar{F}_R(w) = (1 - \pi_0) e^{-\mu w} + \pi_0 \sum_{k=0}^{\infty} (u_k^d - u_{k+1}^d) \sum_{n=0}^k \frac{w^n}{n!} e^{-\mu w},$$

as with probability  $(1 - \pi_0)$  the job joins an idle queue from memory (meaning the response time is simply exponential) and with probability  $\pi_0(u_k^d - u_{k+1}^d)$  the job joins a queue with length  $k$  (yielding an Erlang  $k+1$  response time).

Exchanging the order of the sums and using  $\pi_0 u_k^d = v_k^d$ , for  $k \geq 1$ , implies that

$$\bar{F}_R(w) = (1 - \pi_0)e^{-\mu w} + \sum_{n=1}^{\infty} \frac{w^n}{n!} e^{-\mu w} v_n^d + \pi_0 e^{-\mu w} = \sum_{n=0}^{\infty} \frac{w^n}{n!} e^{-\mu w} v_n^d.$$

Similarly,

$$\bar{F}_{\tilde{R}}(w) = \sum_{k=0}^{\infty} (v_k^d - v_{k+1}^d) \sum_{n=0}^k \frac{w^n}{n!} e^{-\mu w} = \sum_{n=0}^{\infty} \frac{w^n}{n!} e^{-\mu w} v_n^d.$$

□

### Phase Type Job Sizes

PH distributions consist of all distributions which have a modulating finite background Markov chain (see also [56]). They form a broad spectrum of distributions as any positive valued distribution can be approximated arbitrarily close by a PH distribution. Moreover, various fitting tools are available online for PH distributions (e.g. [53, 75]). A PH distribution with  $\bar{G}(0) = 1$  is fully characterized by a stochastic vector  $\alpha = (\alpha_i)_{i=1}^n$  and a subgenerator matrix  $A = (a_{i,j})_{i,j=1}^n$  such that  $\bar{G}(w) = \alpha e^{Aw} \mathbf{1}$ , where  $\mathbf{1}$  is a column vector of ones.

We find that the result found in Theorem 84 generalizes to the case of PH distributed job sizes.

**Theorem 85.** *Let  $0 < \lambda < \mu$  (with  $1/\mu$  the mean of the job size distribution) be arbitrary and  $R$  the response time for a memory dependent version of the  $SQ(d)$  policy with PH distributed job sizes with parameters  $(\alpha, A)$ . Further, let  $\tilde{R}$  denote the response time for the classic  $SQ(d)$  policy with the same job size distribution and arrival rate  $\lambda \pi_0^{1/d}$ , then  $R$  and  $\tilde{R}$  have the same distribution.*

*Proof.* Let us denote by  $u_{k,j}(t)$  resp.  $v_{k,j}(t)$  the probability that, at time  $t$ , the queue at the cavity has at least  $k$  jobs and the job at the head of the queue is in phase  $j$  for the memory dependent scheme resp. the memory independent scheme. Furthermore let  $u_{k,j}$  and  $v_{k,j}$  denote the limit of  $t \rightarrow \infty$  for these values. We first show that  $u_{k,j} = \pi_0^{-1/d} \cdot v_{k,j}$ . Throughout we let  $\nu = -A\mathbf{1}$

(with  $\mathbf{1}$  a vector consisting of only ones). For  $v_{k,j}$  we find by an analogous reasoning as in [92] that for  $k \geq 2$ :

$$\begin{aligned} \frac{d}{dt}v_{k,j}(t) &= \lambda\pi_0(t)^{1/d}\frac{v_{k-1,j}(t) - v_{k,j}(t)}{v_{k-1}(t) - v_k(t)}(v_{k-1}(t)^d - v_k(t)^d \\ &\quad + \sum_{j'}(v_{k,j'}(t)A_{j',j} + v_{k+1,j'}(t)\nu_{j'}\alpha_j), \end{aligned} \quad (6.17)$$

where  $v_k(t)$  denotes  $\sum_j v_{k,j}(t)$  (further on, we also use this notation for  $v_k, u_k(t)$  and  $u_k$ ). For  $k = 1$  we find:

$$\frac{d}{dt}v_{1,j}(t) = \alpha_j\lambda\pi_0(t)^{1/d}(1 - v_1(t)^d) + \sum_{j'}(v_{1,j'}(t)A_{j',j} + v_{2,j'}(t)\nu_{j'}\alpha_j). \quad (6.18)$$

Taking the limit of  $t$  to infinity and multiplying by  $\pi_0^{-1/d}$  we find that (6.17) yields for the equilibrium distribution (with  $k \geq 2$ ):

$$\begin{aligned} 0 &= \pi_0\lambda\frac{(\pi_0^{-1/d}v_{k-1,j}) - (\pi_0^{-1/d}v_{k,j})}{(\pi_0^{-1/d}v_{k-1}) - (\pi_0^{-1/d}v_k)} \cdot \left((\pi_0^{-1/d}v_{k-1})^d - (\pi_0^{-1/d}v_k)^d\right) \\ &\quad + \sum_{j'}(\pi_0^{-1/d}v_{k,j'})A_{j',j} + (\pi_0^{-1/d}v_{k+1,j'})\nu_{j'}\alpha_j. \end{aligned} \quad (6.19)$$

while for  $k = 1$  one may compute from (6.18):

$$\begin{aligned} 0 &= \alpha_j\lambda\left(1 - \pi_0(\pi_0^{-1/d}v_1)^d\right) \\ &\quad + \sum_{j'}\left((\pi_0^{-1/d}v_{1,j'})A_{j',j} + (\pi_0^{-1/d}v_{2,j'})\nu_{j'}\alpha_j\right) \end{aligned} \quad (6.20)$$

For  $(u_{k,j}(t))$  with  $k \geq 2$ , we find the same ODE as (6.17) but with  $\lambda\pi_0(t)$  rather than  $\lambda\pi_0^{1/d}(t)$ . Taking the limit  $t \rightarrow \infty$  it is not hard to see that  $u_{k,j}$  satisfies (6.19) with  $\pi_0^{-1/d}v_{k,j}$  replaced by  $u_{k,j}$ . Furthermore for  $u_{1,j}(t)$  we find (similar to Proposition 82):

$$\begin{aligned} \frac{d}{dt}u_{1,j}(t) &= \lambda\alpha_j\pi_0(t)(1 - u_1(t)^d) + \lambda\alpha_j(1 - \pi_0(t)) \\ &\quad + \sum_{j'}(u_{1,j'}(t)A_{j',j} + u_{2,j'}(t)\nu_{j'}\alpha_j). \end{aligned}$$

Taking  $t \rightarrow \infty$  it is not hard to see how this equation for  $u_{k,j}$  reduces to (6.20) with  $\pi_0^{-1/d} v_{k,j}$  replaced by  $u_{k,j}$ . This shows that we indeed have for all  $k$  and  $j$  that  $u_{k,j} = \pi_0^{-1/d} v_{k,j}$ .

For the response time distribution we denote by  $X_{k,j}$  the response time of a job that joins a queue with length  $k$  in phase  $j$ . We find for the memory dependent policy:

$$\begin{aligned}\bar{F}_R(w) &= (1 - \pi_0)\bar{G}(w) + \pi_0 \left( (1 - u_1^d)\bar{G}(w) \right. \\ &\quad \left. + \sum_{k=1}^{\infty} \sum_j \frac{u_{k,j} - u_{k+1,j}}{u_k - u_{k+1}} \cdot (u_k^d - u_{k+1}^d) \mathbb{P}\{X_{k,j} > w\} \right) \\ &= (1 - (\pi_0^{1/d} u_1)^d)\bar{G}(w) \\ &\quad + \sum_{k=1}^{\infty} \sum_j \frac{\pi_0^{1/d} u_{k,j} - \pi_0^{1/d} u_{k+1,j}}{\pi_0^{1/d} u_k - \pi_0^{1/d} u_{k+1}} \left( (\pi_0^{1/d} u_k)^d - (\pi_0^{1/d} u_{k+1})^d \right) \mathbb{P}\{X_{k,j} > w\}\end{aligned}$$

One can now easily check that  $R$  and  $\tilde{R}$  indeed coincide.  $\square$

## General Job Sizes

We further generalize the results given in section 6.7.1 to the case of general job sizes. In particular we show the following result :

**Theorem 86.** *Let  $0 < \lambda < \mu$  (with  $1/\mu$  the mean of the job size distribution) be arbitrary and  $R$  the response time for a memory dependent version of the  $SQ(d)$  policy with an arbitrary job size distribution. Further, let  $\tilde{R}$  denote the response time for the classic  $SQ(d)$  policy with the same job size distribution and arrival rate  $\lambda\pi_0^{1/d}$ , then  $R$  and  $\tilde{R}$  have the same distribution.*

*Proof.* Let us denote by  $x_k(t, w)$  resp.  $y_k(t, w)$  the density at which, at time  $t$ , the queue at the cavity has exactly  $k$  jobs and the job at the head of the queue has a remaining size exactly equal to  $w$  for the memory dependent scheme resp. the memory independent scheme. Associated to these values, we denote  $u_k(t) = \int_0^\infty \sum_{\ell \geq k} x_\ell(t, w) dw$  and  $v_k(t) = \int_0^\infty \sum_{\ell \geq k} y_\ell(t, w) dw$ . Furthermore let  $x_k(w), y_k(w)$  and  $u_k, v_k$  denote the limit of  $t \rightarrow \infty$  for these values. We first show that  $x_k(w) = \pi_0^{-1/d} \cdot y_k(w)$  (and consequently also  $u_k = \pi_0^{-1/d} \cdot v_k$ ).

Let us first consider  $x_k(t, w)$  for  $k \geq 2$ . Analogously to the proof for exponential and PH job sizes, we obtain:

$$\begin{aligned} x_k(t + \Delta, w) &= x_k(t, w + \Delta) - \lambda dx_k(t, w + \Delta) \int_0^\Delta \pi_0(t + \delta) \cdot \sum_{j=0}^{d-1} \frac{1}{j+1} \binom{d-1}{j} \\ &\quad (u_k(t + \delta) - u_{k+1}(t + \delta))^j u_{k+1}(t + \delta)^{d-j-1} d\delta + \lambda dx_{k-1}(t, w + \Delta) \\ &\quad \int_0^\Delta \pi_0(t + \delta) \sum_{j=0}^{d-1} \frac{1}{j+1} \binom{d-1}{j} (u_{k-1}(t + \delta) - u_k(t + \delta))^j \\ &\quad u_{k+1}(t + \delta)^{d-j-1} d\delta + \int_0^\Delta x_{k+1}(t, \delta) g(w + \Delta - \delta) d\delta + o(\Delta). \end{aligned}$$

Subtracting  $x_k(t, w)$  on both sides, dividing both sides by  $\Delta$  and taking the limit  $\Delta \rightarrow 0^+$  we obtain the following system of PDEs:

$$\begin{aligned} \frac{\partial x_k(t, w)}{\partial t} - \frac{\partial x_k(t, w)}{\partial w} &= -\lambda \pi_0(t) \frac{x_k(t, w)}{x_k(t)} (u_k(t)^d - u_{k+1}(t)^d) \\ &\quad + \lambda \pi_0(t) \frac{x_{k-1}(t, w)}{x_{k-1}(t)} (u_{k-1}(t)^d - u_k(t)^d) + x_{k+1}(0^+) g(w). \end{aligned}$$

Taking the limit of  $t \rightarrow \infty$  we obtain:

$$x'_k(w) = \lambda \pi_0 \frac{x_k(w)}{x_k} (u_k^d - u_{k+1}^d) - \lambda \pi_0 \frac{x_{k-1}(w)}{x_{k-1}} (u_{k-1}^d - u_k^d) - x_{k+1}(0^+) g(w). \quad (6.21)$$

A differential equation for the system without memory can be inferred from (6.21) by setting  $\pi_0 = 1$  and replacing  $\lambda$  by  $\lambda \pi_0^{1/d}$ :

$$y'_k(w) = \lambda \pi_0^{1/d} \frac{y_k(w)}{y_k} (v_k^d - v_{k+1}^d) - \lambda \pi_0^{1/d} \frac{y_{k-1}(w)}{y_{k-1}} (v_{k-1}^d - v_k^d) - y_{k+1}(0^+) g(w).$$

Multiplying both sides by  $\pi_0^{-1/d}$ , we find that  $y_k$  satisfies the following (for  $k \geq 2$ ):

$$\begin{aligned} (\pi_0^{-1/d} y_k(w))' &= \lambda \pi_0 \frac{\pi_0^{-1/d} y_k(w)}{\pi_0^{-1/d} y_k} ((\pi_0^{-1/d} v_k)^d - (\pi_0^{-1/d} v_{k-1})^d) - \lambda \pi_0 \frac{\pi_0^{-1/d} y_{k-1}(w)}{\pi_0^{-1/d} y_{k-1}} \\ &\quad ((\pi_0^{-1/d} v_{k-1})^d - (\pi_0^{-1/d} v_k)^d) - (\pi_0^{-1/d} y_{k+1}(0^+)) g(w), \end{aligned}$$

which is identical to (6.21) if we replace  $x_k$  with  $\pi_0^{-1/d} y_k$ .

It remains to look at the case  $k = 1$ . For this case, the arrivals we need to consider are those which occur when the queue at the cavity is empty. Therefore, we need to consider two types of arrivals: those which occur due to the fact that the queue at the cavity is in the memory and those which occur due to the queue at the cavity being selected by the SQ( $d$ ) policy. For the arrivals incurred by the memory we find :

$$\lim_{t \rightarrow \infty} \lim_{\Delta \rightarrow 0^+} \lambda \frac{1}{\Delta} \int_0^\Delta (1 - \pi_0(t + \delta)) g(w + \Delta - \delta) d\delta + \frac{o(\Delta)}{\Delta} = \lambda(1 - \pi_0)g(w).$$

The arrivals incurred from the SQ( $d$ ) policy are similar to the case  $k \geq 2$ , we obtain that  $x'_1(w)$  satisfies:

$$x'_1(w) = \lambda \pi_0 \frac{x_1(w)}{x_1} (u_1^d - u_2^d) - \lambda \pi_0 (1 - u_1^d) g(w) - x_2(0^+) g(w) - \lambda(1 - \pi_0) g(w). \quad (6.22)$$

For the system without memory we replace  $\pi_0$  by 1 and  $\lambda$  by  $\lambda \pi_0^{1/d}$ . If we then multiply both sides by  $\pi_0^{-1/d}$ , we obtain:

$$\begin{aligned} (\pi_0^{-1/d} y_1(w))' &= \lambda \pi_0 \frac{\pi_0^{-1/d} y_1(w)}{\pi_0^{-1/d} y_1} ((\pi_0^{-1/d} v_1)^d - (\pi_0^{-1/d} v_2)^d) \\ &\quad - \lambda g(w) + \lambda \pi_0 g(w) (\pi_0^{-1/d} v_1)^d - (\pi_0^{-1/d} y_2(0^+)) g(w). \end{aligned} \quad (6.23)$$

It is not hard to see that (6.22) and (6.23) are equivalent (with  $x_k$  replaced by  $\pi_0^{-1/d} y_k$ ). This shows that we indeed have  $x_k(w) = \pi_0^{-1/d} y_k(w)$  for all  $k \geq 1$  and  $w \geq 0$ .

For the response time distribution of the system with memory, we obtain:

$$\begin{aligned} \bar{F}_R(w) &= (1 - \pi_0) \bar{G}(w) \\ &\quad + \pi_0 (1 - x_0^d) \bar{G}(w) + \pi_0 \sum_{k=1}^{\infty} \int_0^w \frac{x_k(s)}{x_k} (u_k^d - u_{k+1}^d) \mathbb{P}\{\mathcal{G}^{*k} > w - s\} ds \\ &= (1 - (\pi_0^{1/d} x_0)^d) \bar{G}(w) \\ &\quad + \sum_{k=1}^{\infty} \int_0^w \frac{x_k(s)}{x_k} ((\pi_0^{1/d} u_k)^d - (\pi_0^{1/d} u_{k+1})^d) \mathbb{P}\{\mathcal{G}^{*k} > w - s\} ds. \end{aligned}$$

Analogously one can compute  $\bar{F}_{\tilde{R}}$  to complete the proof.  $\square$

**Remark 51.** When  $d = 1$  in Theorem 86, the system without memory reduces to an ordinary  $M/G/1$  queue with arrival rate  $\lambda\pi_0^{1/d}$  for which many results exist. In particular, we find from the Pollaczek-Khinchin formula that the following holds:

$$R^*(w) = \frac{(1 - \pi_0^{1/d})\rho\mathcal{G}^*(w)w}{\pi_0^{1/d}\lambda\mathcal{G}^*(w) + w - \pi_0^{1/d}\lambda} \quad (6.24)$$

with  $R^*$  and  $\mathcal{G}^*$  the Laplace transform of  $R$  and  $\mathcal{G}$ , respectively. Using the ISM scheme presented in Section 6.5.5, this allows one to analyse the JIQ policy with finite memory by plugging  $\pi_0 = \frac{1-(1-\rho)^{\frac{1}{A+1}}}{\rho}$  into (6.24) (see also Proposition 88).

Using the ideas in Theorem 86 we are able to show that  $u_1 = \rho$  holds:

**Proposition 87.** For the memory dependent  $SQ(d)$  policy with general job sizes we have  $u_1 = \rho$ .

*Proof.* We use the same notation as in the proof of Theorem 86. Furthermore, we denote  $\tilde{x}_k(w) = \int_w^\infty x_k(u) du$ . We now wish to show that  $u_1 = \sum_{k=1}^\infty \int_0^\infty x_k(w) dw = \rho$ .

Integrating (6.22) from  $w$  to  $\infty$ , we find:

$$\begin{aligned} x_1(w) &= -\lambda\pi_0 \frac{\tilde{x}_1(w)}{x_1} (u_1^d - u_2^d) \\ &\quad + \lambda\pi_0(1 - u_1^d)\bar{G}(w) + x_2(0^+)\bar{G}(w) + \lambda(1 - \pi_0)\bar{G}(w). \end{aligned} \quad (6.25)$$

For  $k \geq 2$  we find from (6.21) that (integrate from  $w$  to infinity):

$$x_k(w) = -\lambda\pi_0 \frac{\tilde{x}_k(w)}{x_k} (u_k^d - u_{k+1}^d) + \lambda\pi_0 \frac{\tilde{x}_{k-1}(w)}{x_{k-1}} (u_{k-1}^d - u_k^d) + x_{k+1}(0^+)\bar{G}(w). \quad (6.26)$$

It is now easy to see from taking the sum of (6.25) and (6.26) (for all  $k \geq 2$ ) that for any  $w$ :

$$u_1(w) = \lambda(1 - \pi_0 u_1^d)\bar{G}(w) + u_2(0^+)\bar{G}(w). \quad (6.27)$$

Integrating this expression from 0 to infinity, we obtain:

$$u_1 = (u_2(0^+) + \lambda(1 - \pi_0 u_1^d)) \mathbb{E}[G]. \quad (6.28)$$

Furthermore, it is not hard to see that we have for any  $k \geq 2$ :

$$\begin{aligned} u_k(t + \Delta) &= \int_0^\Delta (u_k(t + \delta) - x_k(t + \delta, \Delta - \delta)) d\delta \\ &\quad + \lambda \int_0^\Delta \pi_0(t + \delta) (u_{k-1}(t + \delta)^d - u_k(t + \delta)^d) d\delta + o(\Delta), \\ u'_k(t) &= -x_k(t, 0^+) + \lambda \pi_0(t) (u_{k-1}^d(t) - u_k(t)^d), \end{aligned}$$

letting  $t \rightarrow \infty$  this leads to:

$$0 = -x_k(0^+) + \lambda \pi_0(u_{k-1}^d - u_k^d).$$

Taking the sum of these equations for  $k \geq 2$  we obtain:

$$u_2(0^+) = \lambda \pi_0 u_1^d.$$

Using this allows us to conclude that  $u_1 = \lambda \mathbb{E}[G] = \rho$  from (6.28).  $\square$

In the following Proposition, we obtain  $\pi_0$  for the ISM memory scheme presented in Section 6.5.5.

**Proposition 88.** *For the SQ( $d$ ) policy with general job sizes and the ISM memory scheme presented in Section 6.5.5 we have*

$$\pi_0 = \frac{1 - (1 - \rho^d)^{\frac{1}{A+1}}}{\rho^d}.$$

*Proof.* We use the same notation as in the proof of Proposition 87. The rate at which servers send probes is equal to  $x_1(0^+)$  (which is equal to the rate at which servers become idle). Therefore, the memory state evolves as a birth-death process with birth rate  $x_1(0^+)$  and death rate  $\lambda$ . From taking the limit  $w \rightarrow 0^+$  in (6.27), we find that  $x_1(0^+) = \lambda(1 - \pi_0 u_1^d)$ .

We consequently find that due to the birth-death structure:

$$\pi_0 = \frac{1}{\sum_{i=0}^A (1 - \pi_0 \rho^d)^i} = \frac{\pi_0 \rho^d}{1 - (1 - \pi_0 \rho^d)^{A+1}}.$$

From this one easily completes the proof.  $\square$

In particular, Proposition 88 holds for  $d = 1$ , which provides a closed form of  $\pi_0$  for JIQ with finite memory size.

### 6.7.2 LL( $d$ )

For LL( $d$ ), we again start by describing the transient regime (the proof is similar to the one presented in Chapter 2).

**Proposition 89.** *The density of the cavity process associated to the memory dependent LL( $d$ ) policy satisfies the following Partial Integro Differential Equations (PIDEs):*

$$\begin{aligned} \frac{\partial f(t, w)}{\partial t} - \frac{\partial f(t, w)}{\partial w} &= \lambda d\pi_0(t) \int_0^w f(t, u)\bar{F}(t, u)^{d-1}g(w-u)du \\ &\quad + \lambda\pi_0(t)(1-\bar{F}(t, 0)^d)g(w) - \lambda d\pi_0(t)f(t, w)\bar{F}(t, w)^{d-1} \\ &\quad + \lambda(1-\pi_0(t))g(w) \end{aligned} \tag{6.29}$$

$$\frac{\partial \bar{F}(t, 0)}{\partial t} = -f(t, 0^+) + \lambda\pi_0(t)(1-\bar{F}(t, w)^d) + \lambda(1-\pi_0(t)), \tag{6.30}$$

for  $w > 0$ , where  $f(x, z^+) = \lim_{y \downarrow z} f(x, y)$ .

*Proof.* Assume  $w > 0$  and let  $w > \Delta > 0$  be arbitrary. As for SQ( $d$ ), we write:

$$f(t + \Delta, w) = Q_{1,w} + Q_{2,w} + Q_{3,w}. \tag{6.31}$$

For  $Q_{1,w}$  we consider the case where no arrivals occur in the interval  $[t, t + \Delta]$ : if the cavity queue at time  $t$  has a workload exactly equal to  $w + \Delta$  and receives no arrivals in  $[t, t + \Delta]$ , it has a workload equal to  $w$  at time  $t + \Delta$ . Therefore we find:

$$Q_{1,w} = f(t, w + \Delta) - \lambda d \left( \int_0^\Delta \pi_0(t + \delta) f(t + \delta, w + \Delta - \delta) d\delta \right) + o(\Delta).$$

For  $Q_{2,w}$  we consider the case where a single arrival occurs when the queue at the cavity is busy: in this case at some time  $t + \delta, \delta \in [0, \Delta]$  an arrival of size  $w + \Delta - u$  occurs, while the queue at the cavity has workload  $u - \delta$  for some  $u \in (\delta, w + \Delta]$ . This arrival only joins the queue at the cavity if the other  $d - 1$  queues have a workload that exceeds  $u - \delta$ , hence we find:

$$\begin{aligned} Q_{2,w} &= \lambda d \int_0^\Delta \pi_0(t + \delta) \\ &\quad \int_{u=\delta}^{w+\Delta} f(t + \delta, u - \delta) \bar{F}(t + \delta, u - \delta)^{d-1} g(w + \Delta - u) du d\delta + o(\Delta). \end{aligned}$$

Finally a single arrival may occur when the cavity queue is empty: in this case a job of size  $w + \Delta - \delta$  arrives at time  $t + \delta$  for some  $\delta \in [0, \Delta]$ . Hence,

$$\begin{aligned} Q_{3,w} &= \lambda d \int_0^\Delta \pi_0(t + \delta) \frac{(1 - \bar{F}(t + \delta, 0)^d)}{d} g(w + \Delta - \delta) d\delta \\ &\quad + \lambda \int_0^\Delta \frac{1 - \pi_0(t + \delta)}{1 - \bar{F}(t + \delta, 0)} (1 - \bar{F}(t + \delta, 0)) g(w + \Delta - \delta) d\delta + o(\Delta). \end{aligned}$$

By subtracting  $f(t, w + \Delta)$ , dividing by  $\Delta$  and letting  $\Delta$  decrease to zero, we find (6.29) from (6.31).

We still require an equation for  $F(t, 0)$ , the probability that the server is idle. A server may be idle at time  $t + \Delta$  by remaining idle in  $[t, t + \Delta]$  or by having a workload equal to  $\Delta - \delta, \delta < \Delta$  at time  $t + \delta$ . We therefore find:

$$\begin{aligned} F(t + \Delta, 0) &= F(t, 0) - \lambda d \int_0^\Delta \pi_0(t + \delta) \frac{(1 - \bar{F}(t + \delta, 0)^d)}{d} d\delta \\ &\quad - \lambda \int_0^\Delta \frac{1 - \pi_0(t + \delta)}{1 - \bar{F}(t + \delta, 0)} (1 - \bar{F}(t + \delta, 0)) d\delta + \int_0^\Delta f(t + \delta, \Delta - \delta) d\delta + o(\Delta), \end{aligned}$$

subtracting  $F(t, 0)$ , dividing by  $\Delta$  and letting  $\Delta$  tend to zero yields (6.30) after multiplying both sides by  $(-1)$ .  $\square$

This result readily provides us with the equilibrium workload distribution for the LL( $d$ ) policy with memory:

**Theorem 90.** *The ccdf of the equilibrium workload distribution for the cavity process associated to an LL( $d$ ) policy with memory satisfies the following DIDE:*

$$\bar{F}'(w) = -\lambda \left[ \bar{G}(w) + \pi_0 \cdot \left( -\bar{F}(w)^d + \int_0^w \bar{F}(u)^d g(w - u) du \right) \right]. \quad (6.32)$$

with boundary condition  $\bar{F}(0) = \rho$ . Equivalently we have:

$$\bar{F}(w) = \rho - \lambda \int_0^w (1 - \pi_0 \bar{F}(u)^d) \bar{G}(w - u) du. \quad (6.33)$$

with  $\pi_0$  the probability that the memory is empty.

*Proof.* To show this result, one first lets  $t \rightarrow \infty$  in (6.29-6.30), this way we remove the  $\frac{\partial f(t,w)}{\partial t}$  and  $\frac{\partial \bar{F}(t,0)}{\partial t}$ . One then integrates (6.29) once and uses (6.30) as a boundary condition. Using Fubini, simple integration techniques and the fact that  $f(w) = -\bar{F}'(w)$  we obtain (6.32). The last equality (6.33) can be shown by integrating once more and applying Fubini's theorem.  $\square$

We can rewrite (6.33) as

$$\pi_0^{1/d} \bar{F}(w) = \mathbb{E}[G](\lambda \pi_0^{1/d}) - (\lambda \pi_0^{1/d}) \int_0^w (1 - (\pi_0^{1/d} \bar{F}(u))^d) \bar{G}(w-u) du.$$

Comparing this expression with (6.3), we note that  $\bar{F}(w)$  in a system with memory is equal to the same probability in a system without memory with arrival rate  $\lambda \pi_0^{1/d}$  divided by  $\pi_0^{1/d}$ . Due to (6.4) we therefore have the following corollary:

**Corollary 91.** *The equilibrium workload of the queue at the cavity of an LL( $d$ ) system with memory and exponential job sizes is given by*

$$\bar{F}(w) = (\rho \pi_0 + (\rho^{1-d} - \rho \pi_0) e^{(d-1)w})^{\frac{1}{1-d}} \quad (6.34)$$

We are now able to show our main result for a memory dependent LL( $d$ ) policy:

**Theorem 92.** *Let  $0 < \rho = \lambda \mathbb{E}[G] < 1$  be arbitrary and  $R$  the response time of the memory dependent LL( $d$ ) policy with mean job size  $\mathbb{E}[G]$  and arrival rate  $\lambda$ . Further, let  $\tilde{R}$  denote the response time for the same system without memory, but with arrival rate  $\lambda \pi_0^{1/d}$ , then  $R$  and  $\tilde{R}$  have the same distribution*

*Proof.* Let  $\bar{F}(w)$  and  $\bar{H}(w)$  be the ccdf of the workload for the system with and without memory, respectively. We have  $\bar{F}(w) \pi_0^{1/d} = \bar{H}(w)$  which yields:

$$\begin{aligned} \bar{F}_R(w) &= (1 - \pi_0) \bar{G}(w) + \pi_0 \left[ \int_0^w \bar{F}(w-u)^d g(u) du + \bar{G}(w) \right] \\ &= \bar{G}(w) + \int_0^w \bar{H}(w-u)^d g(u) du, \end{aligned}$$

which can easily be seen to be equal to  $\bar{F}_{\tilde{R}}(w)$ .  $\square$

By using the results in this section, one can easily generalise many of the results presented in Chapter 2 including an analytical proof that LL( $d$ ) outperforms SQ( $d$ ) and closed form solutions for the response time distribution, mean response time and mean workload.

Setup	$N = 10$	$N = 20$	$N = 50$	$N = 100$	$N = 200$
1	1.8839	1.5363	1.3556	1.3059	1.2832
2	1.4533	1.3119	1.2313	1.2045	1.1926
3	1.5906	1.3860	1.2787	1.2399	1.2215
4	1.9086	1.3981	1.1643	1.1158	1.0999
5	2.3918	2.0132	1.8200	1.7733	1.7407
6	1.7583	1.5920	1.4943	1.4578	1.4404
7	2.0504	1.8040	1.6643	1.6161	1.5901
8	2.2790	1.5924	1.2950	1.2352	1.2186
Setup	$N = 500$	$N = 1000$	$N = 3000$	Cavity Method	
1	1.2683	1.2638	1.2574	1.2583	
2	1.1836	1.1810	1.1794	1.1787	
3	1.2110	1.2097	1.2068	1.2058	
4	1.0928	1.0921	1.0896	1.0888	
5	1.7252	1.7178	1.7146	1.7138	
6	1.4314	1.4304	1.4257	1.4256	
7	1.5753	1.5736	1.5667	1.5660	
8	1.2097	1.2096	1.2070	1.2056	

Table 6.1: Comparison of mean response time for the finite system and the cavity method.

**Proposition 93.** *For the LL( $d$ ) policy with the ISM memory scheme presented in Section 6.5.5 we have*

$$\pi_0 = \frac{1 - (1 - \rho^d)^{\frac{1}{A+1}}}{\rho^d}$$

for any job size distribution.

*Proof.* The rate at which servers send probes is equal to  $f(0) = -\bar{F}'(0)$  and it follows from (6.32) that  $f(0) = \lambda(1 - \pi_0\rho^d)$ . The memory state therefore evolves as a birth-death process with birth rate  $\lambda(1 - \pi_0\rho^d)$  and death rate  $\lambda$ . The remainder of the proof is therefore identical to the proof of Proposition 88.  $\square$

## 6.8 Finite system accuracy

The results presented in Section 6.7 all focused on the cavity process of  $SQ(d)$  and  $LL(d)$  with memory. In Table 6.1 we present simulation results which illustrate that the stationary mean response time in a finite stochastic system consisting of  $N$  servers converges to the mean response time obtained using the cavity method. We simulated a system with  $N = 10, 20, 50, 100, 200, 500, 1000$  and  $3000$  servers. The arrival rate equaled  $\lambda N$ , the runtime was set to  $10^6/N$  and we used a warm-up period equal to a third of the runtime. Job sizes have mean one and are either exponential or hyperexponential with balanced means and a SCV equal to 2 or 3.

The following 8 arbitrarily chosen settings have been considered:

**Setup 1** :  $LL(4)$ ,  $\lambda = 0.9$ , exponential job sizes and the IP memory scheme.

**Setup 2** :  $LL(3)$ ,  $\lambda = 0.8$ , exponential job sizes and the CP memory scheme (meaning memory is of infinite size).

**Setup 3** :  $LL(3)$ ,  $\lambda = 0.8$ , hyperexponential job sizes with SCV equal to 2 and BCP memory scheme with  $A = 5$ .

**Setup 4** :  $LL(2)$ ,  $\lambda = 0.85$ , hyperexponential job sizes with SCV equal to 3 and the ISM memory scheme with  $A = 10$ .

Setups 5 through 8 are the same as 1 through 4, but using  $SQ(d)$  rather than  $LL(d)$ . In all cases the mean response time appears to converge towards the response time of the cavity method. Note that in the last two setups we are considering  $SQ(d)$  with memory and hyperexponential job sizes. In this case the response time of the cavity method is simply computed as the response time in the same system without memory, but with arrival rate  $\lambda\pi_0^{1/d}$ .

## 6.9 Numerical example

In this section we briefly demonstrate the type of numerical results that can be obtained using our findings. This section is not intended as a detailed comparison of the different memory schemes presented in Section 6.5

Figure 6.2 focuses on the  $SQ(5)$  policy with exponential job sizes with mean one and a memory size  $A$  of 4 (except for CP). For the BCP and ISM memory schemes the dispatcher is assumed to send its  $d$  probes one at a time (if memory is empty upon a job arrival) and stops probing as soon as an idle server is found. This is also the case for the setting without memory (labeled *No memory*). For the CP memory scheme we assume that the dispatcher has

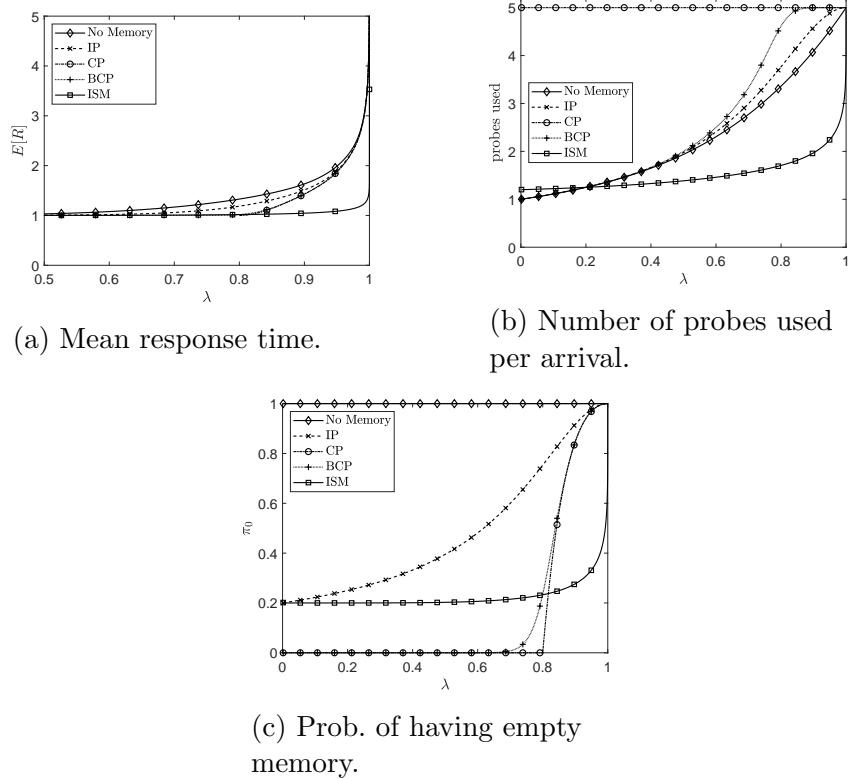


Figure 6.2: Performance of the different memory schemes for  $SQ(5)$  with exponential job sizes with mean one.

infinite memory. We plot the mean response times, the probability of having empty memory  $\pi_0$  and the average number of probes/messages used per job arrival.

In Figure 6.2a we see that the mean response time is nearly optimal for all schemes when the load is low (say below 0.5). For higher loads we see that the ISM scheme is the best, followed by the CP/BCP, IP and No Memory scheme. The ISM scheme is especially powerful when the load is close to one as all the other schemes use probing and probes are highly unlikely to locate an idle server. The results of CP and BCP are very close to each other, which indicates that a very small amount of memory may suffice.

In Figure 6.2b we depict the average number of probes that each of these memory schemes use. If we look at the results for the No Memory, CP/BCP

and IP scheme, we see that the schemes that achieved a lower mean response time use more probes. In this particular case the BCP scheme may appear to be superior to CP as it has a similar response time and uses far less probes, but keep in mind that probes are transmitted one at a time by BCP, while CP can transmit the  $d$  probes at once (which is faster). Looking at both the mean response time and number of probes/messages used, the ISM scheme is clearly best in this case.

In Figure 6.2c we look at the probability of having an empty memory when a job arrives. For the IP scheme and a load  $\lambda \approx 0$ , the dispatcher almost always discovers 5 idle servers and therefore  $\pi_0$  is close to  $1/5$ . For (B)CP we note that as long as the load is sufficiently low (that is,  $5 < \frac{1}{1-\lambda}$  or equivalently  $\lambda < 4/5$ ), we have  $\pi_0 \approx 0$ , but for larger  $\lambda$  values it sharply increases to one. When  $\lambda \approx 4/5$  we also see the most significant gain in response time for (B)CP (see Figure 6.2a). For the ISM memory scheme, we observe that when  $\lambda$  is sufficiently small:

$$\pi_0 \approx \frac{1}{5} = \frac{1}{A+1} = \lim_{\rho \rightarrow 0^+} \frac{1 - (1 - \rho^d)^{\frac{1}{A+1}}}{\rho^d},$$

which is independent of  $d$ . Only when  $\lambda$  is close to one,  $\pi_0$  starts a very steep climb to one.

## 6.10 Mean field limit for a critically loaded system

Throughout this section and Section 6.11, we assume job sizes are exponential with mean equal to one. The assumption that the mean equals one is merely a technicality to ease notation. Our goal is to compute the limit:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1 - \lambda)}, \quad (6.35)$$

where  $R_\lambda$  denotes the response time for some  $SQ(d)/LL(d)$  memory based load balancing policy. To this end, we employ the framework developed in Chapter 5. Note that this limit gives an indication of the performance of the load balancing policy under a high load. Moreover, it is easy to see that this limit remains unchanged if we swap the mean waiting time by either the

mean response time or the mean queue length/workload. To emphasize that  $\pi_0$  depends on  $\lambda$ , we denote  $\pi_0$  as  $\pi_0(\lambda)$  in this section. Define

$$T_\lambda(x) = \lambda\pi_0(\lambda)x^d, \quad (6.36)$$

and note that  $(u_k)_k$  for  $SQ(d)$  resp.  $\bar{F}(w)$  for  $LL(d)$  satisfy the relations  $u_{k+1} = T_\lambda(u_k)$  resp.  $\bar{F}'(w) = T_\lambda(\bar{F}(w)) - \bar{F}(w)$ .

**Theorem 94.** *For the memory dependent  $SQ(d)$  policy, provided that*

$$\lim_{\lambda \rightarrow 1^-} \pi'_0(\lambda) < \infty$$

and  $\lim_{\lambda \rightarrow 1^-} \pi_0(\lambda) = 1$ , we obtain the limit:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda^{(SQ(d))}]}{\log(1-\lambda)} = \frac{1}{\log(d)}, \quad (6.37)$$

while for the  $LL(d)$  variant we have:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda^{(LL(d))}]}{\log(1-\lambda)} = \frac{1}{d-1}. \quad (6.38)$$

*Proof.* We validate assumptions 1-7 introduced in Chapter 5 from which the result directly follows.

- (a) In this step we should show there exists some continuous function  $u : \lambda \rightarrow u_\lambda$  such that  $u_\lambda \in (1, \infty)$ ,  $T_\lambda(u_\lambda) = u_\lambda$  and  $\lim_{\lambda \rightarrow 1^-} u_\lambda = 1$ . For our model, it is not hard to find an explicit formula for  $u_\lambda = T_\lambda(u_\lambda)$ , namely :

$$u_\lambda = (\lambda\pi_0(\lambda))^{1/(1-d)}.$$

- (b) One trivially verifies that  $T_\lambda(0) = 0$ , and for any  $u \in (0, 1)$  we have  $T_\lambda(u) < u$  and  $\lim_{\lambda \rightarrow 1^-} \frac{T_\lambda(u)}{u} < 1$ .

- (c) We define  $h_\lambda(x) = \frac{u_\lambda - T_\lambda(u_\lambda - x)}{x}$ , we now verify that  $h'_\lambda(x) < 0$  for any  $x \in [u_\lambda - 1, u_\lambda]$ . To this end, we first compute:

$$h'_\lambda(x) = \frac{\lambda\pi_0(\lambda)(u_\lambda - x)^d - u_\lambda + \lambda\pi_0(\lambda)dx(u_\lambda - x)^{d-1}}{x^2}.$$

In case  $d = 1$  this expression simplifies to  $-\frac{u_\lambda}{x^2}(1 - \lambda\pi_0(\lambda)) < 0$ . For  $d \geq 2$  we compute the derivative of  $(x^2 \cdot h'_\lambda(x))$ :

$$(x^2 h'_\lambda(x))' = -\lambda\pi_0(\lambda)d(d-1)x(u_\lambda - x)^{d-2},$$

which is negative. As one easily verifies that  $(x^2 h'_\lambda(x))$  equals zero in  $x = 0$ , this indeed shows  $h_\lambda$  is decreasing on  $[u_\lambda - 1, u_\lambda]$ .

- (d) This is a technicality which is automatically satisfied because  $h_\lambda$  is decreasing on  $[u_\lambda - 1, u_\lambda]$ , which we showed in the previous step.
- (e) For this step we need to compute the value of:

$$A = \lim_{\lambda \rightarrow 1^-} h_\lambda(u_\lambda - 1) = \lim_{\lambda \rightarrow 1^-} \frac{u_\lambda - \lambda\pi_0(\lambda)}{u_\lambda - 1} = \lim_{\lambda \rightarrow 1^-} \frac{u'_\lambda - \pi_0(\lambda) - \lambda\pi'_0(\lambda)}{u'_\lambda}. \quad (6.39)$$

It is not hard to see that:

$$u'_\lambda = \frac{1}{1-d}(\lambda\pi_0(\lambda))^{-d/(d-1)} \cdot (\pi_0(\lambda) + \lambda\pi'_0(\lambda)).$$

Using the fact that  $\lim_{\lambda \rightarrow 1^-} \pi_0(\lambda) = 1$ , we obtain (continuing from (6.39)):

$$A = \lim_{\lambda \rightarrow 1^-} \frac{\frac{1+\pi'_0(\lambda)}{1-d} - 1 - \pi'_0(\lambda)}{\frac{1+\pi'_0(\lambda)}{1-d}} = d.$$

- (f) For this this step, we need to compute the value

$$B = \lim_{\lambda \rightarrow 1^-} \frac{\log(u_\lambda - 1)}{\log(1 - \lambda)} = \lim_{\lambda \rightarrow 1^-} -\frac{(1 - \lambda)u'_\lambda}{u_\lambda - 1},$$

at this point, we use the assumption that  $\lim_{\lambda \rightarrow 1^-} \pi'_0(\lambda) < \infty$ , as this implies that  $u'_\lambda < \infty$ , allowing us to use l'Hopital only on  $\frac{1-\lambda}{u_\lambda-1}$ , by which it trivially follows that  $B = 1$ .

- (g) For the last step, we should verify that  $\lim_{\varepsilon \rightarrow 0^+} \lim_{\lambda \rightarrow 1^-} h_\lambda(\varepsilon) = A$ . Indeed,

$$\lim_{\varepsilon \rightarrow 0^+} \lim_{\lambda \rightarrow 1^-} h_\lambda(\varepsilon) = \lim_{\varepsilon \rightarrow 0^+} \frac{1 - (1 - \varepsilon)^d}{\varepsilon} = d = A.$$

□

It is not hard to see that Theorem 94 applies to all policies described in Section 6.5, except the ISM policy which we discussed in Section 6.5.5. Indeed, ISM is the only policy for which  $\lim_{\lambda \rightarrow 1^-} \pi'_0(\lambda) = \infty$ , see also Figure 6.2c. We therefore find the limiting value to be slightly different in case of ISM.

**Theorem 95.** *For the memory dependent SQ(d) policy with ISM and a memory size equal to M, we obtain the limit:*

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[R_\lambda^{(SQ)}]}{\log(1-\lambda)} = \frac{1}{A+1} \frac{1}{\log(d)}, \quad (6.40)$$

while for the LL(d) variant we have:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[R_\lambda^{(LL)}]}{\log(1-\lambda)} = \frac{1}{M+1} \frac{1}{d-1}. \quad (6.41)$$

*Proof.* One can copy the proof of Theorem 94, except for step (f), as it was used that  $\lim_{\lambda \rightarrow 1^-} \pi'_0(\lambda) < \infty$ , while it is easy to verify that this limit is indeed infinite for ISM. Let us first compute  $u'_\lambda$ , using (6.9) we find:

$$\begin{aligned} u'_\lambda &= [(\lambda \pi_0(\lambda))^{1/(1-d)}]' \\ &= \left( \frac{\lambda}{(1 - (1 - \lambda^d)^{1/(M+1)})^{1/(d-1)}} \right)' \\ &= \frac{(d-1)(M+1) - d\lambda^d(1 - (1 - \lambda^d)^{\frac{1}{M+1}})^{-1}(1 - \lambda^d)^{\frac{-M}{M+1}}}{(d-1)(M+1)(1 - (1 - \lambda^d)^{1/(M+1)})^{1/(d-1)}}, \end{aligned}$$

noting that (by a simple application of l'Hopital's rule) we have  $\lim_{\lambda \rightarrow 1^-} \frac{1-\lambda}{1-u_\lambda} =$

0, we obtain:

$$\begin{aligned}
B &= \lim_{\lambda \rightarrow 1^-} \frac{(1-\lambda)u'_\lambda}{1-u_\lambda} \\
&= - \lim_{\lambda \rightarrow 1^-} \left[ \frac{1-\lambda}{1-u_\lambda} \frac{d\lambda^d}{(d-1)(M+1)} \frac{(1-\lambda^d)^{-M/(M+1)}}{(1-(1-\lambda^d)^{1/(M+1)})^{d/(d-1)}} \right] \\
&= - \frac{d}{(d-1)(M+1)} \lim_{\lambda \rightarrow 1^-} \left[ \frac{1-\lambda}{(1-\lambda^d)^{M/(M+1)}(1-u_\lambda)} \right] \\
&= - \frac{d}{(d-1)(M+1)} \lim_{\lambda \rightarrow 1^-} \frac{1-\lambda}{1-\lambda^d} \cdot \lim_{\lambda \rightarrow 1^-} \frac{(1-\lambda^d)^{1/(M+1)}}{1-u_\lambda} \\
&= - \frac{1}{(d-1)(M+1)} \cdot \lim_{\lambda \rightarrow 1^-} \frac{(1-\lambda^d)^{1/(M+1)}}{1 - \frac{\lambda}{(1-(1-\lambda^d)^{1/(M+1)})^{1/(d-1)}}}.
\end{aligned}$$

For ease of notation, let us define  $\xi = (1-\lambda^d)^{1/(M+1)}$ . We find that the above simplifies to :

$$B = - \frac{1}{(d-1)(M+1)} \lim_{\xi \rightarrow 0^+} \frac{\xi(1-\xi)^{1/(d-1)}}{(1-\xi)^{1/(d-1)} - (1-\xi^{M+1})^{1/d}} = \frac{1}{M+1},$$

where the last equality follows from a final use of l'Hopital's rule. Combining this with the proof of Theorem 94, we may conclude the proof.  $\square$

## 6.11 Low load limit

In this section, we investigate the system in with  $\lambda$  close to 0 rather than 1. In particular, we investigate the behaviour of the expected waiting time in the low load limit, i.e.  $\lim_{\lambda \rightarrow 1^-} (\mathbb{E}[W_\lambda] - 1)$ . The value of this limit is always zero but we show that in case of exponential job sizes, we are able to obtain a closed form expression for

$$\lim_{\lambda \rightarrow 1^-} \frac{\mathbb{E}[W_\lambda^{(1)}]}{\mathbb{E}[W_\lambda^{(2)}]}.$$

Here  $W_\lambda^{(1)}, W_\lambda^{(2)}$  denote the response time distribution of two different load balancing policies with arrival rate  $\lambda$ . We take the quotient of the expected waiting times rather than response times as the quotient for the expected

response times is trivially one for any 2 load balancing policies. This quantity signifies the quality of a policy under a low arrival rate. In particular we have the following result:

**Proposition 96.** *Let  $W_\lambda^{(i)}$  ( $i = 1, 2$ ) denote the response time for a memory dependent load balancing policy with probability  $\pi_0^{(i)}(\lambda)$  of having an empty memory, using either  $SQ(d_i)$  or  $LL(d_i)$ . Furthermore, we assume job sizes are exponentially distributed with mean 1. We find:*

1. If  $d_1 < d_2$ , we have

$$\lim_{\lambda \rightarrow 0^+} \frac{\mathbb{E}[W_\lambda^{(1)}]}{\mathbb{E}[W_\lambda^{(2)}]} = \infty.$$

2. If  $d_1 = d_2 = d$  and both policies use the same strategy (either  $SQ(d)$  or  $LL(d)$ ), we have:

$$\lim_{\lambda \rightarrow 0^+} \frac{\mathbb{E}[W_\lambda^{(1)}]}{\mathbb{E}[W_\lambda^{(2)}]} = \lim_{\lambda \rightarrow 0^+} \frac{\pi_0^{(1)}(\lambda)}{\pi_0^{(2)}(\lambda)}.$$

3. If  $d_1 = d_2 = d$  and  $W_\lambda^{(1)}$  employs the  $SQ(d)$  policy while  $W_\lambda^{(2)}$  uses the  $LL(d)$  policies, we find:

$$\lim_{\lambda \rightarrow 0^+} \frac{\mathbb{E}[W_\lambda^{(1)}]}{\mathbb{E}[W_\lambda^{(2)}]} = \lim_{\lambda \rightarrow 0^+} \frac{\pi_0^{(1)}(\lambda)}{d\pi_0^{(2)}(\lambda)}.$$

*Proof.* From (6.2) with arrival rate  $\lambda\pi_0^{1/d}$  and mean job size equal to 1, one finds that the expected response time for  $SQ(d)$  is given by:

$$\mathbb{E}[W_\lambda^{(SQ(d))}] = \sum_{n=2}^{\infty} (\lambda\pi_0^{1/d})^{\frac{dn}{d-1}-1}.$$

By Theorem 84, we find that this expression corresponds to the mean response time of a memory based  $SQ(d)$  policy. Analogously, for  $LL(d)$  and using Theorem 92, we obtain that the expected response time for a memory based  $LL(d)$  policy with exponential job sizes of mean one is given by:

$$\mathbb{E}[R_\lambda] - 1 = \sum_{n=1}^{\infty} \frac{(\lambda\pi_0^{1/d})^{dn}}{1 + n(d-1)}. \quad (6.42)$$

In order to compute the sought limits, one only retains the terms with the lowest power of  $\lambda$ . For example, assume  $d_1 < d_2$  and we wish to compare  $\text{LL}(d_1)$  with  $\text{LL}(d_2)$ , it follows from (6.42) that:

$$\lim_{\lambda \rightarrow 0^+} \frac{\mathbb{E}[W_\lambda^{(1)}]}{\mathbb{E}[W_\lambda^{(2)}]} = \frac{\lambda^{d_1} \pi_0}{\lambda^{d_2} \pi_0} \cdot \frac{1 + (d_2 - 1)}{1 + (d_1 - 1)} = \infty.$$

As a second example let us consider case (3), we find:

$$\lim_{\lambda \rightarrow 0^+} \frac{\mathbb{E}[W_\lambda^{(1)}]}{\mathbb{E}[W_\lambda^{(2)}]} = \frac{(\lambda \pi_0^{1/d})^d}{\frac{(\lambda \pi_0^{1/d})^d}{d}} = \lim_{\lambda \rightarrow 0^+} \frac{d \pi_0^{(1)}(\lambda)}{\pi_0^{(2)}(\lambda)}.$$

□

**Remark 52.** For the methods discussed in Section 6.5 we can easily compute the limit  $\lim_{\rho \rightarrow 0^+} \pi_0(\rho)$ . Indeed, by elementary calculus we find:

- For IP we have  $\lim_{\rho \rightarrow 0^+} \pi_0(\rho) = \frac{1}{d}$ .
- For CP and BCP we have  $\lim_{\rho \rightarrow 0^+} \pi_0(\rho) = 0$ .
- For ISM with memory size  $A$  we have  $\lim_{\rho \rightarrow 0^+} \pi_0(\rho) = \frac{1}{A+1}$ .

In particular, we see that, while ISM is the dominant policy in for  $\lambda \approx 1$ , it does not perform as well when the system load is low. See also Figure 6.2c for an example.

## 6.12 Conclusions and future work

In this chapter we studied the cavity process of the SQ( $d$ ) and LL( $d$ ) load balancing policies with memory. The main insight provided was that the response time distribution of the cavity process with memory is identical to the response time distribution of the cavity process of the system without memory if the arrival rate is properly set. This result holds for a large variety of memory schemes including the ones presented in Section 6.5. This insight allowed us to analyse the system when the load is either large or small. Simulation results were presented which suggest that the cavity process corresponds to the exact limit process as the number of servers tends to infinity.

As future work, it may be possible to prove that the cavity process is the proper limit process. For  $SQ(d)$  with exponential job sizes, one can build upon the framework of [16], whilst for  $LL(d)$  it might be possible to extent the framework in [83] to prove the ansatz for general job sizes.

# Chapter 7

## Improved load balancing in large scale systems using attained service time reporting

Your assumptions are your windows on the world.

Scrub them off every once in a while, or the light won't come in.

---

Isaac Asimov

### Abstract

Our interest lies in load balancing jobs in large scale systems consisting of multiple dispatchers and FCFS servers. In the absence of any information on job sizes, dispatchers typically use queue length information reported by the servers to assign incoming jobs. When job sizes are highly variable, using only queue length information is clearly suboptimal and performance can be improved if some indication can be provided to the dispatcher about the size of an ongoing job. In a FCFS server measuring the attained service time of the ongoing job is easy and servers can therefore report this attained service time together with the queue length when queried by a dispatcher.

In this chapter we propose and analyse a variety of load balancing policies that exploit both the queue length and attained service time to assign jobs,

as well as policies for which only the attained service time of the job in service is used. We present a unified analysis for all these policies in a large scale system under the usual asymptotic independence assumptions. The accuracy of the proposed analysis is illustrated using simulation.

We present extensive numerical experiments which clearly indicate that a significant improvement in waiting (and thus also in response) time may be achieved by using the attained service time information on top of the queue length of a server. Moreover, the policies which do not make use of the queue length still provide an improved waiting time for moderately loaded systems.

## 7.1 Introduction

Load balancing plays a vital role to achieve a low latency in large scale clusters. It was proven in [98] that Join the Shortest Queue (JSQ) is the optimal policy to distribute jobs in a system with  $N$  identical servers and exponential job sizes if jobs must be assigned immediately and no jockeying between servers is allowed. Ever since, the JSQ policy has often been referred to as the golden standard for load balancers. As  $N$  grows large, the overhead created by probing *all* servers at every arrival becomes too large. Therefore the Shortest Queue  $d$  (SQ( $d$ )) policy was introduced and analysed in various papers [66, 5, 94, 95] (see also Section 1.5). A survey of recent advances can be found in [87]. However, the workload in many real systems consists of a mix of many short jobs and some large jobs, where the large jobs contribute a significant part of the total workload (see e.g. [74, 27, 28]). When all servers have the same queue length, the JSQ policy simply assigns the incoming job arbitrarily, while it is better to assign the job to a server which is less likely to be serving a large job. Moreover, selecting a server with one long job may be worse than selecting a server with multiple small jobs.

Recently the Least Loaded  $d$  policy (LL( $d$ )) was analysed in chapter 2 and [12], the LL( $d$ ) policy assigns incoming jobs to a server which has the least amount of work left among  $d$  randomly selected servers. This allows one to improve the SQ( $d$ ) policy significantly. The drawback of this policy is however that it only works if job size information is available (which is mostly not the case) or when using a mechanism like late-binding (which brings significant overhead as jobs are not assigned immediately). While a server is typically unaware of its remaining workload, it can measure (up to some accuracy  $\Delta$ ) the time it has spent processing the job(s) in service. This

is especially true in FCFS servers. As jobs that have been in service for a substantial amount of time are highly likely to be long jobs that potentially have a long residual service time, using the attained service time to assign jobs to servers may improve performance.

In this chapter we propose a collection of load balancing policies that exploit both queue length and attained service time (up to some granularity  $\Delta$ ) information reported by the servers to assign jobs. Note that such policies do not require any knowledge of the size of incoming jobs or the job size distribution. We develop a unified analysis which may be used to analyse these load balancing policies in a large scale system under the usual asymptotic independence assumptions. Our main observation is that for small to moderate system loads and  $d$  sufficiently large, the improvement from using the attained service time information is substantial, while this improvement decreases as the system becomes critically loaded. For example, even policies which solely rely on the attained service time of the job in service may outperform the  $SQ(d)$  policy with  $d = 5$  for a large range of arrival rates (see Figure 7.11a).

As we may not aspire to approach the performance of the  $LL(d)$  policy (as it is impossible to predict the exact workload using only the attained service time of one job and the queue length), we set ourselves a different goal. We define the Least Expected Workload policy ( $LEW(d)$ ) as the policy which assigns any incoming job to the server which has the least expected work left at its queue among  $d$  randomly selected servers. Note that the  $LEW(d)$  policy uses knowledge of the job size distribution to estimate the residual service time. We find that many of our policies that do not require such knowledge are able to achieve performance which is similar to that of the  $LEW(d)$  policy.

In our analysis we assume incoming jobs are PH Distributed. PH distributions are distributions with a modulating finite state background Markov chain [55] and any general positive-valued distribution can be approximated arbitrary close with a PH distribution. Further, various fitting tools are available online for PH distributions (e.g., [75, 52]).

The main contributions of this chapter are as follows:

1. We propose various load balancing policies that exploit queue length and attained service time information reported by the servers. We demonstrate that all of our policies achieve a significant reduction for the average waiting time of a job compared to  $SQ(d)$  under low to mod-

erate workloads, with a performance that is often close to the LEW( $d$ ) policy. For some policies the waiting time is reduced for all workloads.

2. We present a unified analysis which is applicable for all policies under consideration (and other variations thereof). This analysis may be of independent interest as it provides a means to assess the performance of an M/PH/1 queue with queue length and attained service time (up to some granularity  $\Delta$ ) dependent arrival rates.
3. We validate the accuracy of the asymptotic independence assumption used in the analysis using simulation experiments.

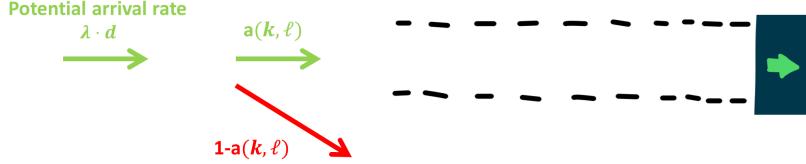
## 7.2 Structure of the chapter

The chapter is structured as follows. First, we give an intuitive explanation for the analysis carried out in this chapter, see Section 7.3. In Section 7.4 we formally define the model of interest. In Section 7.5 we present seven different policies which we study throughout the chapter. In Section 7.6 we present the method used to analyse our load balancing policies. In Section 7.7 we verify our analysis by means of simulation of finite systems. Section 7.8 consists of extensive numerical experimentation investigating the impact of all parameters in our proposed model. Finally, we conclude in Section 7.9.

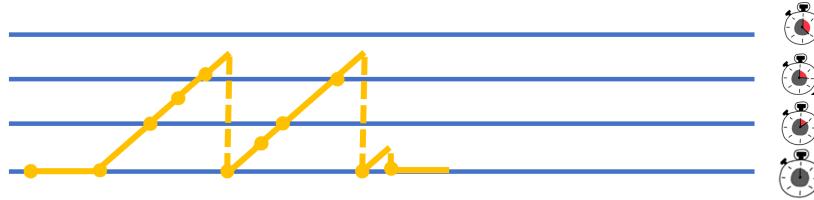
## 7.3 Informal intuition

The type of models we analyse throughout this work take both the runtime of the job at the head of the queue and the queue length into account. This makes sense as in most computer systems, jobs tend to be either really small or very large. Therefore, if a job has received a lot of service already, it is likely to be a long job. In addition, if a server is having issues, it will also take the server longer to process its jobs and our policies are therefore less likely to assign jobs to broken servers. For ease of presentation, we assume job sizes are exponential throughout this section.

**Remark 53.** *Due to the memoryless property of the exponential distribution, using the attained service time in case of exponential job sizes is completely useless. However, a good understanding of the methodology for exponential job sizes, aids to understand the more general analysis for PH job sizes.*



(a) Depiction of the cavity queue for the attained service time dependent load balancing policies considered in this work.



(b) Age process for job at the head of the queue. Each circle represents an event, either passing a time boundary, an arrival, a departure or the start of our observations.

Figure 7.1: Figures which aid in the understanding of the cavity queue associated to the attained service time dependent load balancing policies.

In order to analyse these policies, we employ the queue at the cavity methodology, for which we presented an intuitive explanation in Section 1.5. In Figure 7.1a, we depict the cavity queue, there is a *potential* arrival rate to the cavity queue equal to  $\lambda \cdot d$ , each potential arrival then has some probability, denoted by  $a(k, \ell)$  to become an actual arrival. We have  $k \in \mathbb{N}$  which denotes the amount of time that the job currently at the head of the queue has been receiving service. The value  $\ell \in \mathbb{N}$  denotes the number of jobs present in the cavity queue at the time of the potential arrival. Intuitively, we can picture these  $k$  values as ticks on a clock, indeed: while time is a continuous process which takes place in  $[0, \infty)$ , any observation of time we make as humans is discrete (that is in minutes, seconds, milliseconds or ...). In Figure 7.1b, we plot time as a continuum, but the horizontal blue lines represent the ticks on our clock. We have plotted a snippet of a typical process for  $(k, \ell)$  in Figure 7.1b, the first orange dot represents the start of our observation, let us assume that we start in state  $(0, 0)$ . We then describe how our state changes over time, each circle represents a state change. We

$$(k, \ell) \rightarrow (k+1, \ell'); \ell' \geq \ell \quad (k, \ell) \rightarrow (0, \ell'); \ell' \geq \ell - 1$$

$$P(\text{Exp}(1) > \Delta) \cdot P(\ell' = \ell \text{ arrivals}) \quad P(\text{Exp}(1) < \Delta) \cdot P(\ell' = \ell + 1 \text{ arrivals})$$



Figure 7.2: Representation of all possible transitions starting in  $(k, \ell)$  assuming exponential job sizes.

therefore describe the transition represented by each dot from left to right:

- The second orange dot represents an arrival, therefore our state changed from  $(0, 0)$  to  $(0, 1)$ .
- The third orange dot represents the crossing of the first boundary which corresponds to the state change from  $(0, 1)$  to  $(1, 1)$ .
- We then have an arrival, changing the state  $(1, 1)$  to  $(1, 2)$ .
- Next, we again cross a clock tick changing our state to  $(2, 2)$ .
- We then have a job departure, which changes the state  $(2, 2)$  to  $(0, 1)$ .
- The next state changes are  $(0, 1) \rightarrow (0, 2) \rightarrow (1, 2) \rightarrow (2, 2) \rightarrow (0, 1) \rightarrow (0, 0)$ .

If we were to keep track of the exact runtime information, the state space of a single queue would be  $[0, \infty)^B$  with  $B$  the maximum queue length. For this state space, it is numerically challenging to obtain a solution of the associated stochastic process. Therefore, we restrict our view to the times at which the runtime exactly equals one of our clock ticks and we censor out everything that happens in between two clock ticks.

In Figure 7.2, we depict the possible transitions starting in some state  $(k, \ell)$ . We denote by  $\Delta > 0$  the time between 2 time ticks. Starting in some state  $(k, \ell)$ , there are two options, we either have a job departure before the next clock tick, or we do not. If we have no job departure before the next clock tick, the state simply changes to  $(k, \ell')$ , where  $\ell'$  equals  $\ell$  plus the number arrivals in time  $\Delta$ , where arrivals occur according to a  $\text{Poisson}(\lambda da(k, \ell))$  process (which can be computed without much effort). When there is a job

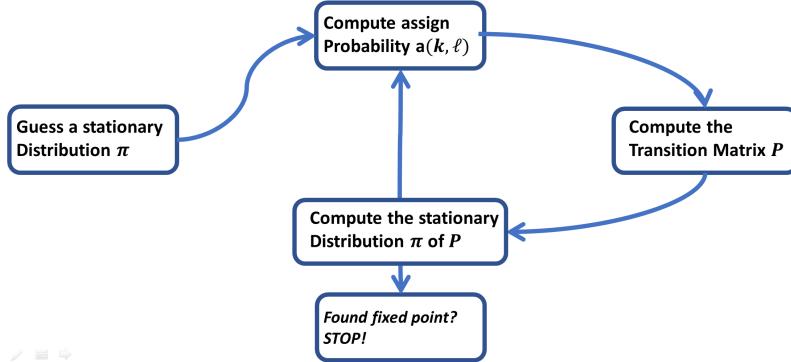


Figure 7.3: Graphical representation of the FPI used to find the stationary distribution for an attained service time based policy.

departure, we reset the clock and the state changes to  $(0, \ell')$ , where  $\ell'$  is equal to  $\ell - 1$  plus the number of arrivals in a time equal to  $(E \mid E < \Delta)$ , with  $E$  an exponential distribution with mean one. The arrivals again occur with a rate equal to  $\lambda d a(k, \ell)$ . We can imagine that it is possible to compute the transition matrix  $P$  associated to this DTMC and the stationary distribution of this DTMC yields the sought stationary distribution  $\pi$ . However, in order to compute  $P$ , we require the value of  $a(k, \ell)$  and these in their turn depend on the stationary distribution  $\pi$ . Therefore, we should compute the stationary distribution  $\pi$  using a FPI. We represent this process as:

1. Guess a stationary distribution  $\pi^{(0)}$ , set  $n = 1$  and choose some tolerance  $\text{tol} > 0$ .
2. Compute the asign probabilities  $a(k, \ell)$  based on  $\pi^{(n-1)}$ .
3. Compute the transition matrix  $P$ .
4. Compute the stationary distribution  $\pi^{(n)}$  of  $P$ .
5. If  $\|\pi^{(n)} - \pi^{(n-1)}\| < \text{tol}$ , set  $\pi = \pi^{(n)}$  and stop, otherwise increment  $n$  by one and return to step 2.

We graphically illustrate this process in Figure 7.3. Using this method we are able to obtain performance metrics such as the response time distribution. In this chapter, we formalize and extend the ideas represented in this section.

## 7.4 Model description

The model we consider consists of  $N$  homogeneous servers (with  $N$  large) which all process jobs using FCFS scheduling. We assume jobs arrive according to a Poisson  $\lambda N$  process, these arrivals may occur to multiple dispatchers. We assume job sizes have a PH distribution with representation  $(\alpha, A)$ . We use the notation  $\mu = -A\mathbf{1}$ , where  $\mathbf{1}$  is a column vector with all its entries equal to one, and denote by  $m$  the number of phases of the job size distribution. Hence, the probability of having a job size smaller than or equal to  $x$  is given by  $1 - \alpha e^{Ax}\mathbf{1}$ . Furthermore, we assume w.l.o.g. that the mean job size is equal to one.

We pick a  $\Delta > 0$  and  $r > 0$  and define  $c_k = k \cdot \Delta$ , for  $1 \leq k \leq r$ , and set  $c_0 = 0$  and  $c_{r+1} = \infty$ . We say a job is in layer- $k$  if its attained service time satisfies  $a \in (c_{k-1}, c_k]$  and in layer-0 when the server is idle. When a server is queried by a dispatcher for queue length and attained service time information, the server reports its queue length  $\ell$  and the layer  $k$  in which the attained service time of the job in process lies. This corresponds to stating that the servers measure the attained service time up to some granularity  $\Delta$ . Whenever an arrival occurs to a dispatcher, the dispatcher picks  $d$  servers at random and queries these  $d$  servers. The dispatcher then uses some policy based on the  $(k, \ell)$  values reported by the  $d$  servers to assign the incoming job to one of these servers. We note that our analysis approach actually applies for any set of threshold values  $c_k$  such that  $0 = c_0 < c_1 < \dots < c_r < c_{r+1} = \infty$ .

## 7.5 Load balancing policies

In order to define our policies, we first define  $\mathcal{R}_n = [0, \infty)^n$  with the lexicographic order. All our policies are based on the same basic idea, from every server, the dispatcher receives the layer and queue length information coded as  $(k, \ell) \in \mathbb{N} \times \mathbb{N}$ . The dispatcher maps the information  $(k, \ell)$  to some value  $\xi(k, \ell) \in \mathcal{R}_n$  which is interpreted as a measure for the “aversion” of the chosen server. The incoming job is then assigned to the server for which the  $\xi(k, \ell)$  value is the smallest (amongst the  $d$  chosen servers), with ties being broken uniformly at random.

**Example 97.** Some basic examples of policies are random routing, which corresponds to picking  $\xi(k, \ell) = 0$  for all  $(k, \ell)$  and  $SQ(d)$  for which one sets

$$\xi(k, \ell) = \ell.$$

We now introduce a number of load balancing policies that are all described by defining  $\xi : \mathbb{N} \times \mathbb{N} \rightarrow \mathcal{R}_n$ . For our policies, we always have  $\xi(0, 0) = (0)_{i=1}^n$  and  $\xi(k, \ell) \neq (0)_{i=1}^n$  if  $\ell \geq 1$ . This ensures that we always assign to idle queues if possible. As jobs with larger attained service times are more likely to be large jobs with a potentially large residual service time,  $\xi$  is always chosen to be non-decreasing in  $k$ .

Throughout, we assume that  $\xi$  is chosen such that our model remains stable for  $\lambda < 1$ . More often than not, it suffices to note that the load balancing policy outperforms the random routing policy.

### 7.5.1 $\text{SQ}(d)$ with Runtime based Tie Breaking ( $\text{SQ}(d)$ -RTB)

This policy mainly relies on the queue length information, but in case multiple chosen servers have the same number of pending jobs, the job is routed to the server for which the job at the head of the queue has currently received the least service, that is, is in the lowest layer  $k$ . The intuition is that the job in service is more likely to be a short job that will finish soon. For this policy, we set:

$$\xi(k, \ell) = (\ell, k). \quad (7.1)$$

We further refer to this policy as the  $\text{SQ}(d)$  with Runtime based Tie Breaking policy, denoted by  $\text{SQ}(d)$ -RTB. It is similar to  $\text{SQ}(d)$  but may improve performance by using the attained service time to resolve ties.

### 7.5.2 $\text{SQ}(d)$ with Runtime Exclusion ( $\text{SQ}(d)$ -RE( $T$ ))

For this policy, we only rely on the queue length information, as long as the attained service time does not exceed some threshold  $T$  (e.g.  $T = 2$ ). When a server is queried it simply replies by stating its queue length and whether or not the attained service time is more than time  $T$ . This corresponds to setting  $\Delta = T$  and  $r = 1$  in our model. Whenever there are  $1 \leq d' \leq d$  servers for which the attained service time does not exceed the threshold  $T$ , we assign the job to the server with the least number of jobs amongst the  $d'$  chosen queues. If all  $d$  servers report an attained service time above  $T$  (meaning  $k = 2$ ), the job is routed to the server with the least number of jobs amongst all  $d$  chosen servers. The idea is that we assume a job is *large*

when its runtime *significantly exceeds* the average runtime of a job. For this policy, we define:

$$\xi(k, \ell) = (k, \ell), \quad (7.2)$$

We refer to this policy as  $\text{SQ}(d)$  with Runtime Exclusion, denoted by  $\text{SQ}(d)\text{-RE}(T)$ . In [69], a somewhat similar policy is studied, where one uses the  $\text{SQ}(d)$  policy to select a queue and each job carries one bit of information (to be interpreted as an indication of whether the job is large). Long jobs are put at the back of the queue, while short jobs are put at the head of the queue. The main difference lies in the fact that we infer the job size information at runtime and do not assume jobs carry an indication about their size. Further all jobs are placed at the back of the queue in our case.

### 7.5.3 $\text{SQ}(d)$ -RTB with Runtime Exclusion ( $\text{SQ}(d)$ -RTB- $\text{RE}(T)$ )

This policy is a mix of  $\text{SQ}(d)$ -RTB and  $\text{SQ}(d)$ -RE( $T$ ). We set some threshold  $T > 0$  where we suspect that a job is large once the attained service time exceeds this threshold. When  $d' \geq 1$  of the randomly chosen servers report an attained service time smaller than  $T$ , we employ the  $\text{SQ}(d)$ -RTB policy to decide which of these  $d'$  servers receives the incoming job. When the attained service time of the  $d$  selected servers exceeds  $T$ , we use  $\text{SQ}(d)$ -RTB to pick a server. This policy can also be described by defining an appropriate  $\xi$ :

$$\xi(k, \ell) = (\delta\{k > s\}, \ell, k), \quad (7.3)$$

here  $\delta\{A\}$  is equal to one if  $A$  is true and zero otherwise and  $T = \Delta s = c_s$  for some  $s \leq r$ . We refer to this policy as the  $\text{SQ}(d)$ -RTB with Runtime Exclusion policy, denoted by  $\text{SQ}(d)$ -RTB-RE( $T$ ).

### 7.5.4 Least Attained Service ( $\text{LAS}(d)$ )

For this policy, we assume that the dispatcher assigns incoming jobs to the server for which the job at the head of the queue has attained the least amount of service among  $d$  randomly selected servers. This policy is defined by:

$$\xi(k, \ell) = k. \quad (7.4)$$

We further refer to this policy as the Least Attained Service policy, denoted as  $\text{LAS}(d)$ . The success of this policy relies on having highly variable jobs,

such that it is more important to know whether the job at the head of a queue is large rather than knowing the number of jobs in the queue. Note that if we pick  $\Delta > 0$  sufficiently small, the probability of having a tie tends to zero.

### 7.5.5 LAS( $d$ ) with Queue length based Tie Breaking (LAS( $d$ )-QTB)

For this policy, we assign the job to the queue for which the attained service time of the job at the head of the queue is minimal, but in case there is a tie between multiple servers, we assign the incoming job to the server with the fewest number of waiting jobs. To this end, we define:

$$\xi(k, \ell) = (k, \ell). \quad (7.5)$$

We refer to this policy as LAS( $d$ ) with Queue length based Tie Breaking, which we denote by LAS( $d$ )-QTB.

**Remark 54.** *While for other policies, having a small  $\Delta > 0$  is always beneficial, the performance of LAS( $d$ )-QTB may actually improve by having a larger value of  $\Delta$ . In particular, letting  $r = 1$  and  $c_1 = T$  this policy reduces to SQ( $d$ )-RE( $T$ ). While setting an arbitrary  $r$  and  $c_1, \dots, c_r$ , this policy may be viewed as an SQ( $d$ ) policy with multiple thresholds.*

### 7.5.6 Runtime Exclusion (RE( $d, T$ ))

For this policy, we set  $r = 1$  and  $\Delta = T$ . In this case having  $k = 0$  means that the server is idle,  $k = 1$  means the job in service has an attained service time below  $T$  and  $k = 2$  otherwise. We find:

$$\xi(k, \ell) = k. \quad (7.6)$$

We refer to this policy as the Runtime Exclusion policy, denoted by RE( $d, T$ ). It is a special case of the LAS( $d$ ) policy with  $r = 1$ .

### 7.5.7 Least Expected Workload (LEW( $d$ ))

For this policy we assume that the job size distribution of the incoming jobs is known, such that we can compute the mean residual service time given the

attained service time. In this case we can use the more refined information of the expected residual service time  $\mathbb{E}[X \mid X > c_k] - c_k$  to decide which queue should receive the incoming job. This policy also fits in our framework by defining:

$$\xi(k, \ell) = (\ell - 1) \cdot \mathbb{E}[X] + \mathbb{E}[X \mid X \geq c_k] - c_k, \quad (7.7)$$

we refer to this policy as the Least Expected Workload policy, denoted by  $\text{LEW}(d)$ .

**Remark 55.** *As our main objective is to study load balancers which are not aware of the job size distribution, we only use this policy to see how it compares with the other strategies. This policy may be viewed as an idealized version of what the other policies attempt to do: avoid queues with a large expected workload. As such, we view the performance of  $\text{LEW}(d)$  as the goal of what the other policies try to achieve. In Section 7.8 we find that the performance of some policies indeed closely approximates that of  $\text{LEW}(d)$ .*

## 7.6 Model analysis

In this section we use the cavity approach presented in [20] to study the performance of the load balancing algorithms introduced in the previous section. We refer to the attained service time of a job at a server as its *age*  $a$ . Note that in our case the age  $a$  of a job does not include the waiting time of the job, only the time it has been in service.

### 7.6.1 Description of the cavity map

The cavity process intends to capture the evolution of a single server assuming asymptotic independence among servers (see below). The state of a single server is captured by the service phase ( $j$ ), the age ( $a$ ) of the job in service and the queue length ( $\ell$ ). The state of a server is thus denoted as a triple  $(j, a, \ell)$ . As arrivals occur according to a Poisson( $\lambda N$ ) process and each arrival has a probability of  $\frac{d}{N}$  to select any particular queue, the rate at which a server is selected as one of the  $d$  random servers is equal to  $\lambda d$ , we refer to this rate as the *potential* arrival rate. At each potential arrival,  $d - 1$  independent copies of the queue at the cavity are considered. The state  $(j, a, \ell)$  for each of these  $d - 1$  independent copies at time  $t$  has the same distribution as the distribution of the queue at the cavity at time  $t$ . The potential arrival is

assigned to one of the  $d$  selected queues based on the  $(k, \ell)$  values reported by the queue at the cavity and the  $d-1$  independent queues. Thus, the *actual* arrival rate depends on both the queue length  $\ell$  and the layer  $k$  containing the age  $a$  of the job at the head of the cavity queue at the time of a potential arrival. Let us denote this value by  $\lambda_{act}(k, \ell)$ . In general, we find that the number of jobs present in the queue at the cavity increases by one with a rate equal to  $\lambda_{act}(k, \ell)$ , whilst its age  $a$  continuously increases at rate one, and the service phase  $j$  evolves as dictated by the phase type distribution  $(\alpha, A)$ . When a job completes service, the age  $a$  jumps to zero,  $\ell$  decreases by one and a new job starts service if present.

In order to formally prove that the results presented in this chapter correspond to the limiting behavior as the number of servers  $N$  tends to infinity, the modularized program presented in [20] can be followed:

- a. Asymptotic Independence.** Demonstrate  $\Pi^N \rightarrow \Pi$  as  $N \rightarrow \infty$ , where  $\Pi^N$  is the stationary distribution for the studied policy with  $N$  queues and  $\Pi$  is a stationary and ergodic distribution on  $(\{1, \dots, m\} \times [0, \infty) \times \mathbb{N})^\infty$ . Show that the limit  $\Pi$  is unique. Show that, for every  $k$ :

$$\Pi^{(k)} = \bigotimes_{i=1}^k \Pi^{(1)},$$

where  $\Pi^{(k)}$  is  $\Pi$  restricted to its first  $k$  coordinates.

- b. The queue at the cavity.** Let  $A_{act}^N$  denote the process of actual arrivals to the first queue. Show that  $A_{act}^N \rightarrow \lambda_{act}$  in distribution as the number of servers  $N$  tends to infinity.
- c. Calculations.** Given  $\lambda_{act}$ , the actual arrival rates, analyse the queue at the cavity in the large  $N$  limit using queueing techniques to express  $\Pi^{(1)}$  as a function of  $\lambda_{act}$ :

$$\Pi^{(1)} = T(\lambda_{act}).$$

Moreover, the arrival rate is also determined by the state of a server  $\Pi^{(1)}$  we thus have:

$$\lambda_{act} = H(\Pi^{(1)}).$$

We then must determine the fixed point of the cavity map, that is, solve the equation  $\Pi^{(1)} = T(H(\Pi^{(1)}))$  to obtain  $\Pi^{(1)}$ .

In this work, we focus on  $\mathbf{c}$ , the computational step of the program. We present a numerical method to compute  $\Pi^{(1)}$  for the load balancing policies in Section 7.5. For ease of notation we denote  $\Pi^{(1)}$  as  $\pi$ . We validate our results using simulation to show that the obtained solutions indeed correspond to the system under study (as  $N \rightarrow \infty$ ).

### 7.6.2 Obtaining the steady state

In this section we indicate how to compute  $\pi$  given  $\lambda_{act}$ . Given the discussion in the previous subsection, this step corresponds to determining the steady state of a queueing system with the following characteristics:

1. A single server queue that serves jobs in a FCFS manner.
2. Service times of a job follow an order  $m$  phase type distribution with parameters  $(\alpha, A)$ .
3. Poisson arrivals with a rate that depends on the queue length  $\ell$  and the attained service time  $a$  (if the queue is busy).
4. No arrivals when the queue length equals some large value  $B$ .

More specifically, let  $0 = c_0 < c_1 < \dots < c_r < c_{r+1} = \infty$ , then the dependence on  $a$  is such that the Poisson arrival rate is only influenced by  $k$ , where  $k$  is the unique index such that  $a \in (c_{k-1}, c_k]$ . In other words, the queueing system under consideration is fully determined by  $B$ ,  $(\alpha, A)$  and a set of arrival rates  $\{\lambda_0\} \cup \{\lambda_{k,\ell} | k \in \{1, \dots, r+1\}, \ell \in \{1, \dots, B-1\}\}$ . The reason why we can assume an arrival rate equal to zero when the queue length equals some  $B < \infty$  is explained further on.

For the purpose of determining a fixed point of the cavity map, it suffices to develop a method to compute the steady-state probabilities  $\pi_{k,\ell,j}^{busy}$  that the service phase equals  $j$ , queue length equals  $\ell$  and the attained service time  $a$  belongs to  $(c_{k-1}, c_k]$  given that the queue is busy, for  $k = 1, \dots, r+1$ ,  $j = 1, \dots, m$  and  $\ell = 1, \dots, B$ . These probabilities are not affected by  $\lambda_0$ .

We start by defining a finite state discrete-time Markov chain on the state space

$$\mathcal{S} = \{(k, \ell, j) | k = 0, \dots, r; \ell = 1, \dots, B; j = 1, \dots, m\},$$

by observing the queue whenever the attained service time equals  $c_k$  for some  $k = 0, \dots, r$ . Note that this implies that we observe the queueing system

exactly  $i$  times for a job with length  $z \in [c_{i-1}, c_i]$ : once when the service starts and  $i - 1$  times during its service.

Given that this DTMC is in state  $(k, \ell, j)$  we can have two types of transitions:

1. The job in service remains in service for at least  $c_{k+1} - c_k$  more time and the chain transitions to a state of the form  $(k+1, \ell', j')$  with  $\ell' \geq \ell$ .
2. The length of the job in service is below  $c_k$  and the chain transitions to a state of the form  $(0, \ell', j')$  with  $\ell' \geq \ell - 1$ .

Hence, if we order the states in  $\mathcal{S}$  in lexicographical order, the transition probability matrix  $P^{DTMC}$  has the following form

$$P^{DTMC} = \begin{pmatrix} \Xi^{(1)} & \Lambda^{(1)} & & \\ \Xi^{(2)} & & \Lambda^{(2)} & \\ \vdots & & & \ddots \\ \Xi^{(r)} & & & \Lambda^{(r)} \\ \Xi^{(r+1)} & & & \end{pmatrix}, \quad (7.8)$$

where  $\Xi^{(k)}$  and  $\Lambda^{(k)}$ , for  $k = 1, \dots, r + 1$ , are square matrices of size  $mB$ . To express these matrices we define the size  $mB$  matrix

$$G = (I_B \otimes \mu) \cdot \left( \begin{pmatrix} 1 & 0 & \dots & 0 \\ 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ 0 & 0 & \ddots & 1 \end{pmatrix} \otimes \alpha \right), \quad (7.9)$$

where  $\mu$  was defined as  $-A\mathbf{1}$  and the size  $mB$  matrices

$$F^{(k)} = I_B \otimes A + \begin{pmatrix} -\lambda_{k,1} & \lambda_{k,1} & & \\ & -\lambda_{k,2} & \lambda_{k,2} & \\ & & \ddots & \ddots \\ & & & -\lambda_{k,B-1} & \lambda_{k,B-1} \\ & & & & -\lambda_{k,B} \end{pmatrix} \otimes I_m, \quad (7.10)$$

for  $k = 1, \dots, r + 1$ , with  $\lambda_{k,B} = 0$ .

The matrix  $G$  contains the rates at which  $(\ell, j)$  changes due to a service completion, which does not depend on the attained service time. The matrix

$F^{(k)}$  captures the evolution of the queue length and service phase when there is no service completion and the attained service time is between  $c_{k-1}$  and  $c_k$ . Based on these interpretations we have, for  $k = 1, \dots, r+1$ ,

$$\Lambda^{(k)} = e^{F^{(k)}(c_k - c_{k-1})}, \quad (7.11)$$

and  $\Xi^{(k)} = \Psi^{(k)}G$ , where

$$\Psi^{(k)} = \int_0^{c_k - c_{k-1}} e^{F^{(k)}\delta} d\delta = (I_{mB} - \Lambda^{(k)}) \cdot (-F^{(k)})^{-1}. \quad (7.12)$$

Let  $\hat{\pi}^{(k)}$  be the size  $mB$  vector holding the steady state probabilities of the DTMC characterized by  $P^{DTMC}$  corresponding to the states of the form  $(k, \ell, j)$ , then due to the structure of  $P^{DTMC}$ , we have

$$\hat{\pi}^{(k)} = \hat{\pi}^{(k-1)} \Lambda^{(k)} = \hat{\pi}^{(0)} \prod_{i=1}^k \Lambda^{(i)},$$

for  $k = 1, \dots, r+1$ . Using the first balance equation, the vector  $\hat{\pi}^{(0)}$  is therefore found as

$$\hat{\pi}^{(0)} = \hat{\pi}^{(0)} \sum_{k=0}^r \left( \prod_{i=1}^k \Lambda^{(i)} \right) \Xi^{(k+1)}.$$

As  $\Xi^{(k)} = \Psi^{(k)}G$ , the above equation can be restated as  $\hat{\pi}^{(0)} = \hat{\pi}^{(0)} \Omega^{(1)}$ , where  $\Omega^{(r+1)} = \Psi^{(r+1)}G$  and

$$\Omega^{(k)} = \Psi^{(k)}G + \Lambda^{(k)}\Omega^{(k+1)}.$$

Hence, the time complexity to compute the steady state probabilities of the DTMC characterized by  $P^{DTMC}$  equals  $O(r m^3 B^3)$ .

As stated before, our aim is to compute the steady state probabilities  $\pi_{k,\ell,j}^{busy}$  that the service phase equals  $j$ , queue length equals  $\ell$  and the attained service time  $a$  belongs to  $(c_{k-1}, c_k]$  given that queue is busy. These probabilities can now be computed from the steady state probabilities  $\hat{\pi}_{k,\ell,j}$  of the DTMC, by looking at the amount of time that the queue length and service phase equal  $(\ell', j')$  in between two points of observation of the DTMC. Note that entry  $((\ell, j), (\ell', j'))$  of the matrix  $\Psi^{(k)}$  contains the expected amount of time that the queue length equals  $\ell'$  and the server is in phase  $j'$  during a single

transition of the DTMC from state  $(k - 1, \ell, j)$  to any other state. We therefore have

$$\pi_{k,\ell',j'}^{busy} = \nu(\hat{\pi}_{(k-1)}\Psi^{(k)})_{(\ell',j')},$$

for  $k = 1, \dots, r + 1$ ,  $\ell' = 1, \dots, B$  and  $j' = 1, \dots, m$ , with  $\nu$  being the normalization constant. As each job brings  $\mathbb{E}[X] = 1$  work on average, the queue at the cavity is empty with probability  $1 - \lambda$ . This allows us to express the stationary distribution  $\pi$  by setting  $\pi_0 = \lambda$  and renormalizing  $\pi^{busy}$  to sum to  $\lambda$ .

### 7.6.3 Determining the arrival rate

In this section we indicate how to determine  $\lambda_{act}$  given  $\pi$ . For any  $(k, \ell)$ , we define:

$$\mathcal{A}(k, \ell) = \{(k', \ell') \mid \xi(k', \ell') \geq \xi(k, \ell)\}, \mathcal{B}(k, \ell) = \{(k', \ell') \mid \xi(k', \ell') > \xi(k, \ell)\}. \quad (7.13)$$

We denote  $x_{k,\ell} = \sum_j \pi_{k,\ell,j}$  the probability that, in the stationary regime, the queue at the cavity is in layer- $k$  with queue length  $\ell$ . Furthermore we let:

$$u_{k,\ell} = \sum_{(k',\ell') \in \mathcal{A}(k,\ell)} x_{k',\ell'} \quad v_{k,\ell} = \sum_{(k',\ell') \in \mathcal{B}(k,\ell)} x_{k',\ell'} \quad w_{k,\ell} = u_{k,\ell} - v_{k,\ell}. \quad (7.14)$$

**Remark 56.** When  $\xi$  is injective, we simply have  $w_{k,\ell} = x_{k,\ell}$ .

We obtain the arrival rate in layer- $k$  with queue length  $\ell$  in the following Proposition:

**Proposition 98.** *The arrival rate to the queue at the cavity, given its queue length  $\ell$  and service layer- $k$  is given by:*

$$\lambda_{act}(k, \ell) = \frac{\lambda}{w_{k,\ell}} (u_{k,\ell}^d - v_{k,\ell}^d). \quad (7.15)$$

*Proof.* Whenever a job arrives to the system, it samples the queue at the cavity and  $d - 1$  i.i.d. servers with distribution  $x_{k,\ell}$ . Potential arrivals occur with rate  $\lambda \cdot d$  and each potential arrival joins the queue at the cavity with probability  $\frac{1}{j+1}$  if  $j$  of the chosen servers are in some state  $(k', \ell')$  with

$\xi(k', \ell') = \xi(k, \ell)$ , while all other servers are in state  $(k', \ell')$  with  $\xi(k', \ell') > \xi(k, \ell)$ . We obtain:

$$\begin{aligned}\lambda_{act}(k, \ell) &= \lambda d \sum_{j=0}^{d-1} \frac{1}{j+1} \binom{d-1}{j} \cdot w_{k,\ell}^j \cdot v_{k,\ell}^{d-1-j} \\ s &= \frac{\lambda}{w_{k,\ell}} ((w_{k,\ell} + v_{k,\ell})^d - v_{k,\ell}^d),\end{aligned}\tag{7.16}$$

which simplifies to (7.15), finishing the proof.  $\square$

**Remark 57.** While (7.15) is the more elegant formula, the expression in (7.16) is actually more stable for numerical purposes.

#### 7.6.4 Iterative procedure

In this section we show how to compute the fixed point  $\pi$  described in Section 7.6.1 using Sections 7.6.2 and 7.6.3. The procedure in Section 7.6.2 corresponds to the function  $T$  such that the stationary distribution of the queue at the cavity  $\pi = T(\lambda_{act})$ . The result in Section 7.6.3 corresponds to the map  $H$  such that  $\lambda_{act} = H(\pi)$ . The fixed point of the cavity map is given by the fixed point  $\pi$  which satisfies  $\pi = T(H(\pi))$ . To this end, we propose the following iterative scheme to compute  $\pi$ :

1. Pick some  $\pi^{(0)}$  (e.g.  $\pi_0^{(0)} = 1$ ), set some tolerance  $tol$  and  $n = 0$ .
2. Compute  $\pi^{(n+1)} = T(H(\pi^{(n)}))$ .
3. If  $\|\pi^{(n+1)} - \pi^{(n)}\|_1 = \sum_{k,\ell,j} |\pi_{k,\ell,j}^{(n+1)} - \pi_{k,\ell,j}^{(n)}| < tol$  we accept  $\pi^{(n+1)}$ , otherwise increment  $n$  by one and return to step 2.

Throughout our numerical experiments we typically employ the tolerance  $tol = 10^{-10}$ . Moreover, as illustrated in Figure 7.4,  $\|\pi^{(n+1)} - \pi^{(n)}\|_1$  decreases exponentially in  $n$  and the number of iteration required is typically below 50, where each iteration requires  $O(m^3 B^3 r)$  time. If we start with  $\pi_0^{(0)} = 1$ , then setting  $B = n$  during the  $n$ -th iteration suffices. We can even make use of a smaller  $B$  value as the arrival rates  $\lambda_{k,\ell}$  decrease very rapidly in  $\ell$  for most policies considered.

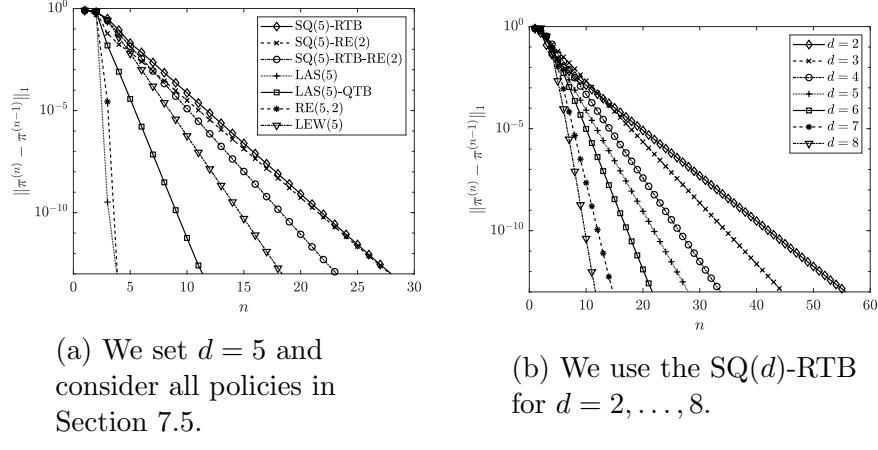


Figure 7.4: Convergence of  $\|\pi^{(n)} - \pi^{(n-1)}\|_1$  for  $\lambda = 0.8$ ,  $\Delta = 0.1$  and HEXP(10, 1/2) job sizes.

### 7.6.5 Obtaining performance measures

Given the stationary distribution  $\pi$  obtained in Section 7.6.4, we show how to obtain the performance measures associated to our model. In particular we are interested in the average queue length, response time and waiting time. The expected queue length can easily be computed as  $\mathbb{E}[Q] = \sum_{k,\ell,j} \ell \cdot \pi_{k,\ell,j}$ , while the expected response time can then be derived using Little's Law  $\mathbb{E}[R] = \mathbb{E}[Q]/\lambda$ . The expected waiting time is then given by  $\mathbb{E}[W] = \mathbb{E}[R] - 1$ .

In order to compute the waiting time distribution, we denote by  $J_{k,\ell,j}$  the probability that a random job arriving to the system joins a queue with length  $\ell$  for which the job at the head of the queue is in phase  $j$  and resides in layer- $k$ . It is not hard to see (similar to the proof of Proposition 98) that:

$$J_{k,\ell,j} = d \cdot \pi_{k,\ell,j} \cdot \sum_{s=0}^{d-1} \frac{1}{s+1} \binom{d-1}{s} w_{k,\ell}^s \cdot v_{k,\ell}^{d-1-s} = \frac{\pi_{k,\ell,j}}{w_{k,\ell}} \cdot (u_{k,\ell}^d - v_{k,\ell}^d). \quad (7.17)$$

From this one easily computes the probability of joining a queue with length  $\ell$  for which the job currently being served is in phase  $j$ :  $J_{\ell,j} = \sum_k J_{k,\ell,j}$ . Let us denote by  $X$  a generic job size variable, and by  $X_j$  a generic random Phase Type random variable with rate matrix  $A$  which starts in phase  $j$ . Associated with these values we denote  $X_{\ell,j}$ , the convolution of  $X_j$  with  $\ell-1$

i.i.d. copies of  $X$ . We find that the waiting time distribution is given by:

$$\bar{F}_W(w) = \sum_{\ell \geq 1} \sum_j J_{\ell,j} \bar{F}_{X_{\ell,j}}(w). \quad (7.18)$$

From this it is not hard to obtain the response time distribution:

$$\bar{F}_R(w) = \sum_{\ell \geq 1} \sum_j J_{\ell,j} \bar{F}_{X_{\ell+1,j}}(w) + (1 - \lambda^d) \bar{F}_X(w). \quad (7.19)$$

### 7.6.6 Job size distribution

In real systems job sizes are known to be highly variable and a significant part of the total workload is often offered by a small fraction of long jobs, while the remaining workload consists mostly of short jobs (e.g., [74, 27, 28]). A measure for the variability of a distribution is the SCV, which is defined as  $\frac{\text{Var}(X)}{\mathbb{E}[X]^2}$ . The SCV of an exponential random variable is exactly equal to one, while measurements in real systems reveal much higher SCVs (e.g., [45, Chapter 20]). Therefore we use the class of Mixed Erlang (MErlang) distributions to investigate the performance of the proposed policies.

The parameters of the MErlang distribution are set such that we can vary the SCV in a systematic manner as well as the fraction  $f$  of the workload offered by the small jobs. More precisely we fix a value of  $k$ , with probability  $p$  a job is a type-1 job and has an Erlang( $k, k \cdot \mu_1$ ) length with  $\mu_1 > 1$  and with the remaining probability  $1-p$  a job is a type-2 job and has an Erlang( $k, k \cdot \mu_2$ ) length with  $\mu_2 < 1$ . Hence, the type-2 jobs are longer on average and we therefore sometimes refer to the type-2 jobs as the *long* jobs. Note that when  $k = 1$  the MErlang distribution is an order 2 hyperexponential distribution. In order to set the parameters of the MErlang distribution, we first pick some value for the parameter  $k$  and subsequently, the parameters  $p, \mu_1$  and  $\mu_2$  are set such that the following three values are matched:

- the mean job length (set to one),
- the SCV and
- the fraction  $f$  of the workload that is offered by the type-1 jobs.

Due to [32, Equation (4)] one finds that for  $p, \mu_1$  and  $\mu_2$  fixed, the SCV as a function of  $k$  is such that  $(SCV(k) + 1)/(1 + 1/k)$  is a constant equal

Setup	$N = 10$	$N = 50$	$N = 100$	$N = 500$
SQ(3)-RTB	1.620430923	1.040911774	0.982958771	0.927865971
SQ(5)-RE(2)	3.308384612	1.896950514	1.722976246	1.601744382
SQ(10)-RTB-RE(2)	2.114036541	0.363542917	0.237407164	0.156558754
LAS(2)	4.868804414	3.967386221	3.850319215	3.77891573
LAS(7)-QTB	7.623873752	1.48914112	0.990905434	0.698607573
RE(6, 2)	3.45247775	1.782071768	1.583839017	1.429217462
LEW(8)	6.068363318	1.549444444	1.131295810	0.880167915
Setup	$N = 1000$	$N = 2000$	Cavity Method	
SQ(3)-RTB	0.921726545	0.918214211	0.9172330056	
SQ(5)-RE(2)	1.581067397	1.565666044	1.564915051	
SQ(10)-RTB-RE(2)	0.145208955	0.138413413	0.136679150	
LAS(2)	3.774355410	3.755167910	3.715613554	
LAS(7)-QTB	0.682478855	0.652344141	0.646230433	
RE(6, 2)	1.408752332	1.393480643	1.384668047	
LEW(8)	0.848402363	0.827531301	0.826663214	

Table 7.1: Table of the expected waiting time for finite  $N$  and the cavity method ( $N = \infty$ ).

to  $(p/\mu_1^2 + (1-p)/\mu_2^2)/(p/\mu_1 + (1-p)/\mu_2)^2$ . Hence, if we define  $\widetilde{SCV} = 2\frac{\widetilde{SCV}+1}{1+\frac{1}{k}} - 1$ , we can match  $(1, SCV, f)$  for a general  $k$  by matching  $(1, \widetilde{SCV}, f)$  for a hyperexponential distribution as in Section 1.4, that is , set  $\mu_1, \mu_2$  as:

$$\mu_i = \frac{\widetilde{SCV} + (4f - 1) + (-1)^{i-1} \sqrt{(\widetilde{SCV} - 1)(\widetilde{SCV} - 1 + 8f\bar{f})}}{2f(\widetilde{SCV} + 1)},$$

with  $\bar{f} = 1 - f$  and  $p = \mu_1 f$ .

We note that this distribution has a PH representation with  $\alpha \in \mathbb{R}^{2 \cdot k}$  defined by  $\alpha_1 = p$ ,  $\alpha_{k+1} = 1 - p$  and  $\alpha_i = 0$  for  $i \neq 1, k+1$ . Furthermore we define the transition matrix  $A \in \mathbb{R}^{2 \cdot k}$  by stating its non-zero elements:  $A_{i,i} = -k \cdot \mu_1$ ,  $A_{k+i,k+i} = -k \cdot \mu_2$  for  $i = 1, \dots, k$  and  $A_{i,i+1} = k \cdot \mu_1$ ,  $A_{k+i,k+i+1} = k \cdot \mu_2$  for  $i = 1, \dots, k-1$ . Throughout, we denote this job size distribution as MERlang( $SCV, f, k$ ) and as HEXP( $SCV, f$ ) when  $k = 1$ .

## 7.7 Finite system accuracy

In this section we compare the mean waiting time that corresponds to the fixed point of the cavity map with simulation experiments. The simulation setup is identical to the model, except that the number of servers  $N$  is finite. For each policy in Section 7.5 we selected some arbitrary parameter setting and varied the number of servers from  $N = 10$  to  $N = 2000$ . All simulation runs simulate the system up to time  $t = 10^7/N$  and use a warm-up period of 30%. Each simulation is the mean of 40 runs. The simulation experiments were performed using the following parameter settings:

- SQ(3)-RTB :  $\lambda = 0.7$ ,  $\Delta = 0.01$  and MErlang(10, 1/2, 2) job sizes.
- SQ(5)-RE(2) :  $\lambda = 0.8$  and HEXP(10, 1/10) job sizes.
- SQ(10)-RTB-RE(2) :  $\lambda = 0.8$ ,  $\Delta = 0.1$  and MErlang(15, 1/3, 5) job sizes.
- LAS(2) :  $\lambda = 0.6$ ,  $\Delta = 0.5$  and HEXP(20, 1/4) job sizes.
- LAS(7)-QTB :  $\lambda = 0.9$ ,  $\Delta = 0.01$  and MErlang(20, 2/3, 5) job sizes.
- RE(6, 2) :  $\lambda = 0.8$  and HEXP(10, 1/4) job sizes.
- LEW(8) :  $\lambda = 0.9$ ,  $\Delta = 0.5$  and HEXP(15, 1/3) job sizes.

The results are summarized in Table 7.1. We observe that for all examples there is indeed convergence of the mean waiting time, rendering our models to be accurate for high values of  $N$ .

## 7.8 Numerical experiments

Throughout this section we compare the proposed policies relative to the  $SQ(d)$  policy. Therefore, we are interested in the relative improvement of the proposed policies, in particular we focus on the quantity:

$$E_{W,rel,P} = \frac{\mathbb{E}[W^{(SQ(d))}] - \mathbb{E}[W^{(P)}]}{\mathbb{E}[W^{(SQ(d))}]}, \quad (7.20)$$

where  $\mathbb{E}[W^{(P)}]$  denotes the expected waiting time for some policy  $P$ . This value denotes how much additional waiting time using the  $SQ(d)$  policy

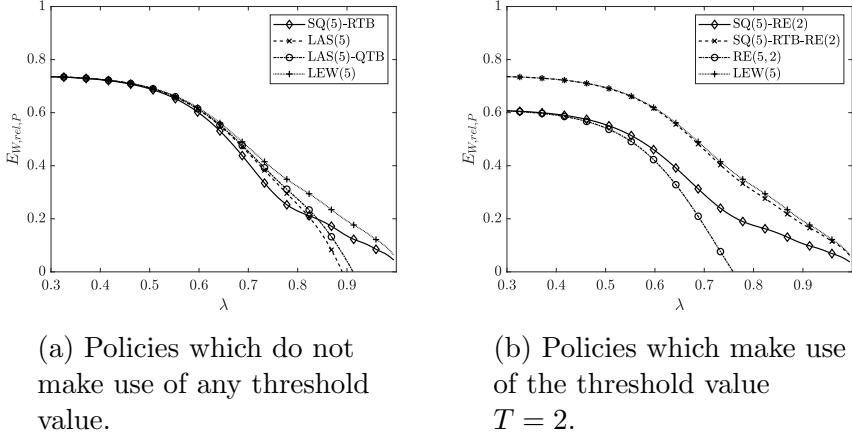


Figure 7.5: Plots of the improvement in mean waiting time (see also (7.20)) as a function of  $\lambda$  for  $d = 5$ ,  $\Delta = 0.1$  and  $\text{HEXP}(10, 1/2)$  job sizes.

yields. Note that if one was mainly interested in the relative improvement in the response time, one would compute:

$$E_{R,rel,P} = \frac{\mathbb{E}[R^{(SQ(d))}] - \mathbb{E}[R^{(P)}]}{\mathbb{E}[R^{(SQ(d))}]} = \frac{\mathbb{E}[W^{(SQ(d))}] - \mathbb{E}[W^{(P)}]}{\mathbb{E}[W^{(SQ(d))}] + 1},$$

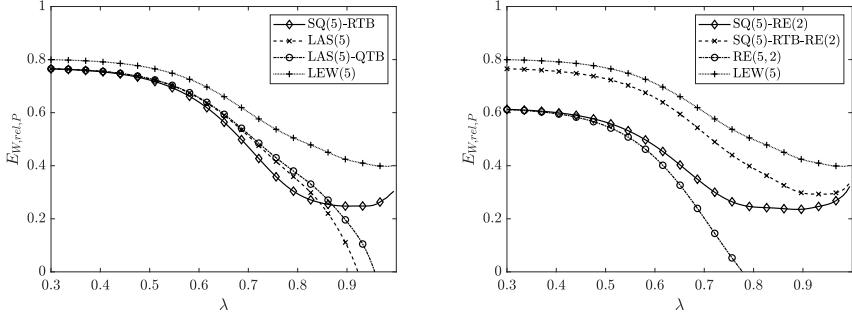
which is simply a flattened version of  $E_{W,rel,P}$ . As  $\mathbb{E}[R] = \mathbb{E}[W] + 1$ , as we are mainly interested in the amount *delay* a job experiences, we focus on  $E_{W,rel,P}$  as defined in (7.20). Throughout our numerical experimentation, we employ one central example as a base case in order to investigate the effect different parameters have on  $E_{W,rel,P}$  for the policies  $P$  discussed in Section 7.5. As the base case, we take  $d = 5$ ,  $\Delta = 0.1$  and  $\text{HEXP}(10, 1/2)$  distributed job sizes. In Figure 7.5 we observe the evolution of  $E_{W,rel,P}$  as a function of the arrival rate  $\lambda$ . From Figure 7.5, we can already make quite a few observations:

- The relative improvement we obtain from using the attained service time information is significant, with relative improvements in the waiting time close to 75%. However, the improvement decreases as the arrival rate  $\lambda$  approaches one. This makes sense as for a higher value of  $\lambda$  the queues become longer and queue length information becomes more important than attained service time information.

- All policies which mainly focus on the queue length information improve the performance for all  $\lambda$ , as such they can be viewed as enhanced  $SQ(d)$  policies. For the policies which mainly use the attained service time to distribute jobs, we observe that there exists some  $\lambda_{\max}$  such that these policies outperform  $SQ(d)$  for all  $\lambda < \lambda_{\max}$  while they are outperformed by  $SQ(d)$  when  $\lambda > \lambda_{\max}$ .
- Many of our policies perform as good as  $LEW(d)$  for loads up to 0.6, while these policies do not make use of the job size distribution. The performance of the  $SQ(5)$ -RTB-RE(2) policy is very close to that of  $LEW(d)$  for all  $\lambda$ , where the threshold  $T = 2$  was chosen quite arbitrarily (cf. Section 7.8.7).
- The improvement has a plateau at first, then the relative improvement tends to decrease with  $\lambda$ . For the policies which mainly employ the queue length information to distribute jobs, the curves become somewhat irregular for high loads. This irregularity is discussed when we consider the impact of larger  $d$  values (see Section 7.8.2).
- The low load limit ( $\lambda \rightarrow 0^+$ ) appears to be the same for all policies which solely rely on a threshold and those which rely on the attained service time information. This makes sense as in the low load limits, whenever a job has a non-zero waiting time, all  $d$  chosen queues have exactly one job waiting, meaning queue length information becomes irrelevant.

**Remark 58.** *As  $\lambda$  increases, the number of jobs in each queue increases. This reduces the value of knowing the attained service time of the job at the head of the queue. However, if we were to use a scheduling policy such as Processor Sharing, dispatchers may request information on the attained service time of all jobs in the queue, which could result in a more significant improvement at high loads.*

In the following sections, we reproduce Figure 7.5, where we change the value of one parameter. This allows us to investigate the effect this parameter has on our basic example.



(a) Policies which do not make use of any threshold value, this figure should be compared to Figure 7.5a.

(b) Policies which make use of the threshold value  $T = 2$ , this figure should be compared to Figure 7.5b.

Figure 7.6: Plots of the improvement in mean waiting time as a function of  $\lambda$  for  $d = 5$ ,  $\Delta = 0.1$ , MERlang( $10, 1/2, 5$ ) job sizes.

### 7.8.1 Impact of the number of phases $k$

In Figures 7.6a and 7.6b we repeat the experiment from Figure 7.5 with MERlang( $10, 1/2, 5$ ) job sizes. We clearly observe a greater improvement in performance than for the case  $k = 1$ . This may seem counter-intuitive as one could argue that our policies are tailor-made for job sizes which have a DHR. Indeed, when the hazard rate is decreasing, the expected remaining workload increases as a function of the attained service time. This is in correspondence with  $\xi(k, l)$  being non-decreasing in  $k$  for our policies. In Figure 7.7 we observe that the MERlang( $SCV, f, k$ ) only has a DHR for  $k = 1$ . However, as  $k$  increases both the small and large jobs become less variable and thus have a more predictable size. This implies that large jobs are somewhat easier to detect based on the attained service time, which explains why we have a greater improvement than with  $k = 1$ . Also, the gap between our policies and the LEW( $d$ ) policy increases. This is natural as it is sometimes better to assign a job to a server with a lower attained service time (as the hazard rate is non-decreasing), which is something the LEW( $d$ ) does automatically while this is never done by the other policies considered.

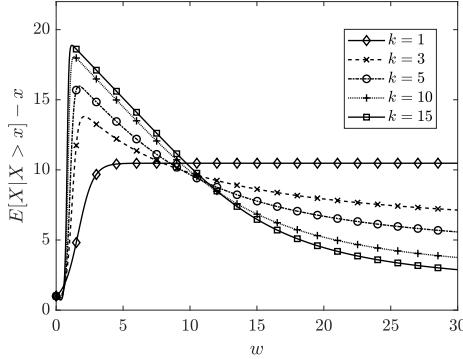


Figure 7.7: Expected remaining service time for a MErlang( $10, 1/2, k$ ) distribution with  $k = 1, 3, 5, 10$  and  $15$ .

### 7.8.2 Impact of the number of chosen servers $d$

In Figure 7.8 we reproduce Figure 7.5 after setting  $d = 20$  rather than  $d = 5$  (and thus setting  $\Delta = 0.1$ ,  $SCV = 10$  and  $f = 1/2$ ). We observe that the relative improvement increases significantly by increasing the value of  $d$ . In particular we make the following observations:

- Increasing the value of  $d$  increases the low load limit and extends the range of the load at which this value is maintained. This is expected as higher  $d$  values result in shorter queues.
- For the same reason, the value of  $\lambda_{max}$  up to which the policies which solely depend on the attained service time outperform SQ( $d$ ) increases for larger  $d$  values.
- The gap between the SQ( $d$ )-RTB-RE(2) and LEW( $d$ ) becomes negligible by increasing the value of  $d$  to 20.
- For some policies, the behaviour becomes irregular as  $\lambda$  approaches one. To understand the cause of this irregularity, we define  $\mathcal{T}_d(\lambda)$  as the expected number of jobs which have the least number of jobs pending amongst  $d$  randomly selected servers (given that all  $d$  chosen servers are busy). Letting  $u_k$  denote the probability that a server has  $k$  or more pending jobs, we find that:

$$\mathcal{T}_d(\lambda) = \frac{\sum_{k=1}^d k \cdot \sum_{\ell=1}^{\infty} p_{\ell,k}}{u_1^d}, \quad (7.21)$$

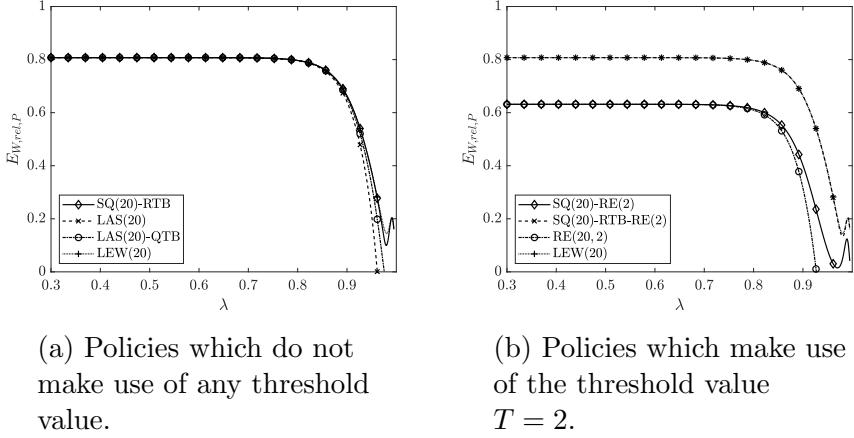
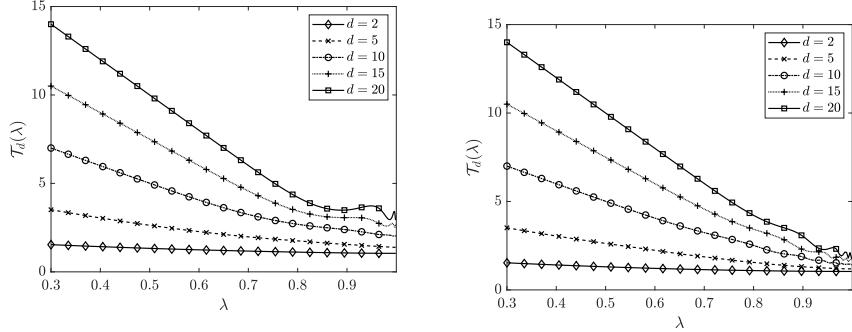


Figure 7.8: Plots of the improvement in mean waiting time as a function of  $\lambda$  for  $d=20$ ,  $\Delta = 0.1$  and HEXP(10, 1/2) job sizes. This figure should be compared to Figure 7.5.

with  $p_{\ell,k} = \binom{d}{k} (u_\ell - u_{\ell+1})^k \cdot u_{\ell+1}^{d-k}$  the probability that  $k$  servers have exactly  $\ell$  pending jobs and all other selected servers have at least  $\ell + 1$  jobs in their queue. In Figure 7.9 we plot the evolution of  $T_d(\lambda)$  for the SQ( $d$ ) policy with various values of  $d$ ,  $SCV = 10$  and  $f = 1/2$  in Figure 7.9a and  $f = 1/10$  in Figure 7.9b. We observe the same type of irregularity as in Figure 7.8. For  $T_d(\lambda)$  this behaviour can be understood as follows: for low loads the  $d$  chosen servers have queue length 1 and the number of chosen servers with queue length one then decreases as  $\lambda$  increases. If we look at the mean number of selected servers with queue length 2, then this mean increases with  $\lambda$  (except for very high  $\lambda$ ). For larger  $\lambda$  values, the expected number of chosen servers with the least number of jobs also depends on the mean number of servers with queue length 2 as all  $d$  chosen servers may have a queue length of at least 2. This causes the “waves” close to  $\lambda = 1$  which become more pronounced as  $d$  grows. These waves also explain the irregularity in Figure 7.8 (and Figure 7.5) as a higher value for  $T_d(\lambda)$  implies we have more ties in the queue length and the attained service time information is more valuable. In Figure 7.9b we observe that decreasing the value of  $f$  decreases the height of the waves.



(a) Here we set  $f = 1/2$ , this figure is used to motivate the behaviour in Figure 7.8 and 7.10 for  $\lambda \approx 1$ .

(b) Here we set  $f = 1/10$ , as such this setting can be seen to correspond to that considered in Figure 7.13.

Figure 7.9: Plots of  $T_d(\lambda)$  as a function of  $\lambda$  for the SQ( $d$ ) policy with various values of  $d$  and HEXP(10,  $f$ ) job sizes.

To further emphasize the aforementioned observations, we single out two policies which were used to create Figure 7.8 and show their performance as a function of  $\lambda$  for various values of  $d : 2, 5, 10, 15, 20$ . In Figure 7.10a we consider the SQ( $d$ )-RE(2) policy whilst in Figure 7.10b we consider the RE( $d$ , 2) policy. We can clearly see our observations being confirmed. In particular for SQ( $d$ )-RE(2), we see waves becoming larger as  $d$  increases, the plots of  $d = 15$  and  $d = 20$  even cross at some point.

### 7.8.3 Impact of the Squared Coefficient of Variation (SCV)

In Figure 7.11 we use the same parameter settings as in Figure 7.5, but change the SCV from 10 to 30 (i.e. we set  $d = 5$ ,  $\Delta = 0.1$  and  $f = 1/2$ ). As expected, increasing the SCV, increases the relative improvement made by using the attained service time information. However, the improvement is not as significant as one might expect, especially for larger values of  $\lambda$ . Therefore we investigate this further in Figure 7.12, where we show the relative improvement as a function of the SCV for  $\lambda = 0.7$ , we observe that there is a clear improvement if we have a higher SCV, but only up to some point. For

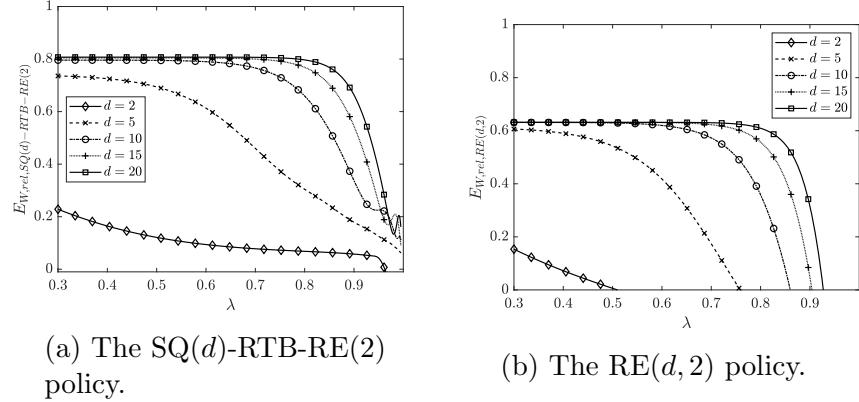


Figure 7.10: Plots of the improvement in mean waiting time as a function of  $\lambda$  for various values of  $d$ ,  $\Delta = 0.1$  and HEXP(10, 1/2) job sizes. This figure should be viewed as a supplement of Figure 7.8.

most policies we observe a strong improvement until  $SCV \approx 15$ , after which the improvement seems to flatten, or even decrease. The reason for the slight decrease is probably due to the fact that a higher SCV also results in longer queues and similar to higher  $\lambda$  values, this decreases the value of knowing the attained service time somewhat. Additional experiments showed that for smaller  $\lambda$  values the decrease as a function of the SCV occurs further on.

#### 7.8.4 Impact of the load from small jobs $f$

In Figure 7.13 we repeat the experiment in Figure 7.5, but we replace the value of  $f$  by  $f = 1/10$ . We observe that, somewhat surprisingly, having a small value for  $f$  appears to decrease the gain from using the attained service time information. This may be explained by the fact that, as the value of  $f$  decreases, the probability that all chosen servers are working on a large job increases. In particular for our example there is a probability around  $(9/10)^5 \approx 59\%$  that all chosen servers are currently working on a large job (given that all chosen servers are currently busy). That is, in 59% of cases, our policies reduce to either random routing or the standard SQ( $d$ ) policy. We emphasize this point using Figure 7.14, where we show the performance of our policies as a function of  $f$ , we clearly see how the improvement reaches a peak around  $f \approx 2/3$ . The performance decreases

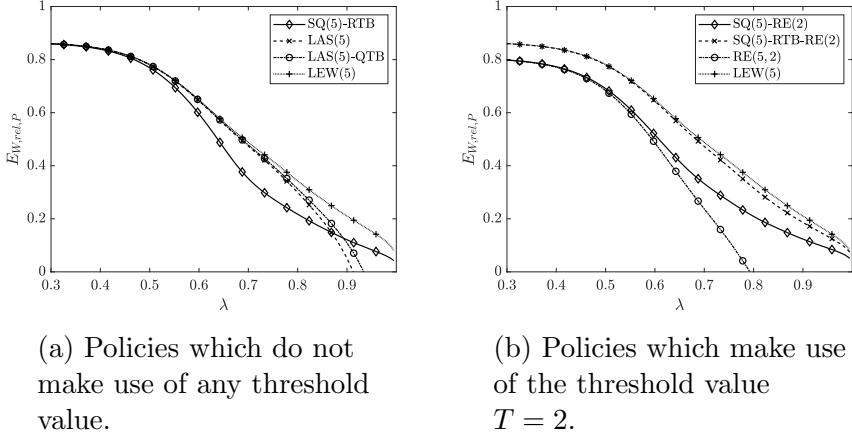


Figure 7.11: Plots of the improvement in mean waiting time as a function of  $\lambda$  for  $d = 5$ ,  $\Delta = 0.1$  and  $\text{HEXP}(30, 1/2)$  job sizes. This figure should be compared to Figure 7.5.

sharply as  $f \approx 0$  and  $f \approx 1$ , in the first case, (almost) all servers are working on large jobs, turning our policies into random routing resp.  $\text{SQ}(d)$  while for  $f \approx 1$ , almost all servers are working on short jobs, which again turns our policies into random routing resp.  $\text{SQ}(d)$ .

### 7.8.5 Tail of the waiting time distribution

In this section we take a closer look at the tail of the workload distribution  $\bar{F}_W(w) = \mathbb{P}\{W \geq w\}$ . In Figure 7.15 we show  $\bar{F}_W(w)$  as a function of  $w$  for the same setting as in Figure 7.5 with  $\lambda = 0.8$  (i.e.  $d = 5$ ,  $T = 2$  and  $\text{HEXP}(10, 1/2)$  job sizes). We observe that the tail behaviour is identical for all policies (and is the same as the tail of  $\text{SQ}(5)$ ). This result is not unexpected as the tail is heavily influenced by the job size distribution of the long jobs. This entails that studying the tail behaviour does not add much value to our discussion, therefore we keep our focus on the mean waiting time.

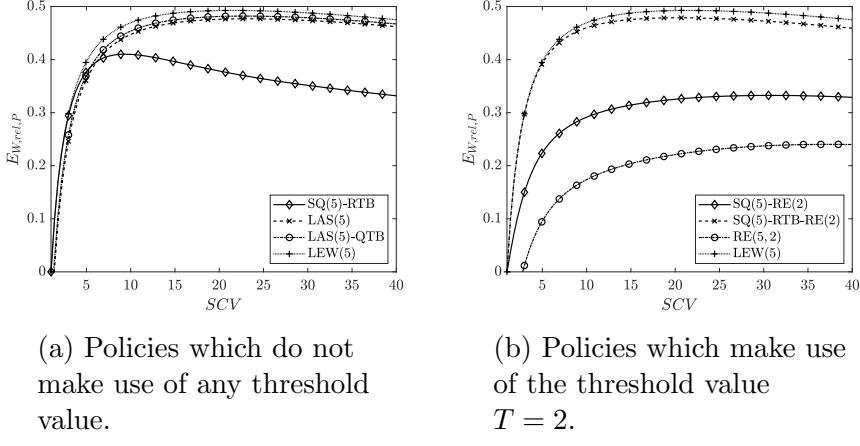


Figure 7.12: Plots of the improvement in mean waiting time as a function of the  $SCV$  for  $\lambda = 0.7$ ,  $d = 5$ ,  $\Delta = 0.1$  and  $HEXP(\mathbf{SCV}, 1/2)$  job sizes.

### 7.8.6 The granularity $\Delta$

We have always used the same granularity  $\Delta = 0.1$ , one could argue that in a real system the servers have more fine grained information on the attained service time of the job at the head of its queue. In Figure 7.16a we consider the same setting as in Figure 7.5 ( $d = 5$ ,  $T = 2$  and  $HEXP(10, 1/2)$  job sizes), but with  $\Delta = 0.01$ . We do not generate the plots associated to  $SQ(5)\text{-RE}(2)$  and  $RE(5, 2)$  as these policies are independent of the granularity (the server simply states whether or not it has exceeded the threshold  $T$ ).

With the exception of  $LAS(5)\text{-QTB}$ , we can hardly spot a difference with the plots in Figure 7.5. The exception for  $LAS(5)\text{-QTB}$  can be explained by noting that a smaller granularity  $\Delta$  implies fewer ties in the reported  $k$  value, meaning the queue length information is neglected more often and as  $\Delta$  tends to zero, the performance of  $LAS(5)\text{-QTB}$  converges to that of  $LAS(5)$  (while for a very large granularity its performance resembles  $SQ(d)\text{-RE}(T)$ ). We confirm our findings in Figure 7.16b, where we repeat the plots made in Figure 7.11, but with  $\Delta = 0.01$  rather than  $\Delta = 0.1$  (i.e.  $d = 5$ ,  $T = 2$  and  $HEXP(30, 1/2)$  job sizes).

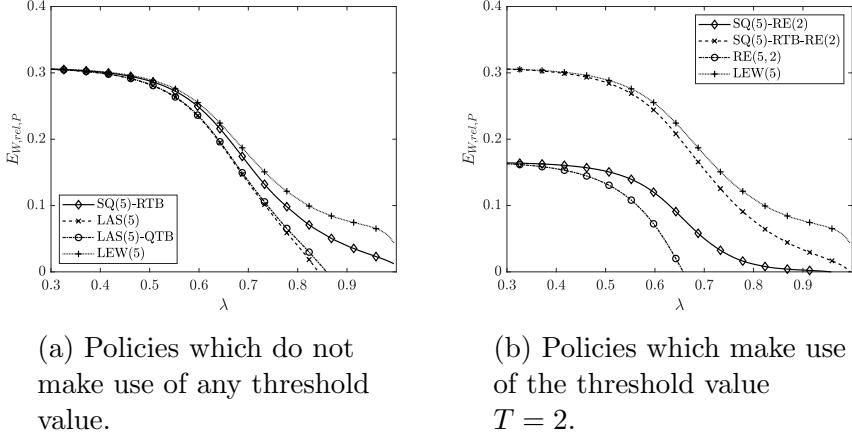


Figure 7.13: Plots of the improvement in mean waiting time as a function of  $\lambda$  for  $d = 5$ ,  $\Delta = 0.1$  and  $\text{HEXP}(10, 1/10)$  job sizes. This figure should be compared to Figure 7.5.

### 7.8.7 Choice of the threshold $T$

Thus far, we have always used a (somewhat arbitrary) threshold  $T = 2 \cdot \mathbb{E}[X]$ . With this choice we noticed that we obtain a significant improvement over the standard  $\text{SQ}(d)$  policy. However, it would make more sense if we chose a threshold which depends on the mean of the small jobs rather than the mean of all jobs. This way, our threshold is more directly linked to the probability that a job is long given that a server has been working on it for a time  $T$ . We now look at the impact of  $T$  on the performance of the policies which depend on a threshold value. To that end, we use the setting of Figures 7.5b and 7.13b with  $\lambda = 0.8$  (that is, we take  $d = 5$  and consider  $\text{HEXP}(10, f)$  with  $f$  equal to  $1/2$  and  $1/10$ ). Let  $X_1$  denote the job size distribution for the small jobs, then we have  $\mathbb{E}[X_1] \approx 0.53$  in the case of  $f = 1/2$ , while  $\mathbb{E}[X_1] \approx 0.12$  for  $f = 1/10$ . In Figure 7.17 we notice that the plots attain a maximum and this maximum appears to be close to the value  $T \approx 2\mathbb{E}[X_1]$ .

In Figure 7.18 we use this improved threshold value of  $T = 2\mathbb{E}[X_1]$  and show the improvement in mean waiting time as a function of the arrival rate  $\lambda$ . We observe that (comparing with Figures 7.5b and 7.13b) there is a noticeable improvement in performance. This is especially true for the case of  $f = 1/10$ , which is quite natural as for this case we now take  $T = 0.24$

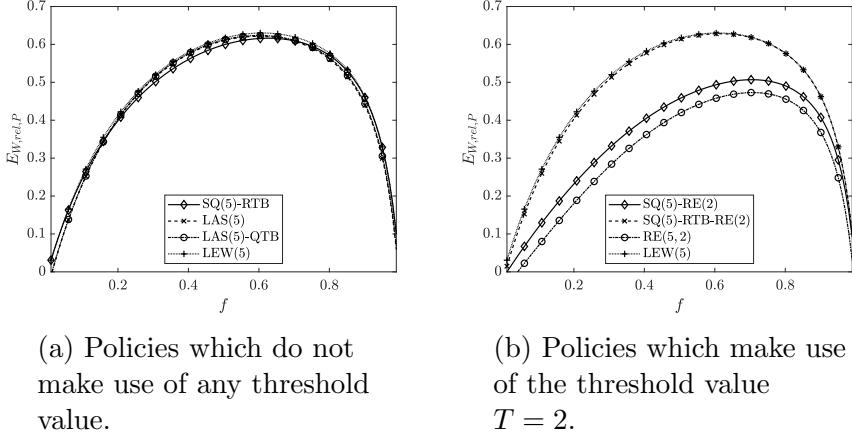


Figure 7.14: Plots of the improvement in mean waiting time as a function of  $f$  for  $\lambda = 0.6$ ,  $d = 5$ ,  $\Delta = 0.1$  and HEXP(10, f) job sizes. This figure should be viewed as a supplement of Figure 7.13.

instead of  $T = 2$  while for  $f = 1/2$  the threshold value of 2 is replaced by the value  $T = 1.06 > 0.24$ . However, we note that in both cases, the improvement was still significant for all considered policies even when we were using a suboptimal threshold value.

## 7.9 Conclusions and future work

In this chapter we showed that the mean waiting time of the  $SQ(d)$  policy can be significantly reduced using simple policies if the servers report the attained service time in addition to their queue length when the workload consists of a mixture of short and long jobs. The attained service time is a quantity that is easy to measure in FCFS servers. It is even possible to achieve a sizeable improvement over  $SQ(d)$  by simply setting a single threshold value  $T$  such that a server can flag a job as large when this threshold is exceeded. While choosing a good value for  $T$  may further improve performance, most policies are not too sensitive to  $T$  and there is a wide range of  $T$  values which achieve a notable performance improvement.

We did note that the improvement coming from the attained service time information generally decreases as the system load approaches one, as the

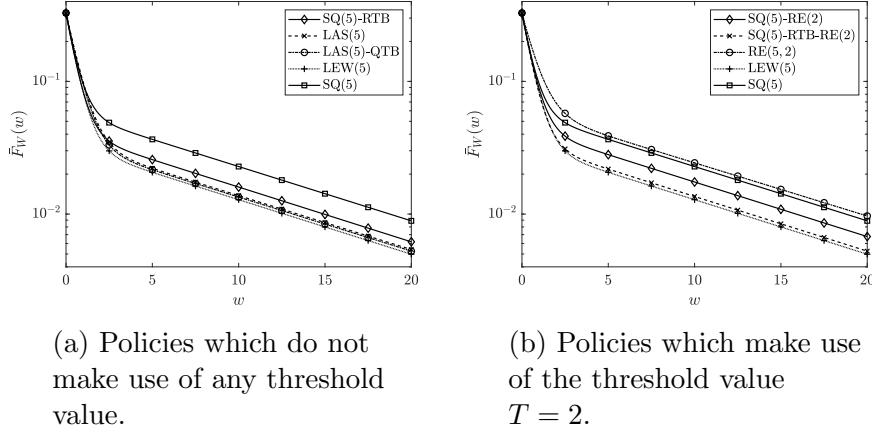
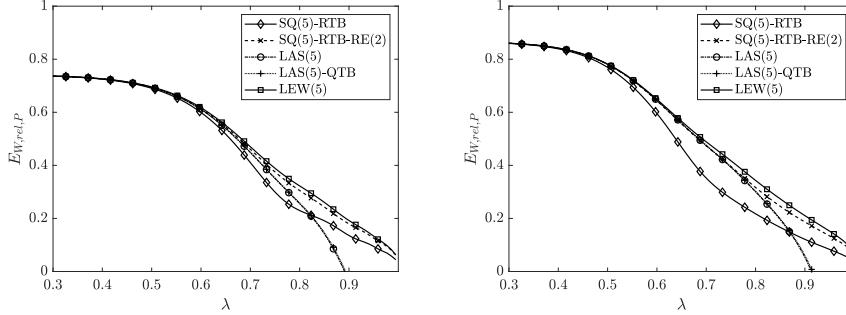


Figure 7.15: Plots of the tails of the waiting time distributions  $\bar{F}_W(w) = \mathbb{P}\{W \geq w\}$  for  $\lambda = 0.8$ ,  $\Delta = 0.1$ ,  $d = 5$  and HEXP(10, 1/2) job sizes. This figure should be seen as a supplement to Figure 7.5.

attained service time information becomes less valuable in the presence of long queues. However, if one were to use another scheduling policy which is aware of the attained service time of multiple jobs in its queue, larger gains may also be feasible at high load (e.g. using Processor Sharing).

In our experiments we have focused solely on job sizes which represent a system in which both large and small jobs arrive. We believe however that the insights obtained in our numerical experiments extend far beyond this class of distributions. Further, the approach developed in the chapter to assess the system performance can be used for any phase-type distribution (including fittings of heavy tailed distributions).

The main idea presented in this chapter can also be used in systems where there is some form of memory at the dispatcher. For instance, the approach of [86] could be adapted such that the dispatcher maintains both an upper bound on the number of jobs in each queue and a lower bound on the attained service time of the job at the head of each queue.



(a) For this plot, we set  $SCV = 10$ , as such it should be compared to Figure 7.5.

(b) For this plot, we set  $SCV = 30$ , as such it should be compared to Figure 7.11.

Figure 7.16: Plots of the improvement in mean waiting time as a function of  $\lambda$  for  $d = 5$ ,  $\Delta = 0.01$  and  $HEXP(SCV, 1/2)$  job sizes.

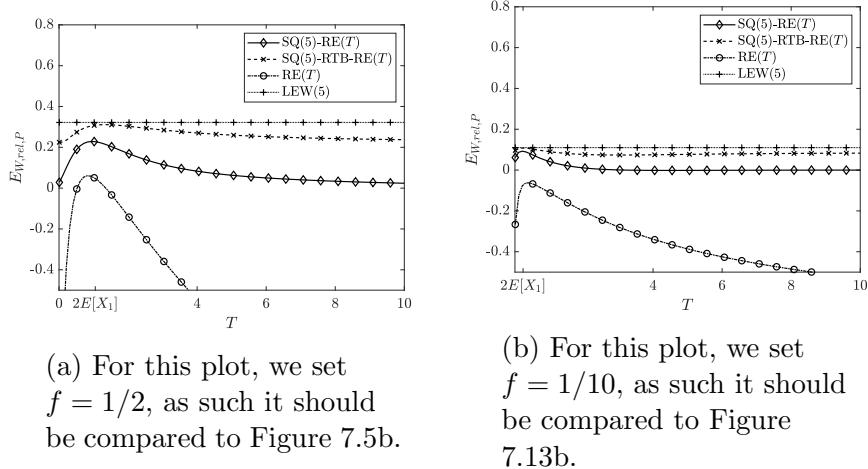
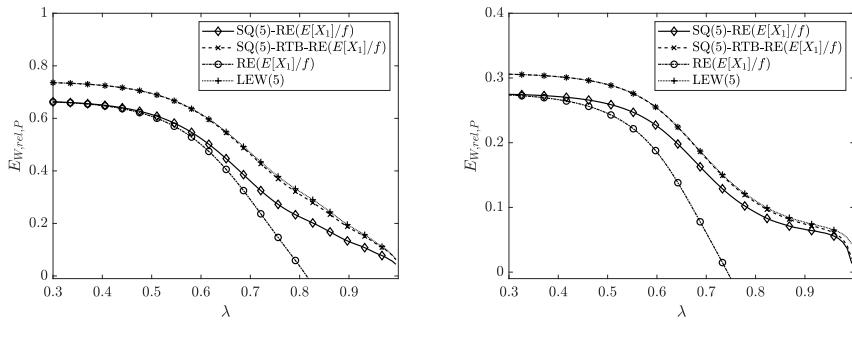


Figure 7.17: Plots of the improvement in mean waiting time as a function of the threshold  $T$  for  $\lambda = 0.8$ ,  $d = 5$ ,  $\Delta = 0.1$  and  $HEXP(10, f)$  job sizes. We denote the job size distribution of the small jobs by  $X_1$ .



(a) Plot for  $f = 1/2$ , this figure should be compared to Figure 7.5b.

(b) Plot for  $f = 1/10$ , this figure should be compared to Figure 7.13b.

Figure 7.18: Plots of the improvement in mean waiting time as a function of  $\lambda$  for  $d = 5$ ,  $\Delta = 0.1$ , HEXP(10,  $f$ ) job sizes and using the improved threshold value  $T = 2 \cdot \mathbb{E}[X_1]$ , with  $X_1$  the job size distribution of the small jobs.

# Chapter 8

## Open problems

There you stood on the edge of  
your feather  
Expecting to fly.

---

Neil Young

### 8.1 Refined mean field approximation

Denote by  $W_N^{(SQ(d))}$  the average queue length for a system of  $N$  homogeneous servers which distributes jobs using the  $SQ(d)$  policy. Further assume that job sizes are exponential and the system load is given by  $\rho = \lambda \cdot \mathbb{E}[X]$ .

In [43], it was shown for  $SQ(d)$  that there exists a (computable) value  $\zeta(N, \rho)$  such that:  $W_N^{(SQ(d))} + \zeta(N, \rho) - W^{(SQ(d))} = O(1/N^2)$ . This means that it is possible to refine the mean field approximation to obtain an  $O(1/N^2)$  approximation (one can go even further, c.f. [43]). It was noted that  $\zeta(N, \rho) \approx \frac{\rho^2}{2(1-\rho)}$ . We repeat the numeric experiment carried out in that paper, but we fill our table with the value of

$$N \cdot (W_N^{(SQ(d))} - W^{(SQ(d))}),$$

the results are shown in Table 8.1. We indeed observe that  $N \cdot (W_N^{(SQ(d))} - W^{(SQ(d))})$  converges to a constant and (as noted in [43]) this constant may be approximated by  $\rho^2/(2 \cdot (1 - \rho))$  (note that the actual limiting value is not  $\rho^2/(2 \cdot (1 - \rho))$  but may be computed numerically).

	$N = 10$	$N = 20$	$N = 30$	$N = 50$	$N = 100$	$\rho^2/(2 \cdot (1 - \rho))$
$\rho = 0.7$	0.8930	0.8680	0.8490	0.8500	0.8300	0.8167
$\rho = 0.9$	4.5130	4.2760	4.1400	4.0850	4.0400	4.0500
$\rho = 0.95$	10.8130	10.0420	9.6270	9.3150	9.0800	9.0250

Table 8.1: Table of the value  $N \cdot (W_N^{(SQ(d))} - W^{(SQ(d))})$  for different choices of  $\rho$  and  $N$ . In the last column, we add an almost  $O(1/N)$  approximation for  $N \cdot (W_N^{(SQ(d))} - W^{(SQ(d))})$ .

	$N = 10$	$N = 20$	$N = 30$	$N = 50$	$N = 100$	$\rho^2/(2 \cdot (1 - \rho))$
$\rho = 0.7$	1.0912	1.0068	0.9880	1.0243	0.8825	0.8167
$\rho = 0.9$	4.9676	4.5711	4.4835	4.4857	4.2606	4.0500
$\rho = 0.95$	11.8948	10.7204	10.2907	9.9708	9.4035	9.0250

Table 8.2: Table of the value  $N \cdot (W_N^{(LL(d))} - W^{(LL(d))})$  for different choices of  $\rho$  and  $N$ . In the last column, we add an almost  $O(1/N)$  approximation for  $N \cdot (W_N^{(LL(d))} - W^{(LL(d))})$ .

We repeat this experiment for LL( $d$ ). To approximate the value of  $W_N^{(LL(d))}$ , we ran 40 runs of length  $t = 10^8/N$  with a warm-up period of 30%. The results are shown in Table 8.2

We observe the same phenomenon: the value of  $N \cdot (W_N^{(LL(d))} - W^{(LL(d))})$  converges to some constant. Moreover (at least for this example) it appears that the sought constant does not differ that much from the constant for the SQ( $d$ ) policy. It might be worthwhile to investigate whether it's possible to generalize the results in [43] for the LL( $d$ ) policy.

**Remark 59.** *One can make a similar remark for most of the load balancing policies considered in this work. Generalizing the work of [43] could in fact be interesting for workload dependent, memory based and runtime dependent load balancing policies.*

## 8.2 Binary search to compute the load

When the system load (that is,  $\bar{F}(0)$ ) is hard to compute (as is for example the case for Red( $d$ ), see Section 3.6.4), we cannot directly solve the associ-

ated Differential Equations as we do not know the boundary condition  $\bar{F}(0)$ . Moreover, in this case, one does (generally) not know for which values of  $\lambda$  the model remains stable. Therefore one would like to (efficiently) compute some value  $\lambda_{\max}$  such that the system is stable for all  $\lambda < \lambda_{\max}$  but unstable for all  $\lambda > \lambda_{\max}$ .

Throughout this work we made use of a binary search in order to obtain the correct value for  $\bar{F}(0)$  such that  $\inf_{w>0} \bar{F}(w) = 0$ . Moreover, to obtain  $\lambda_{\max}$ , we set  $\bar{F}(0) \approx 1$  and used a binary search on  $\lambda$  to find the  $\lambda$  associated to this (very close to unstable) load, one can then set  $\lambda_{\max}$  equal to the value of  $\lambda$  obtained from the binary search.

In order to ensure that the suggested algorithms always work, we require that the following conjecture holds:

**Conjecture 2.** *Let  $\bar{F}_1(w)$  and  $\bar{F}_2(w)$  denote two solutions of (4.4) with boundary conditions  $\bar{F}_1(0) < \bar{F}_2(0)$ . If  $\inf_{w>0} \bar{F}_2(w) < 0$ , then also  $\inf_{w>0} \bar{F}_1(w) < 0$ .*

This makes sense, as one may interpret the FDE to represent the load which is being removed from the start-load  $\bar{F}(0)$ . When  $\bar{F}(0)$  is too large, we do not reach load 0 while picking a value of  $\bar{F}(0)$  which is too small, leads to a load equal to 0 at some finite value  $w$ .

### 8.3 Obtaining the stability region

For many queue length dependent load balancing policies with exponential job sizes with mean one and arrival rate  $\lambda$ , we can obtain the probability  $u_k$  that the cavity queue has  $k$  or more jobs in its queue using the recursive relation:

$$u_{k+1} = T_\lambda(u_k), \quad (8.1)$$

with  $T_\lambda$  defined appropriately. A necessary condition for stability is the convergence of the solution  $u_{k+1} = T_\lambda(u_k)$  to zero, i.e. we always have  $u_0 = 1$  as a boundary condition and using the recurrence relation (8.1), we should have  $u_k \downarrow 0$ . This is closely related to the existence and location of a fixed point  $u_\lambda$  which satisfies  $u_\lambda = T_\lambda(u_\lambda)$ ,  $u_\lambda \neq 0$ . We have the following conjecture:

**Conjecture 3.** *For a queue length dependent load balancing policy with exponential job sizes with mean one for which:*

$$u_{k+1} = T_\lambda(u_k),$$

the stability region is given by  $[0, \tilde{\lambda})$ , with  $\tilde{\lambda} = \max\{\lambda \in [0, \infty) \mid u_\lambda \in [0, 1]\}$  and  $u_\lambda$  the smallest positive root of  $T_\lambda(u) - u$ .

**Example 99.** For the  $SQ(d)$  policy with unit mean exponential job sizes we have  $T_\lambda(u) = \lambda \cdot u^d$ , therefore the unique root of  $T_\lambda(u) - u$  is given by  $u = (\frac{1}{\lambda})^{\frac{1}{d-1}}$ . From this it easily follows from Conjecture 3 that (as expected) the stability region of  $SQ(d)$  is  $[0, 1)$  for all values of  $d$ .

For many load balancing policies, a simple coupling argument suffices to prove that the stability region is given by the interval  $[0, 1)$ . However, if you would for example consider a load balancing policy which always assigns each incoming job to the second shortest queue amongst  $d$  randomly selected servers, this coupling argument no longer works. Is the stability region still  $[0, 1)$ ?

To this end, we introduce the  $SQ(d, p_1, \dots, p_d)$  policy which assigns each incoming job to the  $i$ 'th shortest queue with probability  $p_i$ . In what follows we use Conjecture 3 in order to analyze the  $SQ(d, p_1, \dots, p_d)$  policy.

### 8.3.1 General results

We introduce the policy  $SQ(d, p_1, \dots, p_d)$  where each incoming job joins the  $i$ 'th shortest queue amongst  $d$  randomly selected servers with probability  $p_i$ . For this policy, we first focus on the case of exponential job sizes with a mean equal to one. In order to analyse this policy, we denote by  $u_k(t)$  the probability that, at time  $t$ , an arbitrary queue has  $k$  or more jobs. Furthermore, we denote by  $q_{j,k}(u(t))$  the probability that the  $j$ 'th shortest queue amongst  $d$  randomly selected queues has at least  $k$  jobs. We find:

$$\begin{aligned} q_{j,k}(u(t)) &= \mathbb{P}\{j\text{'th shortest queue has length at least } k \text{ jobs}\} \\ &= \sum_{i=0}^{j-1} \binom{d}{i} (1 - u_k(t))^i \cdot u_k(t)^{d-i}. \end{aligned}$$

Using the equality:

$$\frac{d}{dt} u_k(t) = \lambda f_k(u(t)) - (u_k(t) - u_{k+1}(t)),$$

where  $f_k(u(t))$  is the up-crossing rate over the level  $k$ . It is not hard to see that we have:

$$f_k(u(t)) = \lambda \sum_{j=1}^d p_j [q_{j,k-1}(u(t)) - q_{j,k}(u(t))].$$

Taking the limit  $t \rightarrow \infty$  and performing some simple computations, we find that  $(u_k)_k$  satisfies the following recurrence relation:

$$u_{k+1} = T_\lambda(u_k) = \lambda \sum_{j=1}^d p_j \sum_{i=0}^{j-1} \binom{d}{i} (1 - u_k)^i u_k^{d-i}. \quad (8.2)$$

Let us first consider some examples of this policy.

**Example 100.** Taking  $p_j = \frac{1}{d}$  for each  $j$ , this policy reduces to random routing. Indeed, we find for (8.2):

$$\begin{aligned} T_\lambda(u) &= \frac{\lambda}{d} \sum_{j=1}^d \sum_{i=0}^{j-1} \binom{d}{i} (1 - u)^i u^{d-i} \\ &= \lambda \sum_{i=0}^{d-1} \frac{(d-i)}{d} \binom{d}{i} (1 - u)^i u^{d-i} \\ &= \lambda u \end{aligned}$$

Therefore we have  $u_{k+1} = \lambda u_k$ , which is also the case for random routing.

**Example 101.** Consider the case  $d = 2$ . We define  $p = p_2$  and thus find that  $p_1 = 1 - p_2 = 1 - p$ . It is not hard to see that:

$$\begin{aligned} T_\lambda(u) &= \lambda \cdot ((1-p)u^2 + p(u^2 + 2(1-u)u)) \\ &= \lambda \cdot ((1-2p)u^2 + 2pu). \end{aligned}$$

Therefore, if  $p \leq \frac{1}{2}$ , we obtain that this policy corresponds to the  $SQ(2p, (1-2p), 1, 2)$  policy, where with probability  $2p$  a job is routed arbitrarily and with probability  $1-2p$  it is routed to the server with the lowest load amongst 2 randomly chosen servers (see also Section 5.6).

### 8.3.2 Stability

We now present a few examples for which we can compute the stability region explicitly.

**Example 102.** For  $d = 2$  we find from Example 101 that the fixed point  $u_\lambda$  is given by (for  $p \neq \frac{1}{2}$ ):

$$u_\lambda = \frac{1 - 2p\lambda}{\lambda(1 - 2p)},$$

it's easy to see that we have  $u_1 = 1$ , therefore we should check if  $u_\lambda$  is increasing or decreasing in one in this case. We find that  $\frac{\partial u_\lambda}{\partial \lambda} = \frac{1}{\lambda^2 - 2p}$ . We distinguish three cases:

1.  $p < \frac{1}{2}$ , for this case we find that  $u_\lambda > 1$  for all  $\lambda < 1$ . We also know from Section 5.6 that  $\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} < \infty$ .
2.  $p = \frac{1}{2}$ , for this case we find that  $u_\lambda$  does not exist for  $\lambda \neq 1$ , we have stability for all  $\lambda < 1$  but  $\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} = \infty$ .
3.  $p > \frac{1}{2}$ , for this case we find that  $u_\lambda$  is increasing, therefore the stability region is smaller than  $[0, 1)$ , in particular due to the monotonicity of  $u_\lambda$ , we should compute the point at which  $u_\lambda$  crosses the  $x$ -axis. This happens at  $\lambda = \frac{1}{2p}$ , therefore we find the stability region  $[0, \frac{1}{2p})$ .

When we set  $p_d = 1$ , the situation also simplifies quite a bit:

**Example 103.** When  $d$  is arbitrary and  $p_d = 1$ , it is not hard to see that:

$$f(u) = T_\lambda(u) - u = (u - \lambda) + (1 - u)^d \lambda.$$

One finds that for  $\lambda = \frac{1}{d}$ , the fixed point  $u_\lambda$  which satisfies  $f(u_\lambda) = 0$  is equal to zero. Therefore, using the same ideas as in Example 102, one can use Conjecture 3 to show that the stability region for this policy is given by  $[0, \frac{1}{d})$ .

## 8.4 More results on critically loaded systems

### 8.4.1 BSQ( $d, K$ )

#### Main Result and Insights

We define the Batch Shortest Queue ( $d, K$ ) model as a model where jobs arrive in batches of size  $K$  the dispatcher randomly selects  $d$  servers and

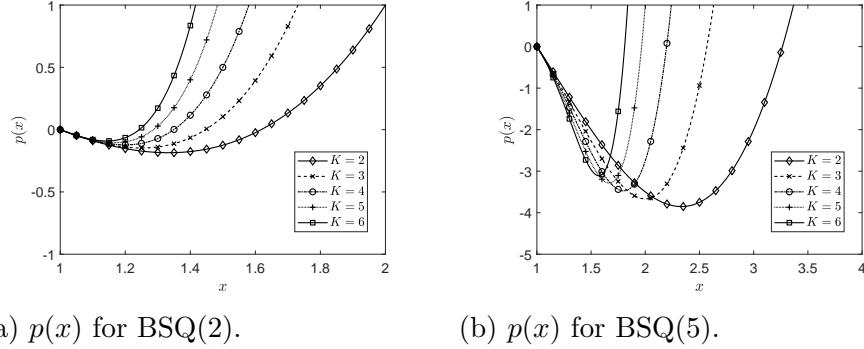


Figure 8.1: Plot of the function  $p(x) = x^{K+1} - (1 + \frac{d}{K})x^K + \frac{d}{K}$  associated to the  $\text{BSQ}(d, K)$ .

assigns all  $K$  jobs to the server with the shortest queue amongst the  $d$  selected servers. In this section, we show that for this policy we have:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1 - \lambda)} = \frac{1}{\log(\zeta)}, \quad (8.3)$$

where  $\zeta \in (1, \infty)$  is the unique root of the polynomial  $p(x) = x^{K+1} - (1 + \frac{d}{K})x^K + \frac{d}{K}$  larger than 1.

**Remark 60.** In particular if  $K = 1$  we recover the classic result for  $SQ(d)$  as we have  $\zeta = d$  in this case. In general, this root is bounded from below by  $\frac{K+d}{K+1}$ .

In Figure 8.1 we plot  $p(x)$  for different values of  $d$  and  $K$ , we observe that as  $K$  increases, the value of  $\zeta$  decreases while  $\zeta$  increases as a function of  $d$ . Due to (8.3) this corresponds to having a higher limiting value as the amount of probes used per arrival increases.

In Figure 8.2, we plot the quotient  $\lim_{\lambda \rightarrow 1^-} \frac{\mathbb{E}[W_\lambda^{SQ(d, K)}]}{\mathbb{E}[W_\lambda^{BSQ(d, K)}]}$ , we observe that the quotient is always greater than one, indicating that in a critically loaded system it is better to assign all  $K$  incoming jobs to the server with the shortest queue rather than spreading out the load amongst the  $K$  least loaded servers. In particular in Figure 8.2a, we observe that if we keep  $d/K$  fixed and take the limit  $K \rightarrow \infty$ , the quotient increases to some finite value which decreases as a function of  $d/K$ . In Figure 8.2b we confirm that the quotient decreases monotonically to one as  $d/K$  increases to infinity.

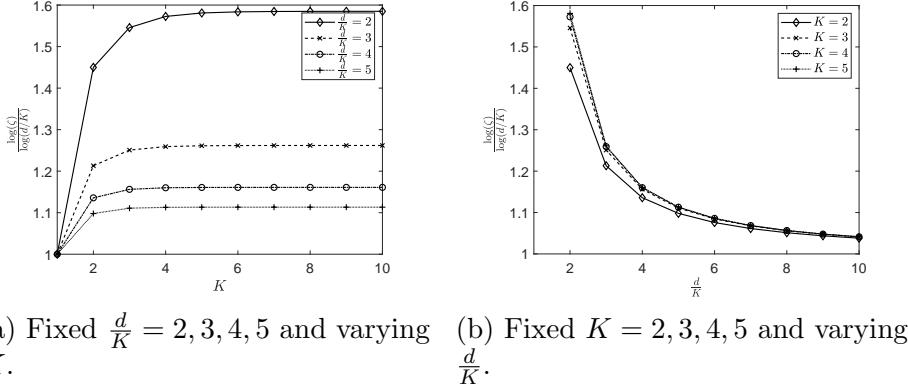


Figure 8.2: Plot comparing the waiting time for  $SQ(d, K)$  and  $BSQ(d, K)$  for  $\lambda \approx 1$ . We have  $SQ(d, K)$  in the nominator, and observe that it is always outperformed by  $BSQ(d, K)$ .

### Proof of (8.3)

*Proof.* It is not hard to show that for  $k < K$

$$u_k - u_{k+1} = \frac{\lambda}{K}(1 - u_k^d),$$

while for  $k \geq K$  we have:

$$u_k - u_{k+1} = \frac{\lambda}{K}(u_{k-K}^d - u_k^d) \quad (8.4)$$

From (8.4), it is not hard to see that  $(u_k)_k$  for  $k \geq K$  satisfies:

$$u_{k+1} = \frac{\lambda}{K} \sum_{j=0}^{K-1} u_{k-j}^d. \quad (8.5)$$

Further, let for any  $\lambda < 1$ :  $u_\lambda = \lambda^{\frac{1}{1-d}}$ , therefore we have:

$$u_\lambda = \frac{\lambda}{K} \sum_{j=0}^{K-1} u_\lambda^d.$$

Define  $\tilde{u}_k = u_\lambda - u_k$ , we find that:

$$\begin{aligned}\tilde{u}_{k+1} &= u_\lambda - u_{k+1} \\ &= u_\lambda - \frac{\lambda}{K} \sum_{j=0}^{K-1} u_{k-j}^d \\ &= \frac{\lambda}{K} \sum_{j=0}^{K-1} u_\lambda^d - (u_\lambda - \tilde{u}_{k-j})^d.\end{aligned}$$

Let us now define  $\xi_\lambda(x) = \lambda \left( \frac{u_\lambda^d - (u_\lambda - x)^d}{x} \right)$  and  $B_{k+1} = \frac{\tilde{u}_k}{\tilde{u}_{k+1}}$ . We find that:

$$\frac{1}{B_{k+1}} = \frac{1}{K} \left[ \xi_\lambda(\tilde{u}_k) + \sum_{j=1}^{K-1} \xi_\lambda(\tilde{u}_{k-j}) B_k \cdot \dots \cdot B_{k-j+1} \right],$$

or equivalently :

$$B_{k+1} = \frac{K}{\xi_\lambda(\tilde{u}_k) + \sum_{j=1}^{K-1} \xi_\lambda(\tilde{u}_{k-j}) B_k \cdot \dots \cdot B_{k-j+1}}. \quad (8.6)$$

From Chapter 5 we already know that  $\frac{\partial}{\partial x} \xi_\lambda(x) < 0$  for all  $x \in (0, u_\lambda]$ . Let us denote

$$N_{\varepsilon, \lambda} = \max\{k \in \mathbb{N} \mid u_k \leq \varepsilon\}.$$

We find for all  $k \leq N_{\varepsilon, \lambda}$  that:

$$\ell_\varepsilon = \frac{1 - (1 - \varepsilon)^d}{\varepsilon} \downarrow \xi_\lambda(\varepsilon) \leq \xi_\lambda(\tilde{u}_k) \leq \xi_\lambda(u_\lambda - 1) \uparrow d.$$

We therefore find that :

$$\ell_\varepsilon \leq \xi_\lambda(\tilde{u}_k) \leq d.$$

Let us denote  $a_k = \xi_\lambda(\tilde{u}_k) \in (\ell_\varepsilon, d)$ . From (8.6) we find that (for  $k \geq K$ ):

$$\begin{aligned}B_{k+1} &= K \frac{1}{a_k + \sum_{j=1}^{K-1} a_{k-j} B_k \cdot \dots \cdot B_{k-j+1}} \\ &= K \frac{1}{a_k + B_k \left[ \sum_{j=2}^{K-1} a_{k-j} B_{k-1} \cdot \dots \cdot B_{k-j+1} + a_{k-1} \right]} \\ &= K \frac{1}{a_k + B_k \left[ \frac{K}{B_k} - \frac{1}{K} B_{k-K} B_{k-1} \cdot \dots \cdot B_{k-K+1} \right]} \\ &= \frac{1}{\frac{1}{K} a_k + 1 - \frac{1}{K} B_{k-K} \prod_{j=0}^{K-1} B_{k-j}}.\end{aligned}$$

Let us denote  $q = \frac{1}{K}$ , we obtain :

$$B_{k+1} = \frac{1}{(1 + qa_k) - qa_{k-K} \prod_{j=0}^{K-1} B_{k-j}}.$$

Let us define:

$$A_K = (1 + qa_K) - qa_0 \prod_{j=0}^{K-1} B_{K-j}$$

we find that  $B_{K+1} = \frac{1}{A_K}$ , further we have:

$$\begin{aligned} B_{K+2} &= \frac{1}{(1 + qa_{K+1}) - qa_1 \prod_{j=0}^{K-1} B_{K+1-j}} \cdot \frac{A_K}{A_K} \\ &= \frac{A_K}{(1 + qa_{K+1})A_K - qa_1 \prod_{j=0}^{K-1} B_{K+1-j}} \\ &= \frac{A_K}{A_{K+1}} \end{aligned}$$

where we define:

$$A_{K+1} = (1 + qa_{K+1})A_K - qa_1 \prod_{j=1}^{K-1} u_{K+1-j}.$$

Similarly one may compute with  $B_{K+3} = \frac{A_{K+1}}{A_{K+2}}$ :

$$A_{K+2} = (1 + qa_{K+2})A_{K+1} - qa_2 \prod_{j=2}^{K-1} u_{K+2-j}.$$

In general we find that  $B_{n+2} = \frac{A_n}{A_{n+1}}$  for  $n = K + 2, \dots, 2K - 1$  by letting :

$$\begin{aligned} A_K &= (1 + qa_K) - qa_0 \prod_{j=0}^{K-1} B_{K-j}, \\ A_{K+1} &= (1 + qa_{K+1})A_K - qa_1 \prod_{j=1}^{K-1} B_{K+1-j}, \\ A_{K+2} &= (1 + qa_{K+2})A_{K+1} - qa_2 \prod_{j=2}^{K-1} B_{K+2-j}, \\ &\vdots \\ A_{2K-1} &= (1 + qa_{2K-1})A_{2K-2} - qa_{K-1}B_K, \\ A_{2K} &= (1 + qa_{2K})B_{2K-1} - qa_K. \end{aligned}$$

After this point (i.e. for  $n \geq 2K + 2$ ) we have  $u_n = \frac{v_{n-2}}{v_{n-1}}$  by using the simple linear recursion :

$$A_n = (1 + qa_n)A_{n-1} - qa_{n-K-1}A_{n-K-1}, n \geq 2K + 1 \quad (8.7)$$

We also define two related recursions which serve as an upper resp. lower bound:

$$\bar{A}_n = (1 + qd) \cdot \bar{A}_{n-1} - q\ell_\varepsilon \bar{A}_{n-K-1} \quad (8.8)$$

$$\underline{A}_n = (1 + q\ell_\varepsilon) \cdot \underline{A}_{n-1} - qd \bar{A}_{n-K-1}, \quad (8.9)$$

and for all  $n < 2K + 1$  we set  $\underline{A}_n = A_n = \bar{A}_n$ . It follows that (for sufficiently small  $\varepsilon$ ) for all  $n$  we have :

$$\underline{A}_n \leq A_n \leq \bar{A}_n.$$

Indeed, when  $\underline{A}_k = A_k = \bar{A}_k$ , this is obvious. When  $\underline{A}_n < A_k < \bar{A}_k$  this inequality is first strengthened by the coefficient of  $\bar{A}_{n-1}$  resp.  $\underline{A}_{n-1}$ . Afterwards this is again weakened by the coefficient of  $\bar{A}_{n-K-1}$  resp.  $\underline{A}_{n-K-1}$ . However, for sufficiently small  $\varepsilon$ , the increase caused by  $(1 + qd)$  resp. decrease by  $(1 + q\ell_\varepsilon)$  is greater than the reverse movement by  $-q\ell_\varepsilon$  resp.  $-qd$ . We should therefore study  $\underline{A}_n$  and  $\bar{A}_n$ , as the analysis depends on the roots of the characteristic polynomial, and the roots of a polynomial are continuous

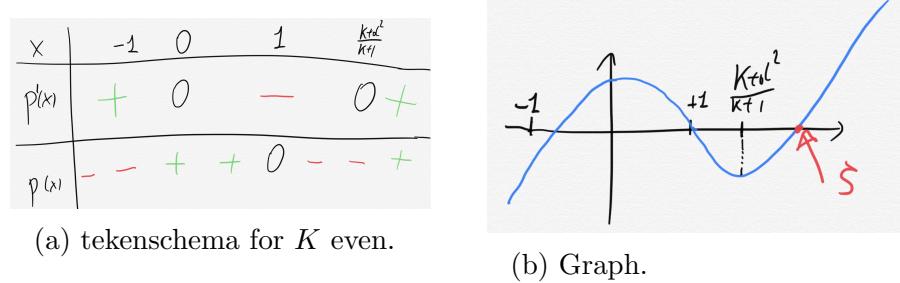


Figure 8.3:  $p(x)$  for  $K$  even. Heb een  $d^2$  ipv  $d$  geschreven, komt doordat ik dit fout had eerst maar als we deze afbeeldingen behouden moet ik ze toch sowieso nog mooi maken :).

functions of its coefficients, it suffices (up to some multiple of  $\varepsilon$ ) to study the behaviour of the recursive relation:

$$x_n = (1 + qd) \cdot \bar{A}_{n-1} - qd\bar{A}_{n-K-1}.$$

its characteristic polynomial is given by:

$$p(x) = x^{K+1} - (1 + qd)x^K + qd.$$

It is not hard to see that this polynomial has exactly one root (which we refer to as  $\zeta$ ) which is bounded away from  $[-1, 1]$ , in particular we have  $\zeta > \frac{K+d}{K+1}$ . We therefore find that for  $n$  sufficiently large,  $A_n \approx c \cdot \zeta^n$  for some constant  $c$ , therefore we have  $B_n \approx \zeta^{-1}$ . This shows that if we take  $\bar{\lambda}$  sufficiently close to one and  $\bar{\varepsilon}$  sufficiently small, we can find a continuous function  $f$  with  $f(0) = 0$ , a value  $\bar{k} < N_{\varepsilon, \lambda}$  such that for all  $\lambda > \bar{\lambda}$ ,  $0 < \varepsilon < \bar{\varepsilon}$  and  $N_{\varepsilon, \lambda} \geq k \geq \bar{k}$  we have:

$$\zeta^{-1} - f(\varepsilon) < B_k < \zeta^{-1} + f(\varepsilon).$$

The remainder of the proof goes along the same lines as the proof of Theorem 59.  $\square$

### 8.4.2 The polynomial $p(x) = x^{K+1} - (1 + \frac{d}{K}x^K + \frac{d}{K})$

We can define the strongly related polynomial :

$$q(x) = x^{K+1} - \left(1 + \frac{K}{d}\right)x + \frac{K}{d} = 0,$$

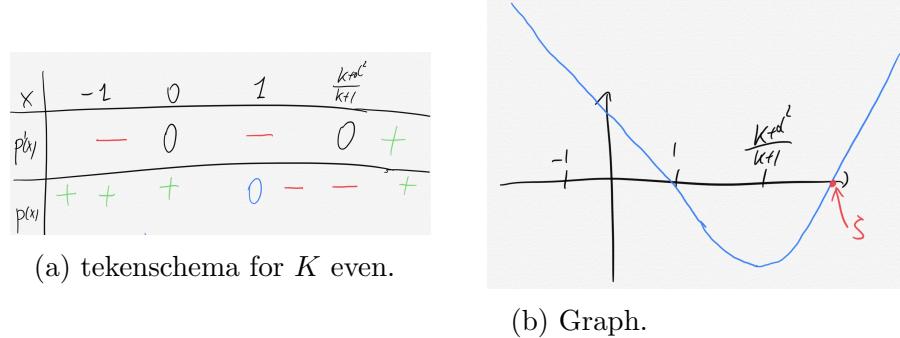


Figure 8.4:  $p(x)$  for  $K$  odd. Heb een  $d^2$  ipv  $d$  geschreven, komt doordat ik dit fout had eerst..

it is not hard to see that  $q(y) = 0$  if and only if  $p\left(\frac{1}{y}\right) = 0$ . Therefore  $\zeta^{-1}$  is the unique root in  $(0, 1)$  of  $q(x)$ . Note that we can write:

$$q(x) = (x - 1) \cdot (x^K + \dots + x - \frac{K}{d}),$$

thus  $\zeta^{-1}$  is the unique root of  $\frac{x^K + \dots + x}{K} - \frac{1}{d} = 0$ . We can consider the case where the amount of redundancy stays constant, i.e.  $d/K = c$  for some constant  $c$ . We find in the limit  $d, K \rightarrow \infty$ :

$$\begin{aligned} 0 &= \lim_{K \rightarrow \infty} (x^K + \dots + x - \frac{1}{c}) \\ &= \frac{x}{1-x} - \frac{1}{c}, \end{aligned}$$

we therefore find that  $\zeta^{-1} = \frac{1}{1+c}$ . We thus find that

$$\lim_{K \rightarrow \infty} \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda^{(BSQ(d,K))}]}{\log(1-\lambda)} = \frac{1}{\log(1+c)},$$

where  $c \in \mathbb{R}$  is the number of probes used per arrival.

### 8.4.3 Erlang job sizes

Now consider the SQ( $d$ ) policy with Erlang( $K, 1/K$ ) distributed job sizes. This system resembles the BSQ( $d, K$ ) policy with Exp( $1/K$ ) distributed job

sizes. If we denote by  $Q_{Er_K}$  resp.  $Q_{BSQ(d,K)}$  the associated queue length distributions, we find through a coupling argument that:

$$\frac{Q_{BSQ(d,K)}}{K} \leq Q_{Er_K} \leq \frac{Q_{BSQ(d,K)}}{K} + 1.$$

This entails by Little's Law that if we denote  $W_\lambda^{Er(K,1/K)}$  the response time distribution for the SQ( $d$ ) policy with  $Er(K, 1/K)$  job sizes that:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda^{Er(K,1/K)}]}{\log(1-\lambda)} = \frac{1}{K \log(\zeta)}.$$

#### 8.4.4 Deterministic job sizes

A deterministic job can be approximated by  $Er(K, 1/K)$  job with  $K$  sufficiently large. Therefore we find that for deterministic job sizes:

$$\lim_{\lambda \rightarrow 1^-} \frac{\mathbb{E}[W_\lambda^{DET}]}{\log(1-\lambda)} = \lim_{K \rightarrow \infty} \frac{1}{K \log(\zeta_K)},$$

where  $\zeta_K^{-1}$  is the unique root of the equation:

$$\frac{x^{K+1} - x}{x - 1} \cdot \frac{1}{K} = \frac{1}{d}.$$

Now let  $C \in \mathbb{R}$  be such that  $x = e^{C/(K+1)}$  we therefore find that  $C$  satisfies:

$$\frac{e^C - e^{C/(K+1)}}{e^{C/(K+1)} - 1} \frac{1}{K} = \frac{1}{d}.$$

Using l'Hopital's rule to simplify the denominator, we obtain in the limit of  $K \rightarrow \infty$  that:

$$e^C - \frac{C}{d} - 1 = 0.$$

The solution of this equation is given by:

$$C = -W(-de^{-d}) - d, \quad (8.10)$$

where  $W(z)$  is the product log function, i.e. it is the primary root of the equation  $we^w = z$ . We therefore find that the limit for  $\lambda$  to one is given by:

$$\lim_{\lambda \rightarrow 1^-} \frac{\mathbb{E}[W_\lambda^{DET}]}{\log(1-\lambda)} = \frac{1}{W(-de^{-d}) + d}.$$

### 8.4.5 The LL( $d$ ) policy with Erlang/Deterministic job sizes

We have not been able to obtain limiting results for the mean waiting time for the LL( $d$ ) policy with Erlang or deterministic job sizes. However, looking at the results of Chapter 5, we expect the limit for the LL( $d$ ) policy with Erlang( $K, 1/K$ ) job sizes to be given by:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda^{(LL(d), Er(K, 1/K))}]}{\log(1 - \lambda)} = \frac{1}{K(\zeta_K - 1)},$$

where  $\zeta_K^- 1$  is the unique root of the equation:

$$\frac{x^{K+1} - x}{x - 1} \cdot \frac{1}{K} = \frac{1}{d}.$$

We have found that in the limit  $K \rightarrow \infty$ , the value of  $\zeta_K$  may be written as  $\zeta_K = e^{C/(K+1)}$ , with  $C$  as defined in (8.10). From this, we are able to show that the limit for  $\lambda \rightarrow 1^-$  of the mean waiting time for SQ( $d$ ) and LL( $d$ ) is the same for deterministic job sizes:

$$\begin{aligned} \lim_{\lambda \rightarrow 1^-} \frac{\mathbb{E}[W_\lambda^{(DET, SQ(d))}]}{\mathbb{E}[W_\lambda^{(DET, LL(d))}]} &= \lim_{K \rightarrow \infty} \frac{K \cdot (e^{C/(K+1)} - 1)}{K \cdot \frac{C}{K+1}} \\ &= 1 \end{aligned}$$

### 8.4.6 Convex combinations

We believe that if one were to consider a policy  $P$ , where policy  $P$  executes some policy  $P_1$  with probability  $p_1$  and  $P_2$  with probability  $p_2$ , the high load system can be written as:

$$\lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_\lambda]}{\log(1 - \lambda)} = p_1 \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_{1,\lambda}]}{\log(1 - \lambda)} + p_2 \lim_{\lambda \rightarrow 1^-} -\frac{\mathbb{E}[W_{2,\lambda}]}{\log(1 - \lambda)}, \quad (8.11)$$

with  $W_\lambda, W_{1,\lambda}$  resp.  $W_{2,\lambda}$  the waiting time distribution for a model which executes policy  $P$ ,  $P_1$  resp.  $P_2$ . We have found this to be true for some choices of  $P_1$  and  $P_2$ , but have not found a proof of this general result. In order to show this result we assume that  $P_1$  and  $P_2$  satisfy all conditions stated in Theorem 59 or 60. We then tried to show that  $P$  also satisfies all

assumptions of the same Theorem. However, we found that assumption 3 is especially hard to establish for the convex combination of policies and have not yet managed to solve it.

One may also try to find alternative criteria for Theorems 59 and 60, or come up with a more direct method which shows that (8.11) indeed holds.

## 8.5 Bounds on the mean waiting time

We considered both the limit for  $\lambda \rightarrow 1^-$  and  $\lambda \rightarrow 0^+$  for

$$-\mathbb{E}[W_\lambda]/\log(1-\lambda) \text{ and } -\mathbb{E}[W_\lambda]/\log(1-p_\lambda),$$

with  $p_\lambda$  defined as in Section 5.7. We observed in Figures 5.3 and 5.4 both quotients are monotonically increasing, therefore the limits can be used as upper and lower bounds rather than approximations. For the LL( $d$ ) policy we can prove these results, however, for other load balancing policies we have not been able to prove this result. This might also be of interest for example for the quotient of the waiting time of the LL( $d$ ) and SQ( $d$ ) policies (as can be seen in Figure 2.3). We now present the proof for the LL( $d$ ) policy:

**Theorem 104.** *Let  $d \geq 2$  and  $\lambda \in (0, 1)$  be arbitrary and assume job sizes are exponential with mean one. Let  $W_\lambda$  denote the mean waiting time for the LL( $d$ ) policy, we find that:*

$$\frac{\partial}{\partial \lambda} - \frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda)} \geq 0 \quad (8.12)$$

$$\frac{\partial}{\partial \lambda} - \frac{\mathbb{E}[W_\lambda]}{\log(1-\lambda^d)} \geq 0 \quad (8.13)$$

*Proof.* We first show that (8.12) holds. To this end we write  $-\log(1-\lambda)$  as  $\sum_{n=1}^{\infty} \frac{\lambda^n}{n}$ . Therefore we find that it suffices to show that:

$$\sum_{n,m=1}^{\infty} \left( \frac{dn+1}{1+n(d-1)} \right) \lambda^{dn} \frac{\lambda^m}{m} \geq \sum_{n,m=1}^{\infty} \frac{1}{1+n(d-1)} \lambda^{dn+m}.$$

Let  $j \in \mathbb{N}_0$  and  $1 \leq k \leq d$  be arbitrary, we now compare the coefficients of  $\lambda^{jd+k}$ , we find that it suffices to show that:

$$\sum_{\ell=1}^j \frac{d\ell+1}{1+\ell(d-1)} \frac{1}{(j-\ell)d+k} \geq \sum_{\ell=1}^j \frac{1}{1+\ell(d-1)}.$$

It is not hard to see that we can assume that  $k = 1$  and some simple algebra shows that this inequality is equivalent to:

$$\sum_{\ell=1}^j \frac{2\ell - j}{(1 + \ell(d-1)) \cdot ((j-\ell)d+1)} \geq 0.$$

We now distinguish the cases where  $j$  is even or odd. Let us first assume that  $j = 2 \cdot i$  is even. We find that:

$$\begin{aligned} & \sum_{\ell=1}^{2i} \frac{2\ell - 2i}{(1 + \ell(d-1)) \cdot ((2i-\ell)d+1)} \\ &= \sum_{\ell=1}^{i-1} \frac{2\ell - 2i}{(1 + \ell(d-1)) \cdot ((2i-\ell)d+1)} + \sum_{\ell=i+1}^{2i} \frac{2\ell - 2i}{(1 + \ell(d-1)) \cdot ((2i-\ell)d+1)} \\ &= \sum_{\ell=1}^{i-1} \left[ \frac{2\ell - 2i}{(1 + \ell(d-1)) \cdot ((2i-\ell)d+1)} + \frac{2i - 2\ell}{(1 + (2i-\ell)(d-1)) \cdot (\ell d+1)} \right] \\ &= \sum_{\ell=1}^{i-1} \frac{(2i-2\ell)^2}{(1 + (2i-\ell)(d-1))(\ell d+1) \cdot (1 + \ell(d-1))((2i-\ell)d+1)} \geq 0 \end{aligned}$$

We now consider the case where  $j$  is odd, we write  $j = 2i + 1$  and note that we should show that:

$$\sum_{\ell=1}^{2i+1} \frac{2\ell - (2i+1)}{(1 + \ell(d-1)) \cdot (((2i+1)-\ell)d+1)} \geq 0.$$

To this end we have:

$$\begin{aligned} & \sum_{\ell=1}^i \frac{2\ell - (2i+1)}{(1 + \ell(d-1)) \cdot (((2i+1)-\ell)d+1)} \\ &+ \sum_{\ell=i+1}^{2i} \frac{2\ell - (2i+1)}{(1 + \ell(d-1)) \cdot (((2i+1)-\ell)d+1)} \\ &= \sum_{\ell=1}^i \frac{(2i+1-2\ell)^2}{(1 + (2i+1-\ell)(d-1))(\ell d+1)(1 + \ell(d-1))((2i+1-\ell)d+1)} \geq 0. \end{aligned}$$

To show that (8.13) holds. To this end we write  $-\log(1-\lambda^d)$  as  $\sum_{n=1}^{\infty} \frac{\lambda^{dn}}{n}$ . Therefore, it suffices to show that:

$$\left( \sum_{n=1}^{\infty} \frac{\lambda^{dn+1}}{1+n(d-1)} \right)' \cdot \sum_{n=1}^{\infty} \frac{\lambda^{dn}}{n} \geq \sum_{n=1}^{\infty} \frac{\lambda^{dn+1}}{1+n(d-1)} \cdot \left( \frac{\lambda^{dn}}{n} \right)'.$$

This easily reduces to:

$$\sum_{n,m=1}^{\infty} \frac{dn+1}{1+n(d-1)} \cdot \frac{1}{m} \lambda^{d(n+m)} \geq \sum_{n,m} \frac{d}{1+n(d-1)} \cdot \lambda^{d(n+m)}.$$

We compare the coefficients of  $\lambda^{dk}$  for  $k \geq 2$ , we find that it suffices to show that:

$$\sum_{n=1}^{k-1} \left( \frac{dn+1}{1+n(d-1)} \frac{1}{k-n} \right) \geq \sum_{n=1}^{k-1} \frac{d}{1+n(d-1)}$$

The remainder of this proof goes along the same lines as the proof of (8.12).  $\square$

VANAF HIER DENK IK DAT HET ALLEMAAL DINGEN ZIJN DIE NOG TE WEINIG UITGEWERKT EN/OF NIET INTERESSANT GENOEG ZIJN OM EFFECTIEF TOE TE VOEGEN!

## 8.6 Alternative scheduling policies

We know that for the LL( $d$ ) the workload distribution we obtained in Chapter 2 holds true for alternative scheduling policies. However, computing the response time distribution for other scheduling policies proves to be difficult. Een aantal dingen die misschien interessant zijn om te bekijken :

1. Voor non-idling & non-preemption aantonen dat de mean response tijd hetzelfde blijft, in deze sectie kan dat evt via simulatie.
2. Snelle simulatie toelichten, maar werkt enkel voor non-preemption
3. Wat als er wel pre-emption is?

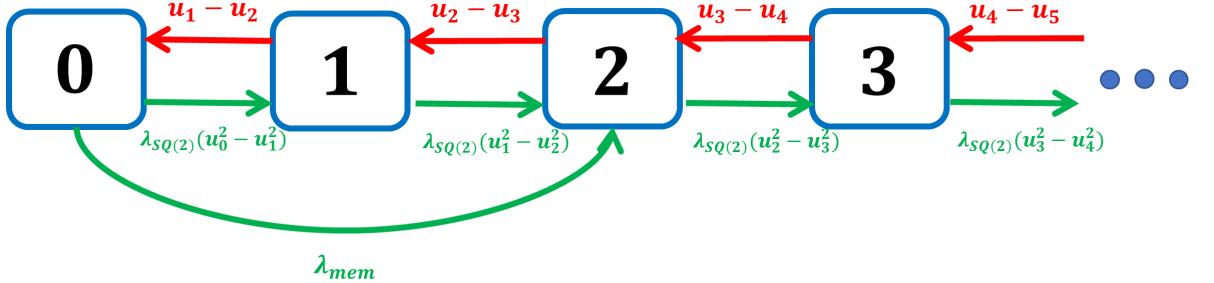


Figure 8.5: The transition rates for the  $\text{MSQ}(2, 2)$  policy.

## 8.7 Memory dependent load balancing continued

In Chapter 6, we assumed that we discover empty queues and we simply assign one job to these servers. In this section, we generalize the results from that chapter to the case where we assign to each empty queue which is discovered not one, but  $n$  jobs. The other jobs are routed using either the  $\text{SQ}(d)$  or  $\text{LL}(d)$  policy. We first focus on the  $\text{SQ}(d)$  policy.

Let us denote the Memory dependent  $\text{SQ}(d)$  policy where we assign  $n$  jobs to servers in memory by  $\text{MSQ}(d, n)$ , likewise we use the notation  $\text{MLL}(d, n)$  for the Memory dependent  $\text{LL}(d)$  policy where we assign  $n$  jobs to servers in memory.

### 8.7.1 $\text{SQ}(d)$

As we saw in Chapter 6, the analysis in case  $n = 1$  boils down to finding the correct value of  $\pi_0$  which denotes the probability that the memory is empty. Let us assume  $\pi_0$  is known, we find that we can divide the arrival stream into two separate streams,  $\pi_{\text{mem}} = (1 - \pi_0) \cdot \lambda$  and  $\pi_{\text{SQ}(d)} = \pi_0 \cdot \lambda$ , that is arrivals which join a queue in memory and arrivals which join a queue using the  $\text{SQ}(d)$  policy. While for the value  $n = 1$ , we can have  $\pi_0 \in [0, 1]$  to take an arbitrary value, we require, for  $n \geq 2$ , that  $\pi_0 \in [0, \frac{1}{n}]$ . Indeed, every time a server is in memory we do not assign 1, but  $n$  jobs to it, therefore  $n\pi_0$  can be no larger than 1. We graphically represent the state transitions for the  $\text{MSQ}(2, 2)$  policy in Figure 8.5. Moreover, we find that the arrival rate to

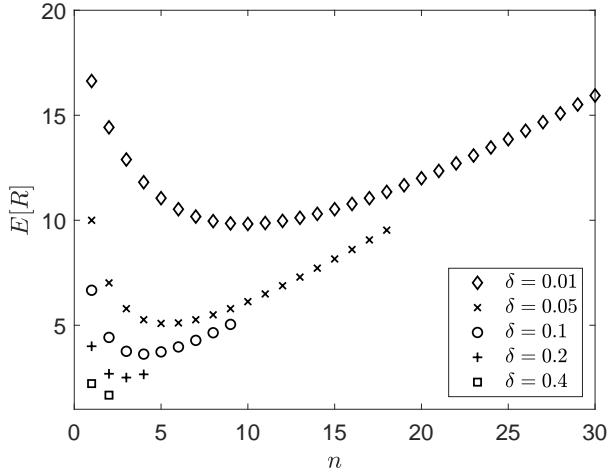


Figure 8.6: Performance of Join Up To  $n$  with update rate  $\delta$  and arrival rate  $\lambda = 0.95$ .

queues in memory is now given by  $\lambda_{\text{mem}} = \lambda\pi_0 n$ , while for the remaining arrivals we have an arrival rate equal to  $\lambda_{SQ(d)} = \lambda(1 - \pi_0 n)$ .

Let us denote by  $u_k$  the probability that a queue has a length which is at least equal to  $k$ . We can again use a level crossing argument to show that for  $k \leq n$  we have:

$$u_k - u_{k+1} = \lambda_{SQ(d)}(u_{k-1}^d - u_k^d) + \lambda_{\text{mem}}(u_0 - u_1),$$

while for  $k > n$  we have:

$$u_k - u_{k+1} = \lambda_{SQ(d)}(u_{k-1}^d - u_k^d).$$

Taking the sum  $\sum_{\ell \geq k}$  on both sides and using the fact that  $u_0 = 1$ ,  $u_1 = \lambda$  we find:

$$u_{k+1} = \lambda_{SQ(d)} u_k^d + \max\{n - k, 0\} \lambda_{\text{mem}} (1 - \lambda).$$

Opmerkingen :

- Closed form solution for the case  $d = 1$ .
- Critically loaded system, in particular for finite memory, we can decrease the high load limit by  $\frac{1}{(M+1)\cdot n}$ . Toont ook wel aan de zwakte van

het hoge load resultaat, kunnen dat willekeurig klein krijgen voor een policy die echt heel slecht werkt! Maar de low load limiet voor deze policy is uiteraard ook niet om over naar huis te schrijven..

- Deze policy komt wel zeer dicht bij het vaarwater van Hyper scalable paper, dus kan misschien beter daar behandeld worden. Voor het hyper scalable verhaal hebben we gewoon rate  $\delta$  waarmee de idle server probes sturen, zie figuur 8.6 voor de performance ifv de update rate  $\delta$ .

## 8.8 Link between queue length and workload dependent load balancing

Iets wat me nu opvalt met gedane werk te bekijken : ik vind dat we het verband tussen de analyse voor workload dependent en queue length dependent policies nog maar te weinig geexploereerd hebben. Ihb : voor policies waar jobs maar per 1 aan servers kunnen gegeven worden heb ik dus gevonden (niet bewezen maar gewoon gevonden voor goed aantal voorbeelden) dat (voor  $\text{Exp}(1)$  job sizes): als queue length dependent gegeven wordt door  $u_{k+1} = T(u_k)$  dan wordt de workload dependent policy gegeven door de opl  $\bar{F}'(w) = T(\bar{F}(w)) - \bar{F}(w)$  Deze analogie zou idealiter iha bewezen moeten worden (of gewoon inzien waarom dit is), dan zien hoe daaruit volgt dat de response tijden voor workload dependent lager zijn dan voor queue length dependent en zien hoe (en of) dit soort dingen veralgemenen naar andere job size verdelen en policies waarbij meerdere jobs per keer gegeven kunnen worden aan een server.

# **Appendices**

# Appendix A

## The M/M/1 queue

There are many good introductory texts on queueing theory (e.g. [45]), however we do not require much queueing theory, therefore we added this section to give the reader just enough information to be able to work through this thesis.

The most basic example of a queue is the M/M/1 queue, this is a single queue to which exponentially distributed jobs arrive which leave the system in a FCFS order. We assume the arrival rate is equal to  $\lambda$  and jobs have a mean equal to one. A graphical representation of this queue is given in Figure A.1. As job sizes are exponential, we find that at each time slot, jobs leave the system at a rate equal to one, irrespective of the time the job has been in the system (by virtue of the memoryless property of the exponential distribution). Therefore at any point in time the queue length grows by one with a rate equal to one and it decreases with a rate equal to  $\lambda$ . Therefore, one only needs to know the queue length to describe the M/M/1 queue.

In this section we elaborate on a few different methods which one may use



Figure A.1: Graphical illustration of an M/M/1 queue.

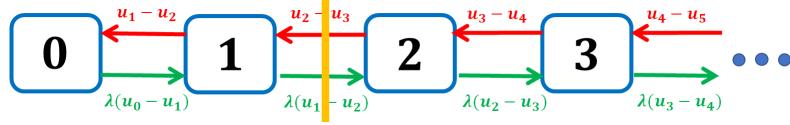


Figure A.2: Diagram illustrating the state transitions for the M/M/1 queue. Squares represent the queue length and arrows the transition rates. We note that the up-crossing rate must equal the down-crossing rate (e.g. at the orange line) this means that  $u_2 - u_3 = \lambda(u_1 - u_2)$ .

to analyse the M/M/1 queue. Each of these methods are different and have a different application in the thesis, a good understanding of these methods will be useful in understanding the idea behind the analytical methods proposed in the main text.

## A.1 Level crossing - queue length

We introduce the notation  $u_k$  as the probability that the queue length of the M/M/1 queue at an arbitrary point in time is at least  $k$ . In Figure A.2, we represent each possible queue length of the M/M/1 queue by a square. The green arrows represent the rate at which the queue length jumps up, for each queue length  $k$ , we have a jump from queue length  $k$  to  $k+1$  if the queue has a length equal to  $k$  (which happens with probability  $u_k$ ) and an arrival occurs (which happens at a rate equal to  $\lambda$ ), therefore the rate at which the queue length jumps up from  $k$  to  $k+1$  is equal to  $\lambda(u_k - u_{k+1})$ . Furthermore, the queue processes jobs at a rate equal to one, therefore we find that a jump from queue length  $k+1$  to length  $k$  occurs at a rate equal to  $(u_k - u_{k+1}) \cdot 1$ . We now apply a level crossing argument to obtain the value of  $(u_k)_k$ , that is, we can draw a line anywhere in the diagram represented in Figure A.2, and the rate at which we cross that line in both ways (left to right and right to left) should be equal. This may be interpreted as : any time the queue length increases from  $k$  to  $k+1$ , it should also decrease from  $k+1$  to  $k$  some time in the future. For the M/M/1 queue, this translates to the equation:

$$u_k - u_{k+1} = \lambda(u_{k-1} - u_k),$$

summing this equality from  $k$  to infinity on both sides, we obtain:

$$u_k = \sum_{\ell=k}^{\infty} (u_{\ell} - u_{\ell+1}) = \lambda \sum_{\ell=k}^{\infty} (u_{\ell-1} - u_{\ell}) = \lambda u_{k-1}.$$

This allows us to conclude that  $u_k = \lambda u_{k-1}$  for all  $k \geq 1$ . As we trivially have  $u_0 = 1$ , we find that  $u_k = \lambda^k$  for all  $k \geq 0$ . From this one can easily compute performance measures such as the mean queue length:

$$\mathbb{E}[Q] = \sum_{k=1}^{\infty} k \cdot (u_k - u_{k+1}) = \sum_{k=1}^{\infty} u_k = \sum_{k=1}^{\infty} \lambda^k = \frac{\lambda}{1-\lambda}.$$

Denoting the mean response time by  $\mathbb{E}[R]$ , Little's Law tells us that for any queueing system considered in this work we have  $\mathbb{E}[R] = \mathbb{E}[Q]/\lambda$ , in particular for the M/M/1 queue we obtain  $\mathbb{E}[R] = \frac{1}{1-\lambda}$ . While this might be the easiest approach to obtain a closed form solution for the queue length distribution, we will see that it does not generalize well to other job size distributions. Moreover, due to the Poisson Arrivals See Time Averages (PASTA) property it follows that the queue length at arrival times is equal to the queue length, that is  $u_k^a = u_k$ , where  $u_k^a$  is the probability that an incoming job finds the M/M/1 queue with  $k$  or more jobs. Throughout this thesis we often use the PASTA property without explicitly mentioning its use.

## A.2 Stationary distribution of a continuous time Markov chain

In this section, we construct the Continuous Time Markov Chain (CTMC) which is associated to the M/M/1 queue. That is, we construct a rate matrix  $Q$  such that  $Q_{i,j}$  represents the rate at which we have a transition from state (i.e. queue length)  $i$  to state  $j$ . Moreover, one defines  $Q_{i,i} = -\sum_{j \neq i} Q_{i,j}$  for all states  $i$ . We find that the non-zero elements of  $Q$  are given by:

- $Q_{i,i+1} = \lambda$ , at each queue length  $i \geq 0$ , arrivals occur to the queue at a rate equal to  $\lambda$  which corresponds to a state transition from length  $i$  to length  $i + 1$ .
- $Q_{i,i-1} = \mu$ , at each non-zero queue length  $i \geq 1$ , jobs leave the queue at a rate equal to  $1$  which corresponds to a state transition from length  $i$  to  $i - 1$ .

- By definition we have  $Q_{0,0} = -\lambda$  and  $Q_{i,i} = -(\lambda + \mu)$  for all  $i \geq 1$ , which represents the rate at which we leave each state  $i$ .

Denoting by  $x_i$  the probability that the queue length is equal to  $i$  at an arbitrary point in time, one finds from theory on CTMCs that if we let  $x$  denote the row vector consisting of the  $x_i$  values,  $x$  is the unique stochastic vector which satisfies the equation  $xQ = 0$ . From this one can recover the formula's we previously obtained. However, we illustrate a numerical method to compute the values  $x_i$  which extends well to Phase Type job sizes.

For any CTMC with transition rate matrix  $Q$ , one can consider the embedded Discrete Time Markov Chain (DTMC), this DTMC is defined by the transition matrix  $P = \frac{Q}{\max\{-Q_{i,i}\}} + I$ , with  $I$  the identity matrix. For the M/M/1 queue, it is not hard to see that the non-zero elements of  $P$  are given by:

- $P_{i,i+1} = \frac{\lambda}{\lambda+\mu}$ , at each queue length  $i \geq 0$  the next event is a job arrival with probability  $\frac{\lambda}{\lambda+\mu}$
- $P_{i,i-1} = \frac{\mu}{\lambda+\mu}$ , at each non-zero queue length  $i \geq 1$  the next event is a job departure with probability  $\frac{\mu}{\lambda+\mu}$ .
- $P_{0,0} = \frac{\mu}{\lambda+\mu}$ , when the next event is a job departure but the queue is already empty, we simply remain in queue length 0.

For a DTMC, for any value  $n \in \mathbb{N}$ , one can interpret  $P^n$  as the system after  $n$  transitions have occurred. As we are interested in the *long term behaviour* of the M/M/1 queue, we are interested in  $\lim_{n \rightarrow \infty} P^n$ . For an irreducible and aperiodic DTMC one finds that the columns of the matrix  $\bar{P} = \lim_{n \rightarrow \infty} P^n$  are all equal, and any row of the matrix  $\bar{P}$  represents the stationary distribution of  $P$ . That is if we denote by  $\bar{x}$  an arbitrary row of  $\bar{P}$ , one has  $\bar{x}P = \bar{x}$ . From this one finds that:

$$xP = x \Rightarrow \bar{x} \left( \frac{Q}{\max\{-Q_{i,i}\}} + I \right) = \bar{x} \Rightarrow \bar{x}Q = 0,$$

that is  $\bar{x}$  is the stationary distribution  $x$  of the CTMC  $Q$ . Therefore  $x_i$  denotes the probability that the M/M/1 queue has exactly  $i$  jobs in its queue (and we have  $u_k = \sum_{\ell \geq k} x_\ell$ ).

**Remark 61.** *Computing the matrix  $\bar{P} = \lim_{n \rightarrow \infty} P^n$  can be done very quickly by recursively computing  $P_{n+1} = P_n^2$ .*

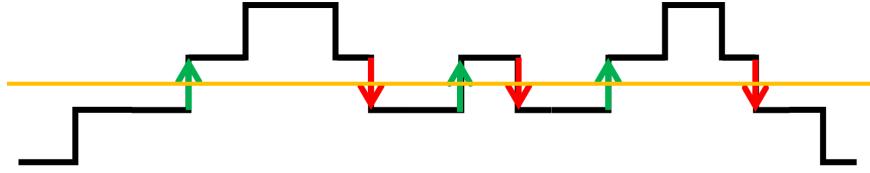


Figure A.3: Drawing indicating that the distribution at arrival times is equal to the distribution at departure times.

**Remark 62.** For the  $M/M/1$  queue one could also easily verify that  $xQ = 0$  and  $xP = x$  for the choice  $x_k = u_k - u_{k+1} = \lambda^k(1-\lambda)$ . However the main goal of this section was to illustrate the numerical method as it generalizes to more complex models. In particular, when jobs have a Phase Type distribution the method proposed in this section requires little extra work.

### A.3 Observing the System at completion times

Alternatively, we could only observe the system at completion times, that is, right after a job completes service. First, we argue that the queue length distribution as seen by an arriving customer is equal to the queue length distribution at departure times, this can best be seen from a drawing, see Figure A.3. Indeed, we observe in Figure A.3 that each time we have an arrival to a queue of length  $k$ , we must (some time in the future) also have a departure from a queue with length  $k + 1$ . Therefore we can write  $x_k, u_k$  for the stationary distribution representing the queue length at departure times, arrival time or at an arbitrary point in time interchangeably. Let us now compute these values with the interpretation of queue length at departure times.

When the queue length after the previous job departure was equal to  $k \geq 1$ , the queue length after the next departure will be equal to a value  $\ell \geq k - 1$ . Indeed its queue length will be equal to  $k - 1$  if there are no arrivals before the next departure, and it will be equal to  $k + a - 1$  if  $a \geq 0$  arrivals occurred before the next departure. When the queue was empty after the previous departure, its length after the next departure is equal to  $a - 1$  with  $a \geq 1$  the number of arrivals since the previous departure. However, for this last case, we can always assume that there is an instant arrival after

the queue becomes empty as there will be no job departures as long as the queue is empty. Therefore we can represent the transition rate matrix of the M/M/1 queue with this interpretation we find that the DTMC corresponding to  $x$  is given by the transition matrix:

$$P = \begin{pmatrix} a_0 & a_1 & a_2 & a_3 & \dots \\ a_0 & a_1 & a_2 & a_3 & \dots \\ 0 & a_0 & a_1 & a_2 & \dots \\ 0 & 0 & a_0 & a_1 & \dots \\ 0 & 0 & 0 & a_0 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

We therefore have  $xP = x$ . It only remains to compute the transition probabilities  $a_k$ , to this end we note that  $a_k$  is equal to the probability that we have exactly  $k$  arrivals in the interval  $[0, X]$ , where  $X$  is exponentially distributed with mean one:

$$\begin{aligned} a_k &= \mathbb{P}\{k \text{ arrivals in the interval } [0, X]\} \\ &= \int_0^\infty \mathbb{P}\{k \text{ arrivals in the interval } [0, x]\} e^{-x} dx \\ &= \int_0^\infty e^{-\lambda x} \frac{(\lambda x)^k}{k!} e^{-x} dx \\ &= \frac{\lambda^k}{(1 + \lambda)^{k+1}}. \end{aligned}$$

One could now easily verify that  $xP = x$  with  $x_k = u_k - u_{k+1} = \lambda^k(1 - \lambda)$ . Alternatively, one could again compute  $\bar{P}$  as in the previous section and from this one could numerically recover  $x$ . The advantage of this method is that, as we only observe the system at completion times, we can carry out this analysis for any job size distribution

## A.4 Level crossing - workload

All of the methods discussed so far keep track of the number of jobs in the M/M/1 queue. In contrast, we can also describe the behaviour of the M/M/1 queue by solely relying on the workload distribution, where we define the workload as the time left until the queue becomes empty. This process is

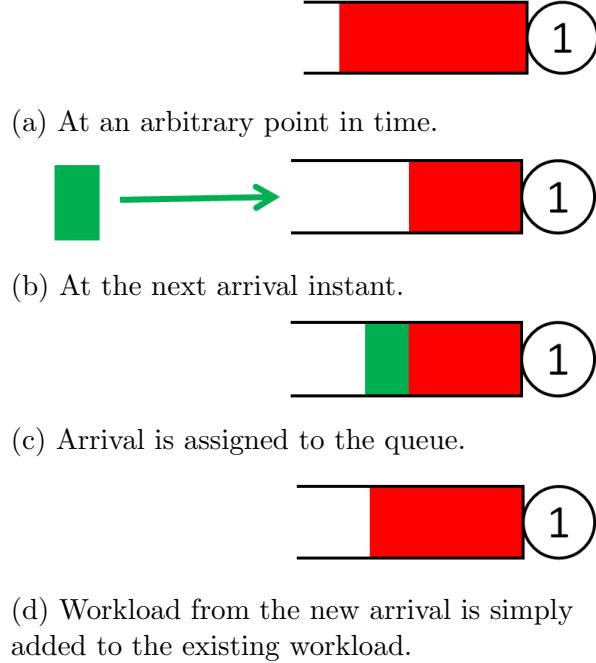


Figure A.4: Graphical depiction of the M/M/1 queue using the workload as its state descriptor.

graphically represented in Figure A.4, with this interpretation, the M/M/1 queue has a certain workload  $w$  at any point in time (represented by the red area), this workload gets depleted at a constant rate equal to one and jobs of size  $\text{Exp}(1)$  arrive at  $\text{Poisson}(\lambda)$  distributed points in time. For this interpretation, we associate to the M/M/1 queue some probability density function (pdf)  $f(w)$ , cumulative distribution function (cdf)  $F(w) = F(0) \int_0^w f(u) du$  and complemented cdf  $\bar{F}(w) = 1 - F(w)$ . Note that the pdf is not an actual pdf as  $F(0) \neq 0$  is the probability that the M/M/1 queue is empty and:

$$\int_0^\infty f(w) dw = 1 - F(0) = \bar{F}(0) < 1.$$

We formulate a level crossing argument for the M/M/1 queue, to this end, we set an arbitrary bound at  $w > 0$  and an arbitrary (but small)  $\Delta > 0$ . We now observe the system at the time instances  $\Delta \cdot n$  for  $n \geq 0$ . We note that there is a down-crossing whenever the queue has a workload which lies in the interval  $[w, w + \Delta]$ , that is we have a down-crossing with probability

$\bar{F}(w) - \bar{F}(w + \Delta)$ . As arrivals occur with rate  $\lambda$ , we have an arrival in each time step with probability  $\lambda\Delta$ . For the arrivals we consider two cases:

- The queue is empty, in this case the job size must be at least  $w$  to have an up-crossing, this happens with probability  $\lambda\Delta F(0)\bar{G}(w)$ , with  $\bar{G}(w)$  the ccdf of the exponential distribution.
- The queue has some workload  $u \in (0, w)$ , in this case the job size must be at least  $w - u$  to have an up-crossing, this happens with probability  $\lambda\Delta \int_0^w f(u)\bar{G}(w-u) du$ .

We find that:

$$\bar{F}(w) - \bar{F}(w + \Delta) = \lambda\Delta F(0)\bar{G}(w) + \lambda\Delta \int_0^w f(u)\bar{G}(w-u) du,$$

dividing both sides by  $\Delta$  and taking the limit  $\Delta \rightarrow 0^+$  on both sides allows us to conclude that:

$$\bar{F}'(w) = \lambda \left( F(0)\bar{G}(w) + \int_0^w f(u)\bar{G}(w-u) du \right).$$

Applying integration by parts on the integral, one finds that the above expression is equivalent to:

$$\bar{F}'(w) = -\lambda \left( \bar{G}(w) - \bar{F}(w) + \int_0^w \bar{F}(u)g(w-u) du \right). \quad (\text{A.1})$$

This expression can easily be solved numerically using the boundary condition  $\bar{F}(0) = \lambda$ . One can see that this equality indeed holds by noting that work arrives at a rate equal to  $\lambda$  while work leaves the system at rate  $\bar{F}(0) \cdot 1$  and the amount of work leaving the system must equal the amount of work entering the system.

We now integrate both sides of (A.1) from  $w$  to infinity, we find by applying Fubini's theorem (which is allowed as the integrand is positive) that:

$$\begin{aligned} \int_w^\infty \bar{F}'(u) du &= -\lambda \left[ \int_w^\infty \bar{G}(u) du - \int_w^\infty \bar{F}(w) dw + \int_w^\infty \int_0^u \bar{F}(v)g(u-v) dv du \right] \\ &= -\lambda \cdot \left[ \int_w^\infty \bar{G}(u) du - \int_w^\infty \bar{F}(w) dw + \int_0^\infty \int_{\max\{w,u\}}^\infty \bar{F}(v)g(u-v) du dv \right] \\ &= -\lambda \left[ \int_w^\infty \bar{G}(u) du - \int_w^\infty \bar{F}(u) du + \int_w^\infty \bar{F}(u) du + \int_0^w \bar{G}(w-u) \bar{F}(u) du \right] \end{aligned}$$

From this, we are able to conclude that (using the fact that  $1 = \mathbb{E}[G] = \int_0^\infty \bar{G}(w) dw$ ) :

$$\begin{aligned}\bar{F}(w) &= \lambda \left[ 1 - \int_0^w \bar{G}(w-u) du + \int_0^w \bar{F}(u) \bar{G}(w-u) du \right] \\ &= \lambda \left[ 1 - \int_0^w (1 - \bar{F}(u)) \bar{G}(w-u) du \right].\end{aligned}$$

This is a FPE which can also be used to solve for  $\bar{F}(w)$  numerically. When job sizes are exponential with mean one, we find that  $\bar{G}(w-u) = g(w-u)$ , therefore it follows from the above equation that:

$$\int_0^w \bar{F}(u) g(w-u) du = \frac{\bar{F}(w)}{\lambda} + G(w) - 1,$$

one can now use this equation to further simplify (A.1), indeed from this we find:

$$\bar{F}'(w) = -(1-\lambda) \cdot \bar{F}(w),$$

which can easily be solved explicitly, and we find that:

$$\bar{F}(w) = \lambda e^{-(1-\lambda)w}.$$

From this one can easily compute the mean workload which is given by  $\int_0^\infty \bar{F}(w) dw = \frac{\lambda}{1-\lambda}$ , which is exactly equal to the mean queue length!

# Bibliography

- [1] A. K. A. Mukhopadhyay and R. R. Mazumdar. Randomized assignment of jobs to servers in heterogeneous clusters of shared servers for low delay. *Stochastic Systems*, 6(1):90 – 131, 2016.
- [2] M. Abramowitz and I. A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover, New York, ninth dover printing, tenth gpo printing edition, 1964.
- [3] R. Aghajani, X. Li, and K. Ramanan. Mean-field dynamics of load-balancing networks with general service distributions. *CoRR*, abs/1512.05056v2, 2016.
- [4] R. Aghajani, X. Li, and K. Ramanan. The pde method for the analysis of randomized load balancing networks. *Proc. ACM Meas. Anal. Comput. Syst.*, 1(2):38:1–38:28, Dec. 2017.
- [5] R. Aghajani, X. Li, and K. Ramanan. The pde method for the analysis of randomized load balancing networks. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(2):38, 2017.
- [6] G. Ananthanarayanan, A. Ghodsi, S. Shenker, and I. Stoica. Effective straggler mitigation: Attack of the clones. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation*, nsdi’13, pages 185–198, Berkeley, CA, USA, 2013. USENIX Association.
- [7] G. Ananthanarayanan, A. Ghodsi, S. Shenker, and I. Stoica. Effective straggler mitigation: Attack of the clones. In *NSDI*, volume 13, pages 185–198, 2013.

- [8] J. Anselmi and F. Dufour. Power-of-d-choices with memory: Fluid limit and optimality. *Mathematics of Operations Research*, 2020.
- [9] E. Anton, U. Ayesta, M. Jonckheere, and I. M. Verloop. On the stability of redundancy models. *arXiv preprint arXiv:1903.04414*, 2019.
- [10] U. Ayesta, T. Bodas, J. Dorsman, and I. Verloop. A token-based central queue with order-independent service rates. *arXiv preprint arXiv:1902.02137*, 2019.
- [11] U. Ayesta, T. Bodas, and I. Maria Verloop. On a unifying product form framework for redundancy models. *Performance Evaluation*, 127:93–119, 2018.
- [12] U. Ayesta, T. Bodas, and I. Verloop. On a unifying product form framework for redundancy models. 2018.
- [13] Y. Azar, A. Broder, A. Karlin, and E. Upfal. Balanced allocations. *SIAM Journal on Computing*, 29(1):180–200, 1999.
- [14] R. Bekker, S. Borst, O. Boxma, and O. Kella. Queues with workload-dependent arrival and service rates. *Queueing Systems*, 46(3):537–556, Mar 2004.
- [15] M. Benaim and J. Le Boudec. A class of mean field interaction models for computer and communication systems. *Performance Evaluation*, 65(11-12):823–838, 2008.
- [16] M. Benaim and J.-Y. Le Boudec. A class of mean field interaction models for computer and communication systems. *Performance evaluation*, 65(11-12):823–838, 2008.
- [17] D. Blackwell. The range of certain vector integrals. *Proceedings of the American Mathematical Society*, 2(3):390–395, 1951.
- [18] J. Borcea. Equilibrium points of logarithmic potentials induced by positive charge distributions. i. generalized de bruijn-springer relations. *Transactions of the American Mathematical Society*, 359(7):3209–3237, 2007.
- [19] M. Bramson. Stability of join the shortest queue networks. *Ann. Appl. Probab.*, 21(4):1568–1625, 2011.

- [20] M. Bramson, Y. Lu, and B. Prabhakar. Randomized load balancing with general service time distributions. In *ACM SIGMETRICS 2010*, pages 275–286, 2010.
- [21] M. Bramson, Y. Lu, and B. Prabhakar. Asymptotic independence of queues under randomized load balancing. *Queueing Systems*, 71(3):247–292, 2012.
- [22] M. Bramson, Y. Lu, and B. Prabhakar. Asymptotic independence of queues under randomized load balancing. *Queueing Syst.*, 71(3):247–292, 2012.
- [23] M. Bramson, Y. Lu, and B. Prabhakar. Decay of tails at equilibrium for fifo join the shortest queue networks. *Ann. Appl. Probab.*, 23(5):1841–1878, 10 2013.
- [24] A. Braverman. Steady-state analysis of the join the shortest queue model in the halfin-whitt regime. *arXiv preprint arXiv:1801.05121*, 2018.
- [25] G. Brightwell and M. Luczak. The supermarket model with arrival rate tending to one. *arXiv preprint arXiv:1201.5523*, 2012.
- [26] Q. Bu, L. Liu, J. Tang, and Y. Q. Zhao. Approximations for a queueing game model with join-the-shortest-queue strategy. *arXiv preprint arXiv:2012.14955*, 2020.
- [27] P. Delgado, D. Didona, F. Dinu, and W. Zwaenepoel. Job-aware scheduling in eagle: Divide and stick to your probes. In *Proceedings of the Seventh ACM Symposium on Cloud Computing*, pages 497–509, 2016.
- [28] P. Delgado, F. Dinu, A.-M. Kermarrec, and W. Zwaenepoel. Hawk: Hybrid datacenter scheduling. In *2015 {USENIX} Annual Technical Conference ({USENIX}){ATC} 15*, pages 499–510, 2015.
- [29] R. D. Driver. *Ordinary and Delay Differential Equations*. Springer-Verlag, Berlin-Heidelberg-New York, 1977.
- [30] P. Eschenfeldt and D. Gamarnik. Join the shortest queue with many servers. the heavy-traffic asymptotics. *Mathematics of Operations Research*, 43(3):867–886, 2018.

- [31] S. Ethier and T. Kurtz. *Markov processes: characterization and convergence*. Wiley, 1986.
- [32] Y. Fang. Hyper-erlang distribution model and its application in wireless mobile networks. *Wireless Networks*, 7(3):211–219, 2001.
- [33] S. Foss and N. Chernova. On the stability of a partially accessible multi-station queue with state-dependent routing. *Queueing Syst. Theory Appl.*, 29(1):55–73, May 1998.
- [34] S. Foss and A. L. Stolyar. Large-scale join-idle-queue system with general service times. *Journal of Applied Probability*, 54(4):995–1007, 2017.
- [35] S. Foss and A. L. Stolyar. Large-scale join-idle-queue system with general service times. *Journal of Applied Probability*, 54(4):995–1007, 2017.
- [36] D. Gamarnik, J. N. Tsitsiklis, M. Zubeldia, et al. A lower bound on the queueing delay in resource constrained load balancing. *Annals of Applied Probability*, 30(2):870–901, 2020.
- [37] K. Gardner, M. Harchol-Balter, and A. Scheller-Wolf. A better model for job redundancy: Decoupling server slowdown and job size. MAS-COTS, 2016.
- [38] K. Gardner, M. Harchol-Balter, A. Scheller-Wolf, and B. Van Houdt. A better model for job redundancy: Decoupling server slowdown and job size. *IEEE/ACM Trans. Netw.*, 25(6):3353–3367, Dec. 2017.
- [39] K. Gardner, M. Harchol-Balter, A. Scheller-Wolf, M. Velednitsky, and S. Zbarsky. Redundancy-d: The power of d choices for redundancy. *Operations Research*, 65(4):1078–1094, 2017.
- [40] K. Gardner, S. Zbarsky, M. Harchol-Balter, and A. Scheller-Wolf. The power of d choices for redundancy. In *Proceedings of the 2016 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science*, pages 409–410, 2016.
- [41] N. Gast and G. Bruno. A mean field model of work stealing in large-scale systems. In *ACM SIGMETRICS Performance Evaluation Review*, volume 38, pages 13–24. ACM, 2010.

- [42] N. Gast and B. Van Houdt. Transient and steady-state regime of a family of list-based cache replacement algorithms. In *ACM SIGMETRICS 2015*, 2015.
- [43] N. Gast and B. Van Houdt. A refined mean field approximation. *Proc. ACM Meas. Anal. Comput. Syst.*, 1(2):33:1–33:28, Dec. 2017.
- [44] M. Harchol-Balter. *Performance modeling and design of computer systems: queueing theory in action*. Cambridge University Press, 2013.
- [45] M. Harchol-Balter. *Performance modeling and design of computer systems: queueing theory in action*. Cambridge University Press, 2013.
- [46] T. Hellemans, T. Bodas, and B. Van Houdt. Performance analysis of workload dependent load balancing policies. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 3(2):35, 2019.
- [47] T. Hellemans and B. Van Houdt. On the power-of-d-choices with least loaded server selection. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 2(2):1–22, 2018.
- [48] R. Jinan, A. Badita, T. Bodas, and P. Parag. Load balancing policies with server-side cancellation of replicas. *arXiv preprint arXiv:2010.13575*, 2020.
- [49] G. Joshi, Y. Liu, and E. Soljanin. Coding for fast content download. In *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*, pages 326–333. IEEE, 2012.
- [50] G. Joshi, E. Soljanin, and G. Wornell. Efficient redundancy techniques for latency reduction in cloud systems. *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS)*, 2(2):12, 2017.
- [51] J. L. Kaplan, M. Sorg, and J. A. Yorke. Solutions of  $x'(t) = f(x(t))$ ,  $x(t-l)$  have limits when  $f$  is an order relation. *Nonlinear Analysis: Theory, Methods & Applications*, 3(1):53–58, 1979.
- [52] J. Krieger and P. Buchholz. *PH and MAP Fitting with Aggregated Traffic Traces*, pages 1–15. Springer International Publishing, Cham, 2014.

- [53] J. Kriege and P. Buchholz. Ph and map fitting with aggregated traffic traces. In *Measurement, Modelling, and Evaluation of Computing Systems and Dependability and Fault Tolerance*, pages 1–15. Springer, 2014.
- [54] T. Kurtz. *Approximation of population processes*. Society for Industrial and Applied Mathematics, 1981.
- [55] G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods and stochastic modeling*. SIAM, Philadelphia, 1999.
- [56] G. Latouche and V. Ramaswami. *Introduction to matrix analytic methods in stochastic modeling*, volume 5. Siam, 1999.
- [57] X. Liu, K. Gong, and L. Ying. Steady-state analysis of load balancing with coxian-2 distributed service times. *arXiv preprint arXiv:2005.09815*, 2020.
- [58] X. Liu and L. Ying. Steady-state analysis of load-balancing algorithms in the sub-halfin–whitt regime. *Journal of Applied Probability*, 57(2):578–596, 2020.
- [59] 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.*, 68:1056–1071, 2011.
- [60] 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. *Performance Evaluation*, 68(11):1056–1071, 2011.
- [61] A. W. Marshall, I. Olkin, and B. C. Arnold. *Inequalities: theory of majorization and its applications*, volume 143. Springer, 1979.
- [62] W. Minnebo and B. Van Houdt. A fair comparison of pull and push strategies in large distributed networks. *IEEE/ACM Transactions on Networking*, 22:996–1006, 2014.
- [63] M. Mitzenmacher. *The Power of Two Choices in Randomized Load Balancing*. PhD thesis, University of California, Berkeley, 1996.
- [64] M. Mitzenmacher. How useful is old information? *IEEE Trans. Parallel Distrib. Syst.*, 11(1):6–20, Jan. 2000.

- [65] M. Mitzenmacher. Analyses of load stealing models based on families of differential equations. *Theory of Computing Systems*, 34:77–98, 2001.
- [66] M. Mitzenmacher. The power of two choices in randomized load balancing. *IEEE Transactions on Parallel and Distributed Systems*, 12(10):1094–1104, 2001.
- [67] M. Mitzenmacher. The power of two choices in randomized load balancing. *IEEE Trans. Parallel Distrib. Syst.*, 12:1094–1104, October 2001.
- [68] M. Mitzenmacher. The supermarket model with known and predicted service times. *arXiv preprint arXiv:1905.12155*, 2019.
- [69] M. Mitzenmacher. Queues with small advice. *arXiv preprint arXiv:2006.15463*, 2020.
- [70] M. Mitzenmacher, B. Prabhakar, and D. Shah. Load balancing with memory. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings.*, pages 799–808. IEEE, 2002.
- [71] D. Mukherjee, S. Borst, J. van Leeuwaarden, and P. Whiting. Universality of power-of-d load balancing schemes. *SIGMETRICS Perform. Eval. Rev.*, 44(2):36–38, Sept. 2016.
- [72] J. R. Norris and J. R. Norris. *Markov chains*. Number 2. Cambridge university press, 1998.
- [73] K. Ousterhout, P. Wendell, M. Zaharia, and I. Stoica. Sparrow: distributed, low latency scheduling. In *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, pages 69–84. ACM, 2013.
- [74] K. Ousterhout, P. Wendell, M. Zaharia, and I. Stoica. Sparrow: Distributed, low latency scheduling. In *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles*, SOSP ’13, pages 69–84, New York, NY, USA, 2013. ACM.
- [75] A. Panchenko and A. Thümmler. Efficient phase-type fitting with aggregated traffic traces. *Perform. Eval.*, 64(7-8):629–645, Aug. 2007.

- [76] M. Pinsky and S. Karlin. *An introduction to stochastic modeling*. Academic press, 2010.
- [77] Y. Raaijmakers, S. Borst, and O. Boxma. Delta probing policies for redundancy. *Performance Evaluation*, 127-128:21 – 35, 2018.
- [78] Y. Raaijmakers, S. Borst, and O. Boxma. Redundancy scheduling with scaled bernoulli service requirements. *arXiv preprint*, 2018. arXiv:1811.06309.
- [79] Z. Schuss. *Theory and applications of stochastic processes: an analytical approach*, volume 170. Springer Science & Business Media, 2009.
- [80] N. B. Shah, K. Lee, and K. Ramchandran. When do redundant requests reduce latency? *IEEE Transactions on Communications*, 64(2):715–722, Feb 2016.
- [81] N. B. Shah, K. Lee, and K. Ramchandran. When do redundant requests reduce latency? *IEEE Transactions on Communications*, 64(2):715–722, 2016.
- [82] V. Shah, A. Bouillard, and F. Baccelli. Delay comparison of delivery and coding policies in data clusters. In *Communication, Control, and Computing (Allerton), 2017 55th Annual Allerton Conference on*, pages 397–404. IEEE, 2017.
- [83] S. Shneer and A. Stolyar. Large-scale parallel server system with multi-component jobs. *arXiv preprint arXiv:2006.11256*, 2020.
- [84] A. Stolyar. Pull-based load distribution in large-scale heterogeneous service systems. *Queueing Systems*, 80(4):341–361, 2015.
- [85] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific J. Math.*, 5(2):285–309, 1955.
- [86] M. van der Boor, S. Borst, and J. van Leeuwaarden. Hyper-scalable jsq with sparse feedback. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 3(1):1–37, 2019.
- [87] M. van der Boor, S. C. Borst, J. S. van Leeuwaarden, and D. Mukherjee. Scalable load balancing in networked systems: A survey of recent advances. *arXiv preprint arXiv:1806.05444*, 2018.

- [88] B. Van Houdt. A mean field model for a class of garbage collection algorithms in flash-based solid state drives. *ACM SIGMETRICS Perform. Eval. Rev.*, 41(1):191–202, 2013.
- [89] B. Van Houdt. Performance of garbage collection algorithms for flash-based solid state drives with hot/cold data. *Perform. Eval.*, 70(10):692–703, 2013.
- [90] B. Van Houdt. Global attraction of ode-based mean field models with hyperexponential job sizes. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 3(2):23, 2019.
- [91] B. Van Houdt and L. Bortolussi. Fluid limit of an asynchronous optical packet switch with shared per link full range wavelength conversion. In *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems*, pages 113–124, 2012.
- [92] I. Van Spilbeeck and B. Van Houdt. Performance of rate-based pull and push strategies in heterogeneous networks. *Performance Evaluation*, 91:2–15, 2015.
- [93] I. Van Spilbeeck and B. Van Houdt. On the impact of job size variability on heterogeneity-aware load balancing. In *International Conference on Queueing Theory and Network Applications*, pages 193–215. Springer, 2018.
- [94] T. Vasantam, A. Mukhopadhyay, and R. R. Mazumdar. The mean-field behavior of processor sharing systems with general job lengths under the sq (d) policy. *Performance Evaluation*, 127:120–153, 2018.
- [95] N. Vvedenskaya, R. Dobrushin, and F. Karpelevich. Queueing system with selection of the shortest of two queues: an asymptotic approach. *Problemy Peredachi Informatsii*, 32:15–27, 1996.
- [96] H. Wadsworth Gould. *Combinatorial Identities: A standardized set of tables listing 500 binomial coefficient summations*. Morgantown, W Va, 1972.

- [97] W. Wang, M. Harchol-Balter, H. Jiang, A. Scheller-Wolf, and R. Srikant. Delay asymptotics and bounds for multi-task parallel jobs. 2017.
- [98] W. Winston. Optimality of the shortest line discipline. *Journal of Applied Probability*, 14(1):181–189, 1977.
- [99] Q. Xie, X. Dong, Y. Lu, and R. Srikant. Power of d choices for large-scale bin packing: A loss model. *SIGMETRICS Perform. Eval. Rev.*, 43(1):321–334, June 2015.
- [100] L. Ying, R. Srikant, and X. Kang. The power of slightly more than one sample in randomized load balancing. *Mathematics of Operations Research*, 42(3):692–722, 2017.