THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :

Institut National Polytechnique de Toulouse (Toulouse INP)

Discipline ou spécialité :

Mathématiques Appliquées

Présentée et soutenue par :

Mme ELENE ANTON

le mercredi 2 juin 2021

Titre :

Performance analysis of redundancy and mobility in multi-server systems

---

Ecole doctorale :

Mathématiques, Informatique, Télécommunications de Toulouse (MITT)

Unité de recherche :

Institut de Recherche en Informatique de Toulouse (IRIT)

Directeur(s) de Thèse :

M. URTZI AYESTA

MME INA MARIA VERLOOP

Rapporteurs :

M. BENNY VAN HOUDT, UNIVERSITE INSTELLING ANTWERPEN

M. ESA HYYTIA, UNIVERSITE D'ISLANDE

Membre(s) du jury :

M. FLORIAN SIMATOS, ISAE-SUPAERO, Président

MME ANA BUSIC, INRIA, Membre

MME INA MARIA VERLOOP, TOULOUSE INP, Membre

MME RHONDA RIGHTER, UNIVERSITY OF CALIFORNIA BERKELEY CA EU, Membre

M. URTZI AYESTA, TOULOUSE INP, Membre

Acknowledgments

I would like to express my sincere gratitude to Urtzi and Maaike, my Ph.D. advisors, for all the support, patience, and motivation during my thesis. For encouraging me to participate in many research activities that have enriched, even more, this experience. Thank you so much for always challenging me to go further: without your help, this would not have been possible. I would also like to thank IRIT-CNRS for funding my Ph.D. thesis, research visits, and conferences.

I would also like to express my sincere gratitude to Matthieu Jonckheere, with whom the three of us have worked side-by-side during my thesis. For the opportunity to discuss and learn from him. To him and his family for their hospitality during my visits to the University of Buenos Aires. I would also like to thank the STIC AMSUD GENE project for funding these visits to Buenos Aires.

I am grateful to Professor Mor Harchol-Balter, for the opportunity to join her team, as well as her hospitality, during a two-month visit at Carnegie Mellon University. I would also like to thank INP and CNRS (Gdr RO) for partially funding this visit.

I would also like to thank Professor Rhonda Righter, who visited Enseeiht in 2020. For the opportunity to work and learn from her during and after her visit.

I would like to thank the jury members for accepting the invitation, reading the thesis, and providing me with valuable comments.

I would also like to thank the permanents and students of my research group RMESS at Enseeiht. To those that were there from the beginning; Oana, Amal, Kevin, Adrien, and Mohamed; and to the most recent incorporations; Romain, Mohamed, Justin, Firmin, Chaima, and Guillaume. Thank you for all the coffee breaks, meals, and seminars that we shared. A special thank you, Santi, my Ph.D. big brother. For all the tips and help, when I first arrived at Enseeiht, and all the fun, the mate, the valuable discussions, and stress moments that we shared afterward. I also thank Marie Chabert for letting me teach in her group SC. Finally, I also thank other Ph.D. students I met during my thesis, Youri and Cèline.

Je veux egalement remercier à mes ami.e.s de Toulouse, pour faire de Toulouse mon deuxième chez moi. Specially to my group of friends Silvia, Elena, Hector, Bea, Gui, Marta, Niti, Marco and Luna. Era berean, Toulouseko Euskaldun taldeari, eta bereziki

mus + apèroren (apèroarena musarena baino gehiago, egia esan) fan garen taldetxo horri: Olaia, Fred, Jon, Bedou, Julie, Anna, Yoann, Kepa, Sandrine eta Bittor. Un grand merci à mes colocs de 17 Arcole: Anna, Kane, Isa, Estelle, Santi, Jade, James et Felix. Pour tous ces moments partagés au quotidien, pour faire que le nombreux confinements/couvre-feus qu'on à vécus ensemble étaient même des moments de bonheur. Ainsi, pour tout votre support dans les derniers mois de ma thèse. Eta amaitzeko zuri, Eneko, zein momentu onak pasa ditugun eta zein garrantzitsua izan zaren niretzako tesiaren azkeneko urtean zehar.

Azkenik, etxekoei ere eskerrak eman nahi dizkizuet. Tesia atzerrian bizitza berri bat eraikitzearekin batera hasi zen, hain nirea sentitzen dudan Donosti utziz. Ezagutzen nauzuenok badakizue Donosti dela ene bihotzeko hiria, beti bueltatuko naizen hiria. Hau ez da bere hiru hondartzengatik, edo tanborradarengatik (pixka bat bai quand même), baizik eta bertan ditudan bizipenengatik, familiarengatik eta lagunengatik. Horregatik, etxekoei eskerrak eman nahi dizkizuet, urrun egon arren bueltatu naizenetan etxean bezela sentiarazi nauzuelako. Familiari eskerrak eman nahi dizkizuet eta bereziki gurasoei, zuen maitasun amaigabe, animo eta urduritasun momentu denak sufritzegatik. Zuei aita eta ama, zuek gabe ez nintzen hain urrun iritsiko. Baita ere eskerrak eman nahi dizkizuet Donostiko lagunei eta abentura honen gora-beherak nirekin gainditu dituzuenoi. Bereziki zuei: Ane, Alba, Leire, Maite, Eva, Larraitz, Manu eta Luli.

Nik Bilbon ikasi nuen Matematika, eta badakit abentura hau ez zela posible izango nire karrerako lagunak gabe. Horregatik, esker bereziak eman nahi dizkizuet txitxiburduntzi taldeari (Amaia, Lore, Rut, Jone, Sara, Lara eta Ibon), Masterreko lagunei (Bruno eta Abel) eta Unamunoko lagunei (Ane, Leire, Anartz, Xegu, Muñoa, Huizi, Jonpe, Lander, Harkaitz, Arru), zuen konfiantza eta animoengatik. Plazer bat da zuekin jarraitzea.

Abstract

In this thesis, we studied how both redundancy and mobility impact the performance of computer systems and cellular networks, respectively. The general notion of redundancy is that upon arrival each job dispatches copies into multiple servers. This allows exploiting the variability of the queue lengths and server capacities in the system. We consider redundancy models with both identical and i.i.d. copies. When copies are i.i.d., we show that with PS and ROS, redundancy does not reduce the stability region. When copies are identical, we characterize the stability condition for systems where either FCFS, PS, or ROS is implemented in the servers. We observe that this condition strongly depends on the scheduling policy implemented in the system. We then investigate how redundancy impacts the performance by comparing it to a non-redundant system. We observe that both the stability and performance improve considerably under redundancy as the heterogeneity of the server capacities increases. Furthermore, for both i.i.d. and identical copies, we characterize redundancy-aware scheduling policies that improve both the stability and performance. Finally, we identify several open problems that might be of interest to the community.

User mobility in wireless networks addresses the fact that users in a cellular network switch from cell to cell when geographically moving in the system. We control the mobility speed of the users among the servers and analyze how mobility impacts the performance at a user level. We observe that the performance of the system under fixed mobility speed strongly depends on the inherent parameters of the system.

Keywords: Load balancing, stability, performance, redundancy, mobility.

Résumé

Dans cette thèse, nous avons étudié l'impact de la redondance et de la mobilité sur les performances des systèmes informatiques et des réseaux cellulaires, respectivement. La notion générale de redondance est qu'à l'arrivée, chaque tâche envoie des copies dans plusieurs serveurs. Cela permet d'exploiter la variabilité de la longueur des files d'attente et des capacités du serveur dans le système. Nous considérons des modèles de redondance où les tâches ont soit des copies i.i.d. ou des copies identiques. Lorsque des copies sont i.i.d., nous montrons que la région de stabilité n'est pas réduit quand PS ou ROS est mis en œuvre. Lorsque les copies sont identiques, nous caractérisons la condition de stabilité pour les systèmes où FCFS, PS ou ROS est mis en œuvre dans les serveurs. Nous observons que cette condition dépend fortement de la discipline de service. Nous examinons ensuite l'incidence de la redondance sur le rendement en la comparant à celle d'un système où il n'y a pas de redondance. Nous observons que la stabilité et les performances sont considérablement améliorées sous l'effet de la redondance, à mesure que l'hétérogénéité des capacités du serveur augmente. De plus, pour les systèmes avec des copies i.i.d. et des copies identiques, nous caractérisons des disciplines de service prenant en compte la redondance qui peuvent améliorer à la fois la stabilité et les performances du système. Enfin, nous identifions plusieurs problèmes ouverts qui pourraient intéresser la collectivité.

La mobilité des utilisateurs dans les réseaux sans fil rend compte du fait que les utilisateurs d'un réseau cellulaire passent d'une cellule à l'autre lorsqu'ils se déplacent géographiquement dans le système. Nous contrôlons la vitesse de mobilité des utilisateurs parmi les serveurs et analysons comment la mobilité affecte les performances au niveau de l'utilisateur. Nous observons que la performance du système à vitesse de mobilité constante dépend fortement des paramètres inhérents au système.

Mots clés: Équilibrage de charge, stabilité, performance, redondance, mobilité.

Table of contents

Chapter 1

Introduction

This thesis is devoted to the analysis of multi-server queuing models in the context of redundancy and user mobility. Redundancy is a load balancing technique recently used in computer clusters in order to improve the delay experienced by the users. In a cellular network, mobility is the phenomenon where a user communicates with a different base station when geographically moving in the network. Both systems have in common that the underlying model is a multi-server system. In this thesis, we aim to characterize the impact of redundancy and mobility in the performance.

The general notion of redundancy is to dispatch multiple copies of each job to a subset of servers and to consider the result of whichever copy completes service first, moment in which the additional copies are removed from the system. Redundancy aims to minimize the system latency by exploiting the variability in the queue lengths and server capacities of the different servers. The potential of redundancy relies in finding the right trade-off between on one hand exploiting variability and on the other hand wasting resources by adding redundant copies.

Mobility addresses the fact that users in a cellular network switch from cell to cell when geographically moving in the system. That is, upon arrival each user is first assigned to a particular base station and then, this user follows a path where it receives service from consecutive cells until it is completely served and leaves the system. Cellular mobile networks are studied due to the various empirical evidences showing that the capacity of base stations are more efficiently used under mobility.

From the mathematical point of view, both redundancy and mobility can be modeled as a multi-server system, where a set of servers (or base stations) provides service to an incoming stream of jobs. Jobs arrive according to some stochastic process and require a randomly distributed amount of service. Relevant performance measures considered in this thesis are stability of the system and the delay experienced by a user.

1.1 Redundancy models

In large computer cluster systems, load balancing has a big impact in the performance perceived by the end-users. A load balancing strategy determines how to distribute the incoming jobs among the servers. For instance, Join-Shortest-Queue strategy dispatches each incoming job into the server with least number of jobs among all the servers in the system. This strategy, which is the optimal dispatching policy under exponentially distributed service times ([74]), can be unpractical, since there is constant signaling between the dispatcher and the servers.

In order to balance the trade-off between information and performance, many different load balancing strategies have been analyzed in literature. The following strategies partially share information between the servers and the dispatcher: Under Power-of-d schemes, upon arrival of a job the dispatcher will probe d servers, and the job either joins the shortest queue (JSQ(d)) or joins the queue with smallest workload (JSW(d)). Under idle-server based policies, a server sends a notification to the dispatcher when it idles. Under pull/push mechanisms, servers that are idle can claim jobs from busy servers, and vice versa, servers with long queues can transfer jobs to idle servers. Redundancy on the other hand, is a load balancer that is unaware of the state of the servers, but does require signaling among the servers for cancellation of the copies.

The general notion of redundancy is to dispatch multiple copies of the same job to multiple servers and wait until a first copy is taken into service or completes service, moment in which the remaining copies of this job are removed. Redundancy exploits the variability of queue lengths and server capacities in the system, and thus potentially reduces the response time. Nevertheless, adding redundant copies induces a waste of resources as servers work on copies that might not end up being completely served. Therefore, there is a trade-off between exploiting the variability of the servers capacities and the additional resources that this incurs.

In spite of the non-negligible waste of resources, many empirical and numerical studies suggest that redundancy could potentially improve the performance of real-world computer system applications. They advocate that redundancy is a robust and reliable technique with promising results for a wide range of conditions. Note that, even though the dispatcher is unaware of the state of the servers, by sending redundant copies, it might avoid that jobs queue up on heavily-loaded servers.

The rest of this section is organized as follows. In Section 1.1.1 we present various studies where redundancy has been implemented in real-world computer applications. In Section 1.1.2 we present the modeling framework and terminology used in the literature, as well as in the thesis. In Section 1.1.3 we provide a survey on the stability results for redundancy models available in the literature. In Section 1.1.4 we focus on results regarding the performance analysis and in particular discuss redundancy models that

have a product-form steady-state distribution.

1.1.1 Implementation in real-world computer systems

Computer systems are clusters composed of hundreds of servers working in parallel managing thousands of arriving queries per second. A low response time is a mayor objective for computer system designers, [51]. As an example, [98] reveals that for Microsoft Bing an additional two second slowdown reduces the requested queries per user by 1.8% and the revenue per user by 4.3%. However, computer systems can have very variable response times due to e.g. sharing resources among different applications/requests, demon programs running in the background, maintenance activities and garbage collection.

In the following we present some studies that show that redundancy can improve the performance of real-world applications in the context of computer systems. These examples are DNS, Google Web Search, MapReduce and YouTube.

Domain Name System (DNS) is a hierarchical and decentralized naming system that associates domain names to their respective IP addresses so that the computer service locates and identifies the underlying network protocol. Since 1985, DNS provides worldwide extended distributed directory service, which is essential for the functionality of the Internet.

Vulimiri et al. [103, 104] consider a list of 10 DNS servers ranked with respect to their mean response time (previously computed by simulation) and Alexa.com's list of the top one million website names. The authors observe that for the system where 10 copies of each query are dispatched to all 10 DNS servers, the fraction of queries later than 500 ms is reduced by a factor 6.5, and the fraction later than 1.5 sec is reduced by a factor 50, compared to the system where queries are assigned to a single server chosen uniformly at random. Additionally, Vulimiri et al. [104] study the response times for the system where d copies of each query are dispatched to the d best-ranked DNS servers compared to dispatching the job only to the best-ranked DNS server, for $d = 2, \ldots, 10$. The authors observe that the mean, median, 95th and 99th percentile of the average reduction is about 20-30% with 2 servers and improves up to 50-62% reduction with 10 servers.

Google Web Search is by far the most used Internet search engines worldwide, with over 92% of the Internet search queries, [99]. Google's file storage system is composed of scalable distributed data-intensive applications [49] and is very sensitive to fluctuations of the response time. For example, [27] observes that when the delay is elongated by 4 seconds, these users perform 0.74% fewer searches after 4-6 weeks.

A Google Web Search query follows a sophisticated process before it presents a list of titles together with the respective uniform source locations to the user, [35, 12]. In a few words, first a DNS associates the query with an IP address and its relative cluster. Within the cluster, a search protocol matches the query with a list of documents identifiers, known as docids, ordered by their relevance. The list of docids serves to compute the title and uniform source location of these documents.

Google applies two short-term redundancy strategies to reduce the tail response time of its users, among other techniques, see [35, 12] for more details: *i*) Hedged requests; each job dispatches a second copy only when the original one fails to complete service before some amount of time, and waits until one of its copies is served. In [35] the authors observe that sending a hedging request after a 0.10 seconds delay reduces the 99.9th-percentile tail response time from 1.8 seconds to 0.74 seconds when reading values for 1000 keys stored in a BigTable distributed across 100 servers. This improvement induces sending just 2% more of request. *ii*) Tied requests; each jobs dispatches multiple copies into the system and when the first copy starts execution, the additional copies are removed from the system. The server sends a cancellation message to the additional copies in the servers. This implies that servers need to communicate among them and induce some small delay when two copies enter service simultaneously, but avoids serving the same job in multiple servers. The authors of [35] observe that when jobs dispatch two copies in the Google's clusters, the median response time is reduced by 16% and the 99.9th-percentile of the tail of the response time distribution is reduced by nearly 40%.

MapReduce is a programming model used to process and generate large data sets, where incoming jobs are partitioned into tasks and executed in parallel, see more details in [34]. MapReduce is composed by a *map* function that generates intermediate key/value pairs from the incoming data set of key/value pairs, and a *reduce* function that merges the intermediate values associated with the same key. Many real-world operations are written as a MapReduce model, such as the Hadoop production cluster at Facebook and the Dyrad cluster at Microsoft Bing.

MapReduce comes with its own challenge named stragglers. A task is said to be a straggler if it runs much slower than other tasks. Stragglers dramatically affect the delay of a job's completion, and particularly that of small jobs, i.e. jobs with a few tasks. This is an important issue in practice, since over 80% and 60% of the jobs are small with fewer than 10 tasks in Hadoop and Dyrad, respectively, [5].

Redundancy has been implemented in a few straggler migration techniques in order to improve the performance of MapReduce. Under the speculative execution technique, if a tasks is detected to be a straggler, an additional copy of that task is send to another server, [34]. The authors observe that the mean response time is significantly reduced while the utilization of the system only increases by a small percentage. Mantri and

LATE are both further improvements of the speculative executions strategy, see [6] and [106] for more details.

Ananthanarayanan et al. [4, 5] implement the Dolly technique and evaluate the performance using a trace-driven simulator replaying Hadoop logs from the Facebook cluster for both LATE and Mantri straggles migration techniques. In contrast to the straggler migration techniques, under Dolly all the tasks of a small job are replicated and send into multiple servers upon arrival. The authors observe that under Dolly, the completion time of small jobs improves by 47% and 39%, compared to LATE and Mantri, respectively, at the cost of increasing utilization by less than 4%. Furthermore, the overall average completion time of jobs improves by 40% and 33% in both cases.

YouTube, the well-known video streaming application by Google, contributes to more than 25% of total mobile application traffic share, and to over 15% of all global traffic during the worldwide lock-down pandemic situation due to COVID-19, according to The Mobile Internet Phenomena Report 2020 [91] and the The Global Internet Phenomena Report COVID-19 Spotlight May 2020 [92], respectively.

The main objective of video content providers is to deliver their users with video playback that satisfies a high Quality of Experience (QoE). YouTube's adaptation algorithm implements redundancy in the following way: When a user requests to see a video, this is segmented in small segments and coded into 4 quality levels. In the beginning of the video reproduction, lower-quality level consecutive segments are requested. Later on, whenever varying bandwidth conditions allow it, the user requests additional higher-quality level consecutive segments of the video. The adaptation algorithm replaces previously downloaded lower-quality segments with higher-quality segments, as soon as these are available (overlapping with lower-quality segments). Hence, the user requests for multiple copies of the same video segment. We note that these copies are not identical, since they are of different quality levels. The adaptation algorithm takes the copy of the segment with highest-quality level, but the additional copies are not removed from the system. Therefore, there is a trade-off between network efficiency and average playback quality that also affects the QoE of the user. See [96] for a more detailed description.

In [95, 96] the authors quantify how much redundant traffic occurs and what is the overall efficiency of this approach. Their results show that under YouTube's adaptation algorithm, the average playback quality can increase significantly by up to 0.7 quality levels by just downloading a 30% of redundant data.

1.1.2 Mathematical modeling

From the previous motivating examples, we observe that redundancy is a powerful load balancing technique that is widely implemented in order to improve the performance for various systems. We also observe that each system presents different properties regarding the inherent structure of the copies of a job. In the following we present mathematical models corresponding to different assumptions made in the literature.

The redundancy model is either flexible or constrained. Under a flexible redundancy model, a job can dispatch copies to any server in the system. Under a constrained redundancy model, *the model topology* determines the set of compatible servers of a job, that is, the set of servers where the copies of the job can be served. The latter is motivated by the fact that in real-world applications, even if servers are intrinsically similar, some servers might be better equipped to process particular jobs because of affinity relations or data locality issues.

Given a multi-server system with K servers, let us denote by $S = \{1, \dots, K\}$ the set of servers. Jobs are labeled by types $c = \{s_1, ..., s_i\} \subset S$, where i is the number of copies and c is the set of compatible servers of the job. We let $\mathcal{C}$ be the set of all types, that is, $\mathcal{C}$ determines the redundancy topology. Two constrained redundancy models that we analyze in this thesis are redundancy-d and nested models.

The redundancy-d model is a multi-server system with K homogeneous servers with capacity μ and the redundancy-d topology. Under the redundancy-d topology, each job sends a copy to d out of K servers chosen uniformly at random, see Figure 1.1 (a). That is, $\mathcal{C} := \{\{s_1, \dots, s_d\} \subset S \ : \ s_i \neq s_j, \quad \forall i \neq j\}$, with $|\mathcal{C}| = \binom{K}{d}$. The number of copies d is known as the redundancy degree.

The nested model is a multi-server system with K heterogeneous servers with the nested topology. A system is said to have a nested topology if $\mathcal{C}$ satisfies the following: for all job types $c, c' \in \mathcal{C}$, either *i)* $c \subseteq c'$ or *ii)* $c' \subseteq c$ or *iii)* $c \cap c' = \emptyset$. First of all, note that

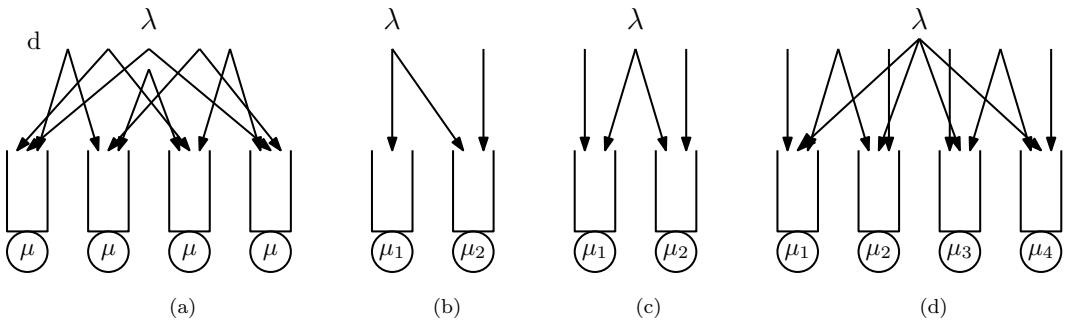


Figure 1.1 From left to right, the redundancy-d model (for $K = 4$ and $d = 2$), the N-model, the W-model and the WW-model.

the redundancy-d model does not fit in the nested topology. The smallest nested system is the so-called N-model (Figure 1.1 (b)): this is a $K = 2$ server system with types $\mathcal{C} = \{\{2\}, \{1, 2\}\}$. Another nested system is the W-model (Figure 1.1 (c)), that is, $K = 2$ servers and types $\mathcal{C} = \{\{1\}, \{2\}, \{1, 2\}\}$. In Figure 1.1 (d), a nested model with $K = 4$ servers and 7 different jobs types, $\mathcal{C} = \{\{1\}, \{2\}, \{3\}, \{4\}, \{1, 2\}, \{3, 4\}, \{1, 2, 3, 4\}\}$ is given. This model is referred to as the WW-model.

Cancellation of copies of a job might occur either as soon as the first copy of this job enters service, known as cancel-on-start (*c.o.s.*) or as soon as a copy is fully served, known as cancel-on-complete (*c.o.c.*). Both *c.o.s.* and *c.o.c.* are interesting since they exploit different aspects of the system: For instance, when FCFS is implemented, under *c.o.s.*, the job is served by the server with least workload. Whereas under *c.o.c.*, the job not only exploits the variability of the workloads in the queues, but also the variability in the server capacities.

Redundancy models are challenging to analyze due to the correlation among the departure processes of the servers induced by the cancellation of the copies. Within a server there is a departure of a copy due to the following two events: (*i*) a local copy departs due to completion, or (*ii*) a copy in another server completes (under *c.o.c.*) or starts service (under *c.o.s.*) causing the corresponding copy in the server to depart.

A scheduling policy determines how copies are served within each server. Common redundancy-unaware scheduling policies are First-Come-First-Served (FCFS) and Processor Sharing (PS), which are widely implemented in real-world computer systems ([56]), and Random-Order-of-Service (ROS), which will present a promising performance under redundancy models. Scheduling policies that try to exploit the redundant copies in the system, so-called redundancy-aware polices, are Least-Redundant-First (LRF), Most-Redundant-First (MRF) and Primaries-First (PF), introduced in [41]. For instance, under LRF (MRF) within a server jobs with fewer (larger) number of copies have priority over jobs with larger (fewer) number of copies. Under PF, among the copies of a job, there is one that is a primary copy, whereas the other copies are secondary copies. Within each server, primary copies have priority over the other copies and copies with the same priority are served according to FCFS.

The correlation structure among the copies plays an important roll when characterizing the instantaneous departure rate of a job in *c.o.c.* systems. This describes how the copies of the same job are related among them. Formally, the service times, a.k.a. service requirements, $X_1, \ldots, X_k$ of the k copies of one job can be sampled by a joint service time distribution $F(x_1, \ldots, x_k)$.

A common assumption is that all the copies of a job are *i.i.d. copies*, that is, all copies sample i.i.d. service times from the service time distribution of the job. In that case there is no correlation among the copies of the same job. If in addition, the service times are exponentially distributed, this considerably simplifies the analysis, since the instantaneous departure rate of a job is given by the sum of the rates of the copies that are in service. As we will see later in Sections 1.1.3 and 1.1.4, many studies on redundancy models have considered i.i.d. copies.

Another common assumption is that copies of a job are *identical copies*, that is, all the copies of a job have the same service time. Thus, there is a full correlation among the copies of each job. The instantaneous departure rate of a job is given by the departure rate of the copy that has received most service, which complicates the analysis.

We further refer to the S&X model, which includes both i.i.d. copies and identical copies. Here, the service time of each copy is decomposed into two components; the inherent job size, which is identical for all the copies of a job, and the experienced slowdown on the server it is being served. This model, first introduced in [42], was inspired by the different response times that the same job can experience in different servers due to server intrinsic characteristics.

We note that the correlation structure among the copies will considerably affect both the stability region and the performance of the system. The i.i.d. copies assumption can be considered when the intrinsic properties of the servers are highly variable due to external conditions, such as garbage collection. The latter can provoke that the service time experienced by each copy can be considerably different at all the compatible servers of the job. On the other hand, when the intrinsic characteristics of the servers are similar, the identical copies assumptions might be preferred, since all the copies of a job will experience similar service times in all the compatible servers of the job. However, we note that the i.i.d. and identical copies assumptions are the extreme cases of the generally correlated copies, where under i.i.d. copies these are completely decorrelated and under identical copies the correlation is 1.

1.1.3 Literature overview of stability results

In the following, we survey the main stability results for *c.o.c.* redundancy models available in the literature, including those presented in this manuscript. This section is based on [SR4], where we provide a survey on the stability condition results and open problems for redundancy models.

From the point of view of stability, under the *c.o.s.* redundancy model, there is a single copy of each job that enters service, as opposed to the redundancy *c.o.c.* models. Hence, there is no waste of computational resources, which implies that redundancy

has no direct impact in the stability condition, see [10] for more details. Therefore, we focus only on *c.o.c.* redundancy models.

Table 1.1 summarizes the main stability results for *c.o.c.* redundancy models available in the literature and proved in this thesis. The table is organized by scheduling policy, service time distribution, redundancy topology and correlation structure. In brackets we specify the additional assumptions that the authors consider in their respective paper. The term "red-d" refers to the redundancy-d model and the term "gen" refers to a general redundancy topology.

Table 1.1 Stability results for *c.o.c.* redundancy models.

	Service time	**i.i.d. copies**		**identical copies**		**General correlation**	
		red-d	**gen**	**red-d**	**gen**	**red-d**	**gen**
FCFS	**Exp.**	[43]	[18, 46]	**Sec 3.3**			
	General	[82](Scaled Bernoulli) [60](Mean field)		[59](Mean field)		[77] (Sufficient condition)	[84] (Comparison result)
PS	**Exp.**	**Sec 3.2**	**Sec 8.1.1** (Conjecture)	**Sec 3.4**			
	General			[83]	**Sec 4.2**	[83] (Necessary condition)	
ROS	**Exp.**	**Sec 3.2**	**Sec 8.1.1** (Conjecture)	**Sec 3.5**			**Sec 8.1.3** (Conjecture)
	General						

We first introduce the stability condition when jobs have i.i.d. copies (first two columns of Table 1.1). Secondly, we discussed the stability condition of systems with identical copies (middle two columns in Table 1.1) and lastly for systems with a general correlation structures among the copies (last two columns in Table 1.1). For the stability results that are proved in this thesis, we provide the section where this is done. The results that are presented in Chapter 8 are still open problems for which we state a conjecture regarding the stability condition.

1.1.3.1 Independent and identically distributed copies

In this section we assume that jobs have i.i.d. copies, first for exponentially distributed service times and then for scaled Bernoulli service times.

Exponential service times

We first discuss results on FCFS and exponentially distributed service times, a setting studied by Bonald and Comte [18] and Gardner et al. [43, 46]. It was shown in [18] that

this model fits the framework of order-independent queues and that the steady-state distribution is of product-form (as will be further discussed in Section 2.4.2). The latter facilitates the derivation of performance measures such as the stability condition and the mean response time. The proposition below states the stability result for this model.

Proposition 1.1.1 ([18, 46])**.** *For a redundancy system with general topology under FCFS with exponentially distributed service times and i.i.d. copies, the system is stable if for all $C \subseteq \mathcal{C}$,*

$$\lambda \sum_{c \in C} p_c < \sum_{s \in S(C)} \mu_s, \tag{1.1}$$

where $S(C) = \bigcup_{c \in C} \{s \in c\}$. The system is unstable if there exists $\tilde{C} \subseteq \mathcal{C}$ such that

$$\lambda \sum_{c \in \tilde{C}} p_c > \sum_{s \in S(\tilde{C})} \mu_s.$$

Informally, Eq. (1.1) states that the arrival rate to any subset of job types must be less than the total capacity of the associated compatible servers. For exponential service times, we note that if Eq. (1.1) is not satisfied, then there does not exists a policy that makes the system stable. Hence, we say that the system is *maximally stable*. Thus, we conclude that the stability region is not reduced due to adding redundant copies. The latter might seem counter-intuitive at first, since even if servers waste resources serving copies that are not fully served, the stability condition is as large as if there was no redundancy.

Extending Proposition 1.1.1 to other scheduling policies is an important open problem (see Section 8.1.1 for more details) and it has only been achieved for the redundancy-d model. In this case, it is easy to see that Eq. (1.1) reduces to $\lambda < \mu K$. In one of the main results of this thesis, we show that this stability condition remains valid when either PS or ROS is implemented.

Proposition 1.1.2 (Section 3.2)**.** *For the redundancy-d model under either PS or ROS with exponentially distributed service times and i.i.d. copies, the system is stable when $\lambda < K\mu$ and unstable when $\lambda > K\mu$.*

Hence, under PS, ROS and FCFS, the redundancy-d model is maximally stable. This however does not hold true in general. In Section 3.2.2, we provide an example of a priority policy that is not maximally stable, i.e., the system becomes unstable even though $\lambda < K\mu$.

General service times

To the best of our knowledge, no stability results exist for general service times with i.i.d. copies. We present here the stability result obtained for scaled Bernoulli service times, defined as

$$\begin{cases} X \cdot M, & \text{w.p. } 1/M \\ 0, & \text{w.p. } 1 - 1/M, \end{cases}$$

where $M > 0$ and X is a strictly positive random variable with $E[X] = 1$. In this setting, Raaijmakers et al. [82] characterize the stability condition for the redundancy-d model where FCFS is implemented and the number of servers grows large.

Proposition 1.1.3 ([82])**.** *Consider the redundancy-d model under FCFS with scaled Bernoulli service times and i.i.d. copies. Then,* $\lambda < \frac{M^{d-1}}{E[\min(X_1,\ldots,X_d)]}$ *is a sufficient stability condition for any* M*. In addition, for any* ϵ*, it holds that* $(1-\epsilon)\lambda < \frac{M^{d-1}}{E[\min(X_1,\ldots,X_d)]}$ *is a necessary condition, for* M *sufficiently large.*

We observe that the stability condition is independent of the number of servers, but strongly depends on the number of copies d. The latter is in contrast to the exponentially distributed service times, where the stability condition does dependent on the number of servers but is independent of d (see Proposition 1.1.2). Thus, we observe that when copies are i.i.d., the stability condition strongly depends on the service time distribution. In addition, we observe that as M grows large (and hence the variance of the service times grows large), the stability region increases by a factor M^{d-1}, by taking advantage of a greater diversity in service times.

Several studies (e.g., [103]) have shown that the i.i.d. copies assumption can be unrealistic, since large jobs remain large when replicated. Hence, having additional copies could lead to high response times and even instability. Motivated by the latter, stability results with correlated copies have been the focus of recent literature. In the following, we first introduce the stability results under identical copies and then, under generally correlated copies.

1.1.3.2 Identical copies

Let us assume now that jobs have identical copies, i.e., all copies belonging to one job have the same size. This correlation makes that a job can only depart due to its copy that has received most service so far. Thus, the instantaneous departure rate of a job depends on its copy that has currently attained most service.

FCFS policy. With FCFS, the eldest job in the system will be served at all of its compatible servers. A job later in the queue will be served at its compatible servers that are not engaged by earlier jobs in the queue.

The stability condition for the redundancy-d system with FCFS and exponentially distributed service times is characterized in Section 3.3, through the average departure rate per type in the so-called *saturated system*. The latter assumes an infinite backlog of jobs waiting for service. The long-run time-average number of jobs in service in the saturated system is denoted by $\bar{\ell}$. A detailed description of the saturated system and the characterization of $\bar{\ell}$ can be found in Section 3.3.

Proposition 1.1.4 (Section 3.3)**.** *For the redundancy-d system under FCFS with exponentially distributed service times and identical copies, the system is stable if* $\lambda < \bar{\ell}\mu$ *and unstable if* $\lambda > \bar{\ell}\mu$.

The value of $\bar{\ell}$, and hence the stability region, can be numerically obtained by solving the balance equations of the saturated system, see Section 3.3 for more details. We note that the instantaneous departure rate in the saturated system strongly depends on the types in service. As a consequence, no expression has been derived so far for $\bar{\ell}$ for general K and d values.

Hellemans et al. [59] analyze the present redundancy-d model under the mean field regime, that is, when the number of servers tends to infinity. The authors provide a numerical method to see whether a given system is stable, but do not provide any characterization of the stability region.

PS policy. Under PS and identical copies, the stability condition is characterized in Section 4.2. There it is shown that the stability condition coincides with that of a K parallel server system where each type-c job is only dispatched to its so-called least-loaded servers. In order to state this result, we first need to define several sets of servers and job types. The first subsystem includes all servers, that is $S_1 = S$. We let $\mathcal{C}_1 = \mathcal{C}$. We denote by $\mathcal{L}_1$ the set of least-loaded servers in the system $S_1 = S$. Thus,

$$\mathcal{L}_1 = \left\{ s \in S_1 \ : \ s = \arg\min_{\tilde{s} \in S_1} \left\{ \frac{1}{\mu_{\tilde{s}}} \sum_{c \in \mathcal{C}(\tilde{s})} p_c \right\} \right\}.$$

For $i = 2, \ldots, K$, we define recursively

$$\begin{aligned} S_i &:= S \backslash \cup_{l=1}^{i-1} \mathcal{L}_l, \\ \mathcal{C}_i &:= \{c \in \mathcal{C} \ : \ c \subset S_i\}, \\ \mathcal{C}_i(s) &:= \mathcal{C}_i \cap \mathcal{C}(s), \\ \mathcal{L}_i &:= \left\{ s \in S_i \ : \ s = \arg\min_{\tilde{s} \in S_i} \left\{ \tfrac{1}{\mu_{\tilde{s}}} \textstyle\sum_{c \in \mathcal{C}_i(\tilde{s})} p_c \right\} \right\}. \end{aligned}$$

The S_i-subsystem refers to the system consisting of the servers in S_i, with only jobs of types in the set $\mathcal{C}_i$. The $\mathcal{C}_i(s)$ is the subset of types that are served in server s in the S_i-subsystem. The set $\mathcal{L}_i$ represents the set of least-loaded servers in the S_i-subsystem.

Finally, we denote by $i^* := \arg\max_{i=1,\dots,K}\{\mathcal{C}_i \ : \ \mathcal{C}_i \neq \emptyset\}$ the last index i for which the subsystem S_i is not empty of job types.

The stability condition is now characterized in Section 4.2 by the least-loaded servers that can serve each job type.

Proposition 1.1.5 (Section 4.2)**.** *Assume that the service time distribution has no atoms and is light-tailed (in the sense of Eq. (4.1)). For a redundancy system with a general topology under PS with identical copies, the system is stable if* $\lambda \sum_{c\in C_i(s)} p_c < \mu_s$*, for all* $s \in \mathcal{L}_i$*,* $i = 1, \dots, i^*$*. The redundancy system is unstable if there exists* $\iota \leq i^*$ *and* $s \in \mathcal{L}_\iota$ *such that* $\lambda \sum_{c\in C_\iota(s)} p_c > \mu_s$*.*

For the redundancy-d model, the above stability condition simplifies into $\lambda < K\mu/d$, proven in Section 3.4. The latter coincides with the stability condition of a system where all the copies need to be served, that is, the worst possible stability condition.

ROS policy. Under ROS and identical copies, the stability condition is characterized in Section 3.5 for the redundancy-d model with exponential service times. We show that this is given by the maximum stability condition, that is, $\lambda < \mu K$. We note that this is also the stability condition when copies are i.i.d., Proposition 1.1.2. We believe that this holds true for any redundancy structure and any correlation structure, as stated in Section 8.1.1.

Proposition 1.1.6 (Section 3.5)**.** *For the redundancy-d model under ROS with exponentially distributed service times and identical copies, the system is stable if* $\lambda < K\mu$*.*

The intuition behind the above result is as follows. Whenever there are many jobs in a server, the probability that this server serves a copy of a job that has also a copy elsewhere in service will be close to zero. Hence, with a probability close to 1, all highly-loaded servers are serving copies of different jobs and their instantaneous departure rate equals the sum of their capacities.

1.1.3.3 Generally correlated copies

We consider now redundancy models where the service times of the copies of each job are correlated according to some general structure.

FCFS policy. For FCFS, Raaijmakers et al. [84] consider a general workload model, which subsumes the S&X model, introduced in [42]. The main difference is that in [84] the server capacities are not fixed, but each job samples server capacities from a discrete and finite distribution. The authors assume that the server speed variations (slowdowns) are either distributed according to New-Better-than-Used (NBU) or New-Worse-than-Used (NWU). See Section 2.1 for more details on NBU and NWU distributions.

Depending on the random variation in the server speed, the authors prove that either no replication ($d = 1$) or full replication ($d = K$) provides a larger stability region. Note that here the stability region refers to a wider concept than what we considered before. That is, it refers to the set of arrival rates such that there exists a static assignment rule that makes the system stable.

Proposition 1.1.7 ([84]). *Consider the following model. Each job is routed to d servers according to some static probabilistic assignment. Servers implement FCFS. Every time a server starts serving a new copy, it samples a speed variation, which is independent across servers. The type of a job is determined by the capacities it would obtain in each server. A job has a generally distributed service time.*

- *If the probabilistic assignment can depend on the job type, and the speed variation follows an NBU distribution, then the stability region for $d = 1$ is larger or equal than that for $d > 1$.*
- *If the probabilistic assignment does not depend on the job type, and the speed variation follows an NWU distribution, then the stability region for $d = K$ is larger or equal than that for $d = 1$.*

From the above we observe that the optimal redundancy degree does not depend on the job size distributions, but rather on the random variation in the server speeds for a given job among the servers.

A sufficient stability condition for the redundancy-d model with FCFS has been obtained in Mendelson [77]. He considers that the service times of the copies $X_1, \ldots, X_d$ are identically distributed with mean 1 and sampled from a joint distribution $F(x_1, \ldots, x_d)$.

Proposition 1.1.8 ([77]). *Consider the redundancy-d model where FCFS is implemented and the service times of the copies are sampled from a general joint distribution $F(x_1, \ldots, x_d)$. Then, $\lambda < \lambda_{lb}$ is a sufficient stability condition, where*

$$\lambda_{lb} := \frac{\mu K}{\sum_{m=0}^{d} \left(\sum_{j=1}^{d-m} E[\min(X_1, \ldots, X_j)] + mE[\min(X_1, \ldots, X_d)] \right) P_m},$$

and $P_m = \binom{K-d}{d-m}\binom{d}{m}/\binom{K}{d}$.

For the special cases $d = 1$ and $d = K$, the sufficient condition simplifies to $\lambda < \lambda_{lb} = K\mu$ and $\lambda < \lambda_{lb} = \mu / E[\min(X_1, \ldots, X_d)]$, respectively, which are in fact also the necessary stability conditions.

PS policy. We now consider the redundancy-d model where PS is implemented. Raaijmakers et al. [83] characterize the stability condition under any service time

distribution through the minimum of the service times of the copies of a job. The latter can be heuristically explained as follows: assume that all servers are equally loaded. Then, due to PS, the copy that completes first is the one with the smallest service time among all copies of the job.

Proposition 1.1.9 ([83])**.** *For the redundancy-d model under PS where the service times of the copies are sampled from a general joint distribution $F(x_1, \ldots, x_d)$, a necessary stability condition is $\lambda d E[\min(X_1, \ldots, X_d)] < K\mu$.*

In the particular case where copies are identical, the authors in [83] prove that Proposition 1.1.9 gives a sufficient and necessary stability condition, which is given by $\lambda d < K\mu$. We note that the latter coincides with the stability condition for light-tailed service times distributions provided in Proposition 1.1.5. Moreover, [83] shows that the stability condition under NWU service time distributions, respectively NBU service time distributions, is larger, respectively smaller, than that for exponential service times.

1.1.4 Literature overview on performance evaluation

Besides stability results, other performance measures have been studied in the literature, such as the steady-state distribution, the workload distribution, as well as the response times. These results are discussed in this section.

1.1.4.1 Performance of redundancy models

In some cases, it has been shown that the steady-state distribution of redundancy models has a product-form distribution. In Section 2.4.1 we overview the class of systems with a product-form steady-state distribution and introduce two particular systems subsumed in this class of systems, the order-independent queues and the multi-type jobs and multi-types server system in Visschers et al. [102]. In the following, we survey the most relevant results for redundancy models with a product-form distribution.

c.o.c. and i.i.d. copies

For the *c.o.c.* redundancy models, the first performance results where obtained for FCFS, exponentially distributed service times and i.i.d. copies. Under this setting Bonald and Comte [18] and Gardner et al. [43, 46] characterize the steady-state distribution, which is of a product-form, by showing that the system is an order-independent queue. It is also shown in [43] that for the redundancy-d model, the mean delay reduces as the redundancy degree d increases.

Cardinaels et al. [28] characterize the stationary behavior of the response time in the heavy-traffic regime, that is, as the total load in the system approaches 1. Due to the product-form of the steady-state distribution, the authors show that the joint

distribution of the numbers of jobs of the various types converges to an exponentially distributed random variable times the vectors of the probability types, a phenomenon known as state space collapse.

Adan et al. [2] consider exponential services times and compare the redundancy model under FCFS with i.i.d. copies to two multi-type multi-server systems without redundancy. These are the FCFS-ALIS parallel queuing model and the FCFS matching model, which both have a product-form steady-state distributions. Under the FCFS-ALIS model, incoming jobs join a central queue in order of arrival and wait until a compatible server idles. If upon arrival, the job finds idle compatibles servers, it joins the one that has been idle for longest time, first introduced in Adan et al. [1]. Under the FCFS matching model, incoming jobs join a central queue in order of arrival. Incoming servers scan the queue of jobs and match with the longest waiting job that it can serve, after which both the server and the matched job leave the system. If an incoming server does not find a match, it leaves immediately without a match, [2]. The authors show that the response times of jobs coincide under the redundancy model and the FCFS matching model. The comparison relation between the redundancy model and the FCFS-ALIS model depends on the parameters of the system, such as service rates and server capacities.

c.o.s. and related models

The *c.o.s.* redundancy model where FCFS is implemented and jobs have exponentially distributed service times is analyzed in Ayesta et al. [10], where the authors prove that the model is equivalent to the Join-the-Shortest-Workload (JSW) model. The key observation made by the authors was to relate both *c.o.s.* and *c.o.c.* redundancy models to the state aggregation approach presented in Visschers et al. [102], which is known to have a product-form steady-state distribution, see Section 2.4.1.

Ayesta et al. [9] present a token-based central queue approach for the parallel FCFS server system where incoming jobs are associated to a set of compatible tokens. The authors prove that the steady-state distribution of the token-based system has a product-form. Furthermore, the authors show that the order-independent queues, the FCFS matching model, and the state-aggregation approach in Visschers et al. [102] are subsumed by this approach.

Furthermore, in a recent paper, Gardner and Righter [45] present a survey on models with a product-form steady-state distribution. The authors consider models with arbitrary compatibility graphs between servers and jobs, where FCFS is implemented and jobs have exponentially distributed sizes. The authors provide a single-server queue approach that subsumes the order-independent queues, and retrieve known product-form distributions, such as the *c.o.c.* and *c.o.s.* redundancy models presented above.

Comte and Dorsman [29] introduce the Pass-and-Swap queues, and prove that these are a generalization of the order-independent queues, and that the product-form of the steady-state distribution is preserved. Under Pass-and-Swap queues, an undirected graph characterizes the swapping relation between job types. Upon a job completion, a chain reaction is triggered, where starting from the job that departed, each ejected job, swaps position with the closest compatible job type in the queue. The authors provide several queues that fall into this framework, such as various closed networks, among which there is a loss variant of the *c.o.s.* redundancy model. The authors show sufficient and necessary stability conditions for Pass-and-Swap queues that also hold for order-independent queues.

Limiting regimes

For *c.o.c.* redundancy models with identical copies, no product-form result has been proved for the steady-state distribution. Instead, researchers have focused on analyzing this model in the mean-field regime, that is, when the number of servers tends to infinity.

Hellemans and van Houdt [59] analyze the redundancy-d model with FCFS and identical copies in the mean-field regime, that is, when the number of servers tends to infinity. The authors develop a numerical method to compute the workload and response time distributions under general service time distributions. In order for this method to work, the authors assume the parameters to be such that the system is stable. The authors can numerically infer whether the system is stable, but do not provide any characterization of the stability region. The authors extend the previous results to i.i.d. copies in [60]. In Hellemans et al. [57], the authors extend this study to include many other load balancing strategies such as JSQ(d) and JSW(d), and general correlation structure among the copies.

Several studies consider the homogeneous server system under JSQ(d) where FCFS is implemented in the mean field, that is, when the number of servers tends to infinity. In Mitzenmacher [80], the author characterizes the mean response time of the limiting system for exponentially distributed service times. Hellemans and van Houdt [58], characterize the workload and response time distribution of the present system where jobs have general service times. In Hellemans and van Houdt [61] the authors propose a method to compute the limit of the mean response time when additionally the load approaches 1, i.e., the heavy traffic regime. This method is valid for models that satisfy certain conditions, which include JSQ(d) and JSW(d).

Atar et al. [8] consider the heterogeneous FCFS server system and introduce the Replicate-to-Shortest-Queue scheduling policy, RSQ(d), where each job is dispatched to the d servers with least number of jobs, and whenever a first copy enters service, the additional $d-1$ copies waiting in the queue are removed, that is the *c.o.s.* model. We

note that when $d = 1$, RSQ(1) coincides with the JSQ model and when $d = N$, RSQ(N) coincides with the JSW model. The authors show that in the heavy-traffic regime, a state state collapse occurs, that is, in the limit the queue lengths of all servers coincide.

1.1.4.2 Impact of the scheduling policy

The following studies focus on proposing redundancy implementations/strategies that not only improve the overall mean response time, i.e., are efficient, but also do not harm the marginal mean response time for job types, i.e., are fair.

Some of the earliest studies on the performance of redundancy models are Kim et al. [68] and Koole and Righter [69], where the authors consider the flexible redundancy system where FCFS is implemented and jobs have i.i.d. copies. The authors show that for NWU service time distributions, full replication minimizes the load in the system and stochastically maximizes the number of completed tasks, respectively in [68] and [69]. In contrast, when service times follow NBU distributions, no-replication minimizes the total load in the system ([68]) and maximizes the number of completed tasks for a two server system ([69]). Borst et al. [19] consider instead a discrete-time flexible redundancy system where jobs have geometrically distributed service times and observe that redundancy harms the mean response time of the system.

Akgun et al. [3] consider a two-server system in which each server has dedicated traffic, that is, each server is a unique compatible server for one job type. The authors consider the DCF (Dedicated-Customers-First) scheduling policy and analyze the efficiency and fairness for both dedicated and redundant jobs.

Sun et al. [100] consider various low-complexity redundancy scheduling techniques for systems where job have i.i.d. copies, and investigate when these are delay-optimal (or nearly-delay optimal) with respect to the stochastic ordering. These new scheduling techniques are based on job replication and job cancellation decision features. For instance, the authors propose show that the *fewest unassigned task first with low-priority replication* and *earliest due date first with replication* policies are nearly delay-optimal with NBU and NWU distributions, respectively.

Gardner et al. [41, 44] investigate the impact that the implemented scheduling policy has on the performance for nested redundancy models (see Section 1.1.2.) where jobs have exponentially distributed service times and i.i.d. copies. In [41], the authors prove that implementing FCFS in the servers is highly effective in reducing the mean response time in the system, although LRF is optimal. However, LRF fails to be fair for redundant jobs. Thus, the authors present PF, which is fair for redundant jobs and minimizes the overall mean response time. In [41], the authors study the marginal effects of adding more redundancy. In particular for the system under LRF, the authors show that even if scheduling more redundant jobs improves the mean response time,

the maximum gains come from adding only a small proportion of redundant jobs.

1.1.4.3 Coding theory and redundancy

There has been a surge of papers from the coding theory community where redundancy is implemented. Coding is implemented in large data storing systems where reliability and fast downloads are primordial, see [89] for more details. Under the (n, k) maximum separable distance (MDS) code, each data content is encoded into n blocks and stored in n disks in such way that k out of n blocks are sufficient to reconstruct the entire file, see [64, 65] for instance. Note that the (n, k) fork-join system can be seen as the redundancy model where the first k out of n copies are requested, further when $k = 1$, the model reduces to the redundancy model with i.i.d. copies with $d = n$.

In Lee et al. [72], the authors consider the (n, k) fork-join system and give upper and lower bound system sequences that converge to the original system. Through these bounds, the authors characterize the mean response time of the system. Furthermore, Li et al. [75] study the mean field-limit and conclude that coding improves the mean response time compared to the redundancy model, i.e., $(n, 1)$.

In Joshi et al. [66], the authors consider an n server system where FCFS is implemented the (n, r, k) partial fork-join model, that is, a job is sent to r out of n servers uniformly chosen at random and waits for the first $k \leq r$ jobs to complete. The authors analyze effective replication strategies for various scenarios and show that both latency and cost are minimized when r increases for log-convex (high variable) service time distributions.

Duffy et al. [36] compare the tail response time of the (n, r, k) partial fork-join model with that of the redundancy-d model where there are r batch arrivals and a single copy of each job is requested. The authors show that the tail distribution of the response time under (n, r, k) partial fork-join is smaller than that under the redundancy-d model if $r - k \geq d$ and as the number of servers tend to infinity.

In a recent paper, Zubeldia [107] considers the $S\&X$ model where the slowdown experienced by each copy in service is independent across servers, but not necessarily independent form the job's service time. The author provides a lower-bound on the mean delay for the (n, r, k) partial fork-join system, and shows that when slowdowns are exponentially distributed and independent of the service time of the job, in the mean-field the expected delay is minimized for a constant r that only depends on the arrival rate and mean slowdown. Furthermore, when the slowdown depends in some particular way on the service time of the job, the author shows that the mean delay is minimized when smaller tasks are replicated more than larger tasks.

1.2 Mobility models

By mobility we refer to the phenomenon arising in cellular networks, in which users receive service from different stations as they move across the networks. A cellular network is defined by static base stations, a.k.a antennas (servers), which divide the fields into cells, and base stations communicate with its clients through radio communications. Mobility addresses the fact that an incoming user will start being served in some particular station, and then follow a path among the different stations in the network while receiving service. Mobility is naturally studied through multi-server models, where base stations correspond to the servers.

The mobility phenomenon also arises in ad-hoc networks. These are wireless networks, where a set of nodes behaves in a decentralized manner and with no preexisting infrastructure. That is, data transfers occur among different nodes, rather than with a common base stations, and each node decides dynamically when to transmit and to which of the nodes in the network. Under mobility, nodes transmit the data to its destination via intermediate nodes. Various studies, such as [54], show that mobility improves the capacity of ad-hoc networks, giving rise to other modeling frameworks not covered in this thesis.

1.2.1 Cellular mobile networks

In this thesis, we focus on mobile cellular networks. As we have mentioned before, each user is attached to the most suitable cell according to its geographic position in the network, and switches cell when geographically moving in the network. These users are classified according to their mobility patterns. A mobility pattern presents particular radio communication features: pedestrians in the street using smart phones move at slow speed and walk among obstacles that obstruct the signal. Vehicles such as car and trains move at high speeds. Particularly railway traffic presents a challenging scenario since trains travel in high speed, face various obstacles and transport groups of passengers for which the relative mobility speed reduces to zero. Other mobility patterns are maritime, aerial and outer space mobility, see [93, 13] for more details.

Relevant performance measures in cellular networks are users' perceived throughput, delay and loss rates. The performance in a cellular network is heavily dependent on the properties of the radio channel, such as attenuations, interferences and fading effects, which is handled by the so-called cellular handover protocol.

1.2.2 Mathematical modeling

There are several approaches studied in literature that mathematically model mobility in cellular networks. The following are the most common mobility models, see [93, 13]

for more details.

The Random-Walk Markovian model: The network is modeled by a multi-server system where each user stays in a server or moves to another server according to some transition probability distribution. Since this model is Markovian, there is no notion of path, or consecutive visits. This model permits to capture the individual movements with respect to the cells, and is the model we consider in this thesis.

The trace-based model: Cell phone companies record hundreds of mobility traces of their users in order to construct mobility patterns and use them as a source of evaluation when improving their handover protocols. This data is valuable for cell phone network companies, since poor performance of their network can cause loss of clients.

Fluid flow mobility models: Individual cells are modeled on a macroscopic level, compared to the mobility of users within the network. Thus, the behavior of the traffic of its users is similar to a fluid flow through a pipe.

1.2.3 Overview of literature

Firstly, we provide an overview of papers characterizing the capacity region and stability conditions of cellular networks. The capacity region is defined as the set of throughputs that the network can achieve, whereas the stability condition is defined as the set of parameter values such as the steady-state of the network exists.

Various scheduling strategies have been implemented in order to improve the throughput of the elastic data users in cellular networks, data transfers of elastic users have no dead-line constraints. For instance, under channel-aware scheduling strategies, the feasible service rate of each user depends on the entire active users population in a fairly intricate fashion. Various authors aim to implementing scheduling strategies that exploit the available service rate variations to improve the throughput performance. However, due to the variation rate dependencies, deriving the stability region, as well as the relevant performance measures of these models is challenging.

Bonald et al. [15, 17], Borst [20], Borst et al. [22] and Jonckheere [63], among others authors, study the capacity region and the flow-level performance of cellular networks under channel-aware scheduling, assuming that there is no mobility.

As for cellular mobile networks, Bonald et al. [16, 17], Borst et al. [21] and Borst [24] consider a wireless system carrying traffic from different classes under various channel-aware scheduling strategies. A summary of those results can be found in Borst [23]. Intra-cell (Inter-cell) mobility concerns user mobility within the base stations in the same cell (among different cells), and manifest itself as slow fading (fast fading). Whereas, mobility is modulated by introducing a speed parameter in this set of feasible service rate vectors.

Bonald et al. [16, 17] consider a single-cell scenario under intra-cell mobility where

fair share policy is implemented within each base station. That is, the feasible transmission rate is equally distributed among the active flows at each base station. The authors prove that the mean response time is lower (upper) bounded by the system with speed infinity (zero), which implies that mobility always improves the performance of the system.

Borst et al. [21, 17] consider a multi-cell system under inter-cell and intra-cell mobility. The authors characterize the stability condition for the global optimal scheduling policies as well as the fair share policy, which is smaller than under the global optimal scheduling policy, and observe that mobility tends to increase the capacity of the system.

For the above system, Borst et al. [24, 17] assume the α-*fair* scheduling policy (for $\alpha > 0$), which is designed to maximize the total server utility, in order to analyze the trade-off between the throughput and fairness in the system. The authors prove that under mobility the capacity region increases monotonically as the value of α decreases, in contrast to the system with no mobility, where the capacity region does not depend on α. Additionally, the authors observe that for a given α, the capacity of the system is larger for the system with mobility than without mobility.

Simatos et al. [97] consider a different system, this is a multi-server system under PS where users' mobility is modeled by an irreducible random walk with exponential rates that are independent across clients. Jobs arrive according to a Poisson arrival process and have exponentially distributed service times. The authors prove through fluid-limit approximations and martingales that the stability condition is given by the total arrival rate to be smaller than the total throughput rate. That is, the system is as stable as an $M/M/K$ system. Borst and Simatos [25] analyze the heavy-traffic regime of the previous model and derive a form of spatial state space collapse.

Ganesh et al. in [40] prove that for the previous multi-server system under PS where mobility is modulated by an irreducible Markov chain, the stability condition is characterized by the following: the total arrival rate to the system needs to be smaller than the total capacity of the system.

1.3 Main contributions

In this thesis, we consider a K parallel multi-server system where jobs arrive according to a Poisson process of rate λ. We analyze the impact of redundancy and mobility within this model. In Chapters 3 through 6, we focus on redundancy models and in Chapter 7 we focus on mobility models.

In Chapter 3, we investigate the redundancy-d model with K homogeneous servers with capacity μ where jobs have exponentially distributed sizes with unit mean and either i.i.d. copies or identical copies. Let us denote by $\rho := \lambda/\mu K$ the total load in the system. We note that in the absence of redundancy, the sufficient and necessary stability

condition is $\rho < 1$. We then show that under exponential service times, the stability region can reduce but can not increase. We prove that when jobs have i.i.d. copies, the stability condition is not reduced for both PS and ROS service policies, as in the case of FCFS [43].

In the case of identical copies, we prove that the stability condition strongly depends on the scheduling policy employed in the servers. For the ROS scheduling policy, redundancy does not impact the stability region of the system, that is $\rho < 1$ is the stability condition, whereas for FCFS and PS that is considerably reduced. The stability condition for FCFS is given by $\rho < \bar{\ell}/K$, where $\bar{\ell}$ denotes the mean number of jobs in an associated saturated system. Hence, $\bar{\ell}/K < 1$ and we show that it reduces as the number of copies d increases. Under PS, the stability condition reduces to $\rho < 1/d$, and coincides with the stability condition of a system where all the d copies need to be fully served. Hence, this represents the worst possible reduction in the system's stability region. Chapter 3 in this thesis is based on [SR1].

In Chapter 4 we generalize the previous results to a general redundancy topology with heterogeneous server capacities and identical copies. When PS is implemented, we characterize the stability condition for the system by a recursive algorithm that depends on the load in each server. We show that this condition coincides with that of a system where each job only dispatches copies into its least-loaded compatible servers.

In Chapter 5 we compare the stability condition and mean response times under redundancy scheduling to those under Bernoulli routing system and JSQ (among the compatible servers). Under Bernoulli routing, a single copy is dispatched to a server chosen uniformly at random among the compatible servers. We show that if the server's load are sufficiently heterogeneous, the stability region under redundancy can be much larger than that under Bernoulli routing. Which implies that the mean response time improves considerably under redundancy. We also compare, the mean response time under redundancy scheduling to that under JSQ. We observe that JSQ outperforms redundancy most of the time, except when the system is highly heterogeneous. However, under JSQ the dispatcher uses full information of the state of the system, which is often unpractical. Chapters 4 and 5 in this thesis are based on [SR3].

In Chapter 6 we investigate scheduling policies that improve the mean response time of redundancy systems with a general topology where jobs have general service times, and either i.i.d. copies or identical copies. We focus on redundancy-aware scheduling policies that are composed of two levels Π_1-Π_2, as well as redundancy unaware scheduling policies FCFS and ROS. Under policy Π_1-Π_2, policy Π_1 (the first-level policy) determines the priority among the job types and, Π_2 (the second-level policy) determines the policy among the jobs with the same priority. When jobs have i.i.d. copies and exponential service times, we generalize the result in [41] and show that for nested models LRF-Π_2 minimizes the mean response time not only with FCFS in the second level, as shown in

[41], but also with any second-level non-idling policy Π_2. We also show that for NWU service times policy Π_1-FCFS minimizes the mean response time among all policies of type $\Pi_1 - \Pi_2$, for any first-level strict priority policy Π_1.

When identical copies are assumed, we prove that MRF-Π_2, and particularly MRF-ROS, outperforms MRF-FCFS for the nested redundancy model with heterogeneous server capacities and any service time distributions, with Π_2 non-idling. The latter is due to the fact that having identical copies induces a waste of resources when serving copies of the same job. Hence, by maximizing the number of different jobs in service the mean number of jobs in the system is optimized. Chapter 6 in this thesis is based on [SR6].

Chapter 7 is devoted to the analysis of the impact of mobility in a cellular network. We consider the same model as the one analyzed in [40] and introduce α the parameter that controls the mobility speed. We investigate the impact that the mobility speed has on the performance of the system. When $\alpha = \infty$, that is, when the mobility speed is fast, we characterize the steady-state distribution and show that this is decomposed into two independent components: the total number of jobs in the system, which is an $M/M/1$ system, and the distribution of these jobs among the servers, which is distributed according to the multinomial distribution.

We then investigate the impact of α on the mean response time. We characterize various conditions such that the mean response time improves or deteriorates as mobility, i.e., α, increases under low loads. Under moderate loads, numerical analysis shows that mobility might have a negative effect on the performance of the system, in contrast to the results in some recent studies, such as [17]. However, under high loads we observe that mobility considerably improves the mean response time and is the lowest when $\alpha = \infty$. Chapter 7 of this thesis is based on [SR5].

We provide the main conclusion of this thesis in Chapter 8. There we also discuss the open problems related to stability condition and impact of the scheduling policy on redundancy models, we as well state several conjectures that are based on our intuitive understanding of the system. Section 8.1 in this thesis is based on [SR4].

In the course of the thesis, Elene Anton did an internship in Carnegie Mellon University under the supervision of Professor Mor Harchol-Balter. During this internship, she developed bounds on the tail of the response time of jobs for a single server queue and particularly for a priority queue. The relevant results of this research collaboration can be found in [SR7], but are not included in this manuscript as the topic was not related to multi-server systems.

Chapter 2

Introduction to stochastic processes

This chapter is devoted to provide a background on the main mathematical notions used throughout the thesis.

In Section 2.1 we give a summary on the notions used for random variables. In Section 2.2 we define the Lyapunov function, which will be used in Section 2.3, where we give a survey on the convergence of stochastic processes, and in Section 2.4, where we focus on the stability of stochastic processes. In Section 2.5 we present the light-traffic approximation, as a method to analyze the performance of stochastic processes.

2.1 Random variables

Let us consider a random variable X with probability distribution function (pdf) f_X, cumulative distribution function (cdf) F_X and complementary cumulative distribution functions (ccdf) $\bar{F}_X$ in the $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n))$ metric space. In this thesis, we assume that random variables live on the Euclidean space, that is the $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n))$ metric space where $\mathcal{B}(\mathbb{R}^n)$ is the Borel field of $\mathbb{R}^n$ with the uniform topology. In this chapter, we assume that the stochastic process $M(t)$ is Markovian and lives on the state space $E \subset \mathbb{R}^n_+$. Without loss of generality, $M(\cdot)$ is a Borel right process, see [26]. We denote by $\mathbb{P}_x(\cdot) := \mathbb{P}(\cdot | M(0) = x)$, $\mathbb{E}_x(\cdot) := \mathbb{E}(\cdot | M(0) = x)$.

We can classify a random variable X through the tail distribution $\bar{F}_X$. We define here two classes of distributions, New-Better-than-Used (NBU) and New-Worse-than-Used (NWU), see [88, 94] for more details.

Definition 2.1.1. The random variable X is said to be *New-Better-than-Used (NBU)* if for all $t_1, t_2 \in \mathbb{R}$,

$$\bar{F}_X(t_1 + t_2) \leq \bar{F}_X(t_1)\bar{F}_X(t_2).$$

The random variable X is said to be *New-Worse-than-Used (NWU)* if for all $t_1, t_2 \in \mathbb{R}$,

$$\bar{F}_X(t_1 + t_2) \geq \bar{F}_X(t_1)\bar{F}_X(t_2).$$

We let $R(x) = -\ln(\bar{F}_X(x))$ be the hazard function, and $r(x) = R'(x)$ be the hazard rate function. That is,

$$r(x) = \frac{f(x)}{\bar{F}(x)} \text{ for } x \in \mathbb{R}.$$

A sufficient condition for X to be NBU (NWU) is to have an increasing (a decreasing) hazard rate, i.e., $r(x)$ is increasing (decreasing) in x. The latter implies that the coefficient of variation, $cv := \sqrt{\mathbb{V}\mathrm{ar}\,(X)}/\mathbb{E}\,(X)$, is at most (least) 1.

In the following we give an equivalent definition for both NBU and NWU in the framework of queuing theory. Let X be the service time of a job in a server with unit capacity, and $X_t = \{X - t : X > t\}$ the remaining service time of a job that has completed t time units of its service time.

Definition 2.1.2. We say that a random variable X is *stochastically larger* than the random variable Y, written $X \geq_{st.} Y$, if

$$\mathbb{P}\,(X > t) \geq \mathbb{P}\,(Y > t)\,, \text{ for all } t \in \mathbb{R}^n_+.$$

Equivalently, $\bar{F}_X(t) \geq \bar{F}_Y(t)$ for all $t \in \mathbb{R}^n_+$. Note that if $X \geq_{st.} Y$, then $\mathbb{E}\,(X) \geq \mathbb{E}\,(Y)$.

Theorem 2.1.3. *The random variable X is New-Better-than-Used (NBU) if the remaining processing time of a job that has received some processing (is used) is stochastically smaller than the processing time of a task that has received no processing (is new), i.e., $X_t \leq_{st} X_0$ for all t.*

The random variable X is New-Worse-than-Used (NWU) if the remaining processing time of a job that has received some processing (is used) is stochastically larger than the processing time of a task that has received no processing (is new), i.e., $X_t \geq_{st} X_0$ for all t.

The notion of NWU and NBU comes from reliability theory, where it is "better" to have lifetimes that are long, that is, NBU. This can be counterintuitive in the queuing context, where long processing times are worse.

Depending on their tail properties, random variables can also be classified as heavy-tailed and light-tailed, see [38] for more details.

Definition 2.1.4. We say that X is *heavy-tailed* if the moment generating function $\mathbb{E}\left(e^{tX}\right)$ is infinite for all $t > 0$. We say that X is *light-tailed* if it is not heavy-tailed, that is, $\mathbb{E}\left(e^{tX}\right) < \infty$, for some $t > 0$.

We note that a light-tailed random variable has all its moments finite. Therefore, any random variable having an infinite moment can not be light-tailed, hence is heavy-tailed.

A sufficient condition for a random variable to be heavy-tailed is that $\lim_{x\to\infty} \bar{F}(x+t)/\bar{F}(x) = 1$, for all $t > 0$. Intuitively, under heavy-tailed distributions, if X ever exceeds a large value, then it is likely that it also exceeds an even larger value as well.

The deterministic, Erlang distributions and Hyperexponential distributions are ligh-tailed, but the deterministic and Erlang distributions are NBU, whereas the Hyperexponential distributions is NWU. The exponential distribution, which is *memoryless*, is both NBU and NWU, since its failure rate is constant, μ, as well as light-tailed.

Pareto distributions are defined through $\bar{F}(x) = \left(\frac{\kappa}{x+\kappa}\right)^{-\alpha}$, where $\kappa > 0$ is the scale parameter and $\alpha > 0$ shape parameter. Pareto distributions are NWU and heavy-tailed, and have all the moments of order $\gamma < \alpha$ finite, whereas the moments of order $\gamma \geq \alpha$ are infinite.

The Weibull distribution, is defined as $\bar{F}(x) = e^{-(x/\kappa)^{\alpha}}$, where $\kappa > 0$ is the scale parameter and $\alpha > 0$ the shape parameter. This distribution is NWU if $0 < \alpha < 1$, NBU if $\alpha > 1$ and coincides with the exponential distribution when $\alpha = 1$. Furthermore, it is heavy-tailed if and only if $\alpha < 1$ and all its moments are finite.

2.2 Lyapunov functions

Lyapunov functions are widely implemented when analyzing Markovian processes and particularly its stability region, see for instance the Foster-Lyapunov criteria in [87]. Stability is introduced in Section 2.4 and is the main topic of Chapters 3 and 4.

Lyapunov functions are scalar functions defined in the phase-plane in order to determine the stability of an equilibrium point. In the following we provide a rigorous definition for Lyapunov functions.

Definition 2.2.1 ([39])**.** A function $f : E \to \mathbb{R}_+$ is said to be a *Lyapunov function* with drift size parameter $-\gamma < 0$, drift time parameter $t_0 > 0$ and exception parameter K, if

$$\sup_{x\in E\,:\,f(x)>K} \{\mathbb{E}_x(f(M(t_0)) - f(x))\} \leq -\gamma.$$

In the following we define geometric Lyapunov functions, which will serve to characterize the stationary behavior of the process.

Definition 2.2.2 ([39])**.** A function $f : E \to \mathbb{R}_+$ is said to be a geometric Lyapunov function with geometric drift size $0 < \gamma < 1$, drift time parameter t_0 and exception parameter K, if

$$\sup_{x\in E\,:\,f(x)>K} \{(f(x))^{-1}\mathbb{E}_x(f(M(t_0)))\} \leq \gamma.$$

In the following theorem, which is introduced in [39], we provide an upper bound on the mean of the Markov process in stationarity. In Chapter 7, we analyze the convergence of a particular sequence of stochastic processes to its limiting stochastic process. We will construct a geometric Lyapunov function, which together with the following theorem shows that the sequence of stochastic processes is tight, defined in Definition 2.3.1.

Theorem 2.2.3 ([39]). *Let us consider the Markov process* $(M(t), t \geq 0)$ *with stationary distribution* Π *and assume that* f *is a* geometric Lyapunov function *with parameters* γ, t_0 *and* K*. For the system that is in steady-state at time* $t = 0$*,*

$$\mathbb{E}(f(M(0))) \leq \frac{\phi(t_0)K}{1-\gamma},$$

where $\phi(t) := \sup_{x \in E}\{(f(x))^{-1}\mathbb{E}_x(f(M(t)))\}$*, for* $t \geq 0$*.*

We note that the above theorem does not assume that the stationary distribution Π is unique. The bound holds for every stationary distribution Π, but is valuable only when $\phi(t_0)$ is finite.

2.3 Convergence of stochastic processes

In the present section we provide a background on convergence, and particularly on the convergence of stochastic processes and their probability measures. We recall that for a Markov processes $(M(t))_{t\geq 0}$ in the countable state space E, the function $t \to M(t)$ is càdlàg if almost surely, the function is continuous on the right on $\mathbb{R}_+$ and has left limit at any point, see [87, 14] for more details.

In Chapter 7 we consider a sequence of Markovian processes for which the limiting Markovian process is not a càdlàg function, but will rather resemble a white noise process. The latter represents a challenge when proving the convergence of the sequence of the Markovian processes to its limit. In the present section, we provide a background on the convergence of probability measures and stochastic processes.

Let us consider a sequence of stochastic processes $\{(M_n(t))_{t\geq 0}\}_{n\in\mathbb{N}}$ and its associated sequence of probability measures $\{P_n\}_{n\in\mathbb{N}}$. The convergence of the sequence $\{(M_n(t))_{t\geq 0}\}_{n\in\mathbb{N}}$ to its limiting stochastic process in decomposed into two steps. In the first step, the tightness of the sequence of probability distributions is proved. In the second step, the convergence in distribution of its finite dimensional distributions of the system is proved. Before we give the result, let us define tightness:

Definition 2.3.1 ([87]). Assume the sequence of stochastic processes $\{(M_n(t))_{t\geq 0}\}_{n\in\mathbb{N}}$ with probability measures $\{P_n\}_{n\in\mathbb{N}}$ and that at time $t = 0$, $M_n(0)$ is distributed

according to P_n, for all $n \in \mathbb{N}$. Then, $\{(M_n(t))_{t\geq 0}\}_{n\in\mathbb{N}}$ is said to be *tight*, if for any $\epsilon > 0$, there exists a sufficiently large constant $K > 0$ such that

$$\sup_n P(M_n(0) > K) < \epsilon.$$

In Chapter 7 we consider a sequence of stochastic processes and prove that their steady-state distribution is tight with respect to n. In order to do so, we consider a positive increasing function ϕ, which is geometric Lyapunov with parameters λ, t_0, K (Definition 2.2.2), with $x \leq \phi(x)$, for all $x \geq 0$. Then, for the system $M(0)$ starting in steady-state the following inequalities hold:

$$\begin{aligned}\sup_n P(|M_n(0)| > C) &\leq \sup_n P(\phi(|M_n(0)|) > \phi(C)) \leq e^{-\phi(C)} \sup_n \mathbb{E}(e^{\phi(|M_n(0)|)}) \\ &\leq e^{-\phi(C)} \frac{\psi(t_0)K}{1-\gamma},\end{aligned}$$

where $\psi(t) := \sup_{x\in E}\{(f(x))^{-1}\mathbb{E}_x(f(M(t)))\}$. The second inequality holds because of the Markov inequality and the third inequality holds due to Theorem 2.2.3. Hence, the $\sup_n P(|M_n(0)| > C)$ is bounded by $e^{-\phi(C)}\frac{\psi(t_0)K}{1-\gamma}$, where the second term is constant and $e^{-\phi(C)} \to 0$ as $C \to \infty$. Therefore, from Definition 2.3.1 the sequence of steady-state distributions of the stochastic processes $\{(M_n(t))_{t\geq 0}\}_{n\in\mathbb{N}}$ is tight.

In the following, we state the convergence of a sequence of stochastic processes, which is given by the convergence in distribution of its probability measures.

Proposition 2.3.2 ([87]). *The sequence of stochastic processes $\{(M_n(t))_{t\geq 0}\}_{n\in\mathbb{N}}$ on E, with probability distributions $\{P_n\}_{n\in\mathbb{N}}$, converges in distribution to $(M(t))_{t\geq 0}$, with probability distribution P, if the following conditions are satisfied:*

- *the sequence $\{P_n\}_{n\in\mathbb{N}}$ is tight,*
- *for any $p \in \mathbb{N}$ and $t_1, \ldots, t_p \in [0, T]$*

$$\lim_{n\to\infty} P_n(M(t_1) \in \cdot, \ldots, M(t_p) \in \cdot) = P(M(t_1) \in \cdot, \ldots, M(t_p) \in \cdot).$$

That is, the finite marginal distributions of P_n converge to that of P.

In Chapters 3, 4 and 6, we consider a Markov process and analyze its fluid limit, introduced later in Section 2.4.2.1. In order to show that the Markovian process converges to its fluid limit, we invoke the Arzela-Ascoli theorem. In order to introduce this theorem, let us consider a family of functions $\{f_n\}_{n\in\mathbb{N}}$ defined in $\mathbb{R} \to \mathbb{R}$. The Arzela-Ascoli theorem shows that any bounded sequence of f_n has a convergent subsequence to its limiting function f.

Theorem 2.3.3 ([101]). *Suppose that $\{f_n\}$ is a sequence of functions $\mathbb{R} \to \mathbb{R}$ which is Lipschitz continuous with uniform Lipschitz parameter K and pointwise bounded, that is, for all $x \in E$ the sequence $\{f_n(x)\}_{n=1}^{\infty}$ is bounded. Then, there exists a subsequence n_k of n which converges uniformly on compact sets to a continuous function f.*

We recall that *uniformly on compact sets (u.o.c.)* is defined as follows: for each compact set C and each $\epsilon > 0$, there exists a K such that

$$|f_{n_k}(x) - f_{n_j}(x)| < \epsilon, \text{ for all } t \in C, \text{ for all } k, j \geq K.$$

2.4 Stability region and steady-state distribution

In Chapters 3 and 4, we characterize the stability region of various redundancy systems. Intuitively, the stability region determines a set of parameters such that the system remains stable, that is, the queue lengths do not build up. From the mathematical viewpoint, we say that a system is *stable* if there exists a stationary probability measure for the queue length process, that is, it is ergodic. Moreover, we say that the system is unstable if it is not ergodic. See [88, 26] for a survey in this topic. In the following, we give a mathematical definition for ergodic systems.

Definition 2.4.1 ([26]). A continuous time Markov chain is said to be *ergodic* if it possesses a stationary probability measure Π for which $\lim_{t\to\infty} ||\mathbb{P}_x(M(t) \in A) - \Pi(A)|| = 0$ for all $x \in E$ and $A \subset E$.

The stationary probability measure, a.k.a. steady-state distribution, is also derived in the following way:

$$\Pi(A) = \lim_{t\to\infty} \frac{1}{t} \int_0^t 1_{(M(u)\in A)} \mathrm{d}u.$$

We say that a process achieved stationarity, or steady-state, when a new transition of the system does not affect the state probability measure. The steady-state distribution allows for example to obtain the mean number of jobs in the system. By Little's law [76], the mean number of jobs in the system is proportional to the mean response time of a job in the system.

In the following we present some particular systems, for which the steady-state distribution is of a product-form.

2.4.1 Product-form steady-state distribution

Under product-form distributions, the stationary probability a state is given by the product of the terms involved in that state. The stability region of the system can then be determined by setting the normalization constant is such that the sum of all the

probabilities equals 1. Furthermore, this structure might facilitate the derivation of other performance measures such as the mean response time.

In the following we introduce the order-independent queues, a queuing system that is subsumed in the class of systems with a product-form steady-state distribution. We also introduce the class of systems in [102], where through the partial-balance method, the authors obtain the steady-state distribution. However, we note that these are not the only systems in the class of system with a product-form steady-state distribution.

In Section 1.1.3 we overviewed the literature on the performance evaluation of redundancy models, where several studies present redundancy models with a product-form steady-state distribution. These studies show that various redundancy models fall into the framework of either order-independent queues or the multi-type job and multi-type server system in [102], which present a product-form steady-state distribution. These are the *c.o.c.* redundancy model ([10, 18]) and the *c.o.s.* redundancy model ([10]), both with i.i.d. copies, FCFS and exponentially distributed service times.

Order-independent queues

Let us consider a queue where incoming jobs are of type v, with $v = 1, \ldots, V$ and V the number of job types. Jobs arrive according to a Poisson process of rate λ and are of type v with probability p_v, with $\sum_{v=1}^{V} p_v = 1$. Type-v jobs have exponentially distributed service times with mean $1/\mu_v$. Arriving customers join the back of the queue and enter the first compatible server that is idle. Let us denote by $\vec{v} = (v_n, v_{n-1}, \ldots, v_2, v_1)$ the state descriptor of the system where n is the number of jobs present and v_i is the type of the ith eldest job present in the system. We denote by $\phi(v_n, \ldots, v_1)$ the total service rate when in state $(v_n, \ldots, v_1)$ and by $\gamma_i(v_n, \ldots, v_1)$ the fraction of service rate provided to the ith job in the queue, for $i = 1, \ldots, n$, where $\sum_{i=1}^{n} \gamma_i = 1$.

Definition 2.4.2 ([70])**.** A queuing system is said to be an *order-independent queue* if for all $(v_n, \ldots, v_1)$ and all $i = 1, \ldots, n$, the service rates can be written as follows

$$\phi(v_n, \ldots, v_1)\mu_{v_i}\gamma_i(v_n, \ldots, v_1) = \mu(n)s_i(v_n, \ldots, v_1),$$

such that

(*i*) $s_i(v_n, \ldots, v_1) = s_i(v_i, \ldots, v_1)$ for all $1 \leq i \leq n$,

(*ii*) $k(v_n, \ldots, v_1) = \sum_{i=1}^{n} s_i(v_n, \ldots, v_1)$ is independent of permutations of $(v_n, \ldots, v_1)$,

(*iii*) $\mu(n) > 0$ for any $n > 0$ and $s_1(v) > 0$ for any type $v = 1, \ldots V$.

The function $s_i(v_n, \ldots, v_1)$ determines the rate at which service is given to the ith job in the queue and the function $\mu(n)$ allows the service rate to depend upon the total

number of jobs in the queue. We note that condition (i) implies that the service rate received by the ith job is independent of the jobs that are behind that job. Furthermore, condition (ii) implies that the service rate received by the ith job is independent of the order of the jobs ahead of it in the queue.

The authors in [70] prove that order-independent queues have a steady-state distribution which is of product-form and are quasi-reversible. See [67] for more details on quasi-reversibility.

Theorem 2.4.3 ([70])**.** *The steady-state distribution of an order-independent queue where $s_i(\cdot)$ satisfies conditions (i) - (iii) for all $(v_n, \ldots, v_1)$ is given by*

$$\Pi(v_n, v_{n-1}, \ldots, v_2, v_1) = C \prod_{i=1}^{n} \frac{\lambda p_{v_i}}{\mu(i) k(v_i, \ldots, v_1)},$$

where C is the normalization constant. Furthermore, the queue is quasi-reversible.

Bonald and Comte [18] prove that the *c.o.c.* redundancy model with FCFS, exponentially distributed service times and i.i.d. copies falls into the order-independent queue framework, and hence, has a product-form steady-state distribution.

Multi-type job and multi-type server model

In the following, we introduce another method in order to show that a given system presents a product-form steady-state distribution. Visschers et al. [102] consider a system where servers implement FCFS and jobs service times are exponentially distributed. Arriving jobs that find several compatible servers idle, are dispatched to one of them according to some assignment rule.

The authors consider the following aggregated state descriptor: $(l_i, s_i, \ldots, l_1, s_1)$, is the state where there are i busy servers, $s_1, \ldots, s_i \subset S$ and l_j denotes the number of jobs that arrived after the job that is in service in server s_j and has as compatible server a subset of the set of servers $\{s_1, \ldots, s_j\}$, for $j = 1, \ldots, i$.

For the present system, the authors characterize the steady-state distribution of the model under some mild assumptions on the assignment probability distributions of the assignment rule. To do so, the authors present a solution candidate of a product-form and prove that the candidate solution satisfies the partial-balance equations. This implies that the balance equations are satisfied, and hence the steady-state distribution is characterized, which is of product-form.

Theorem 2.4.4 ([102])**.** *Assume that the assignment rule satisfies the assignment condition, that is, for every $i = 1, \ldots, K$ and any subset $\{s_1, \ldots, s_i\}$ of S of size i, the*

following is satisfied:

$$\prod_{j=1}^{i} \lambda_{s_i}(\{s_1, \ldots, s_{i-1}\}) = \prod_{j=1}^{i} \lambda_{\bar{s}_i}(\{\bar{s}_1, \ldots, \bar{s}_{i-1}\}),$$

for every permutation $\bar{s}_1, \ldots, \bar{s}_i$ *of* $s_1, \ldots, s_i$*, where* $\lambda_{s_i}(\{s_1, \ldots, s_{i-1}\})$ *is the rate at which server* s_i *receives a copy to serve when servers* $\{s_1, \ldots, s_{i-1}\}$ *are already busy. Then, the steady-state distribution is*

$$\Pi(l_i, s_i, \ldots, l_1, s_1) = \left(\frac{\lambda_{\mathcal{U}(\{s_1,\ldots,s_i\})}}{\mu_{\{s_1,\ldots,s_i\}}}\right)^{l_i} \frac{\lambda_{s_i}(\{s_1, \ldots, s_{i-1}\})}{\mu_{\{s_1,\ldots,s_i\}}} \cdots \left(\frac{\lambda_{\mathcal{U}(\{s_1\})}}{\mu_{\{s_1\}}}\right)^{l_1} \frac{\lambda_{s_1}(\emptyset)}{\mu_{\{s_1\}}} \Pi(0),$$

where $\mu_{\{s_1,\ldots,s_i\}} = \sum_{j=1}^{i} \mu_{s_j}$*,* $\lambda_{\mathcal{U}(\{s_1,\ldots,s_i\})} = \sum_{v \in \mathcal{U}(\{s_1,\ldots,s_i\})} \lambda p_v$*, and* $\mathcal{U}(\{s_1, \ldots, s_i\}) = \{v : S(v) \subseteq \{s_1, \ldots, s_i\}\}$*, with* $S(v)$ *the servers that can serve a type-*v *job.*

Ayesta et al. [10] show that *c.o.s.* redundancy models with FCFS, exponentially distributed service times and i.i.d. copies fit within the framework of multi-type jobs and multi-type servers system studied in Visschers et al. [102], which implies that the steady-state distribution is of product-form. The above results have motivated the development of unifying frameworks to explain the emergence of product-form distributions in redundancy models. For instance, Ayesta et al. [9] and Gardner and Righter [45] extend the frameworks of Visschers et al. [102] and order-independent queues [70], respectively.

2.4.2 Stability region

In the following, we present some criteria to prove whether a Markov process is either stable or unstable. These criteria consider qualitative aspects of the process rather than finding the whole steady-state distribution, which can be extremely challenging. In this thesis we consider stochastic processes with both countable and non-countable state spaces. As we will observe in the following, having a countable state space simplifies the derivation of the sufficient and necessary stability conditions.

Countable state spaces

Let us consider stochastic systems with countable state spaces. For instance, the $M/M/K$ system with FCFS has a Markovian state descriptor $(M(t))_{t\geq 0}$, where $M(t)$ denotes the number of jobs in the system at time t, and the state space is $E = (\mathbb{N} \cup \{0\})$. In Chapter 3, we analyze stochastic processes that have a countable state space. See Robert [87] and Bramson [26] for more details on the stability of systems with countable state spaces.

In the following, we characterize when the system is stable. For countable state spaces, ergodicity and positive recurrence are equivalent.

Definition 2.4.5 ([26])**.** An irreducible Markovian stochastic process is said to be *recurrent* if for all states $x, y \in E$, $\mathbb{P}_x(\tau_y < \infty) = 1$, otherwise it is *transient*, where $\tau_y = \inf\{t > 0 \ : \ M(t) \ = y\}$. Additionally, an irreducible recurrent Markovian stochastic process is said to be *positive recurrent* if for all states $x, y \in E$, $\mathbb{E}_x(\tau_y) < \infty$, and *null recurrent* if for all states $x, y \in E$, $\mathbb{E}_x(\tau_y) = \infty$.

In order to prove stability of its stochastic process, we will apply Foster's criteria, a.k.a. Foster-Lyapunov's criteria, which is commonly used to demonstrate positive recurrence.

Theorem 2.4.6. ([87, Thorem 8.13]) *Consider a irreducible Markov process in a countable state space. If there exists a function $f : E \to \mathbb{R}_+$, constants K, $\gamma > 0$ and a integrable stopping time τ such that for $f(x) > K$,*

$$\mathbb{E}_x\left(f(M(\tau)) - f(x)\right) \leq -\gamma \mathbb{E}_x\left(\tau\right),$$

if $F := \{x \in E \ : \ f(x) \leq K\}$, the hitting time T_F of F is integrable and $\mathbb{E}_x\left(T_F\right) \leq f(x)/\gamma$. Moreover, if F is a finite subset and $\mathbb{E}_x\left(f(M(1))\right) < +\infty$ for all $x \in E$, then the Markov process is ergodic.

We note that the function f is a Lyapunov function and $\mathbb{E}_x\left(M(\tau) - x\right)$ is know as the one-step drift of the process.

We note that most of the times, even if the state representation is tractable, we will not be able to find a function that has a positive or negative drift for the stochastic model. For this reason, in Section 2.4.2.1 we introduce the fluid-limit approximation, which is a valuable approximation that allows to prove stability of the stochastic process.

Non-countable state spaces

Let us consider stochastic systems with non-countable state spaces. For instance, the $M/G/1$ system with PS has a Markovian descriptor $(M(t), r_1(t), \ldots, r_{M(t)}(t))_{t \geq 0}$, where $M(t)$ denotes the number of jobs in the system and $r_i(t)$ denotes the residual service time of the ith eldest job in the system, for $i = 1, \ldots, M(t)$, at time t. Then, $M(t) \in \mathbb{N} \cup \{0\}$ and $r_i(t) \in \mathbb{R}_+$ for all i. Thus, the state space is non-countable. In this thesis, the stochastic processes analyzed in Section 4.2 have a non-countable state space.

For non-countable state spaces in the Euclidean space, ergodicity and positive Harris recurrence are equivalent, [48, 78]. For a survey on stability of systems with non-countable state space, see for instance [79, 7, 26].

In order to define Harris recurrence, we first introduce the petite sets.

Definition 2.4.7 ([26])**.** A set $A \in E$ is called a *petite set* if for probability measure F on $(0, \infty)$ and some non trivial measure ν on $(E, \mathcal{B}(E))$,

$$\nu(B) \leq \int_0^\infty \mathbb{P}_x \left(M(t) \in B\right) F(dt),$$

for all $x \in A$ and $B \in \mathcal{B}(E)$.

An intuitive explanation for petite sets is that each set B is "equally accessible" from all points $x \in A$ with respect to the measure ν. In addition, when the probability measure F is concentrated in a single point, A is said to be a *small* set.

Definition 2.4.8 ([26])**.** A Markovian stochastic process is said to be *Harris recurrent* if there exists a closed petite set A with $\mathbb{P}_x \left(\tau_A < \infty\right) = 1$, for all $x \in E$, where $\tau_A = \inf\{t > 0 \ : \ M(t) \in A\}$. Furthermore, a Harris recurrent Markov chain is *positive Harris recurrent* if there exists a closed petite set A such that for some $\delta > 0$, $\sup_{x \in A} \mathbb{E}_x \left(\tau_A(\delta)\right) < \infty$, where $\tau_A(\delta) = \inf\{t > \delta \ : \ M(t) \in A\}$.

In the particular case where the state space is countable and irreducible, $\{0\}$ is uniformly accessible from A. Thus, when the state space is countable and irreducible, Harris recurrence reduces to positive recurrence. Moreover, under this setting, all points $\{x\}$ are small set, for $x \in E$.

2.4.2.1 Fluid-limit approximation

In the present section, we introduce the fluid limit for a given stochastic process. The fluid limit is an approximation that captures the general dynamics of the system and is easier to study than the stochastic model. Furthermore, there exist results (e.g. regarding stability) that translate the results obtained for the fluid limit back to the stochastic process. For instance, in Chapters 3 and 4 we will rely on fluid limits in order to establish the stability of the redundancy models. We refer to [87, 26] for more details on the fluid-limit approximation.

Let us consider a continuous time Markovian process $M(t)$. For any $r > 0$, we define the fluid-scaled process $M^r(t)$ with initial state $M(0) = rm$ as

$$\bar{M}^r(t) = \frac{1}{r} M^r(rt).$$

We note that the state is scaled by a factor $1/r$ whereas the time is accelerated by r. Then, a fluid limit $m(t)$ associated to process $M(t)$ is defined as the limit of a converging subsequence of r:

$$\lim_{r \to \infty} \bar{M}^r(t) = \lim_{r \to \infty} \frac{M^r(rt)}{r}.$$

Intuitively speaking, we speed up the time by parameter r and apply a Law of Large Numbers in order to obtain the "average" performance of the stochastic process $(M(t))_{t\geq 0}$. Therefore, the fluid-limit approximation captures the macroscopic dynamics of the system whereas microscopic transitions vanish.

For a multi-server system where respectively the arrival times and the departure times follow i.i.d. random variables with finite mean and the cumulative amount of capacity/time spent on serving each job is a Lipschitz continuous function, it is shown that the process has a uniformly convergent subsequence to a fluid limit, see for instance [26]. Furthermore, the fluid limit can be written as the solution to a differential equation, which is constructed from the one-step drift of the process.

For instance, consider the single-server system where FCFS is implemented and jobs arrive according to a Poisson process with arrival rate λ and have exponentially distributed service times with mean $1/\mu$. Using standard arguments, see [26], the fluid-scaled number of jobs in the system, denoted by $M^r(t)$, with initial $M(0) = rm$ is defined as

$$\frac{M^r(rt)}{r} = m + \frac{1}{r}\tilde{A}(rt) - \frac{1}{r}\tilde{S}(T^r(rt)), \tag{2.1}$$

where $\tilde{A}(t)$ and $\tilde{S}(t)$ are independent Poisson processes having rates λ and μ, respectively, and $T(t)$ is the cumulative amount of capacity spend on serving job during the time interval $(0, t]$. For the present system, [26] shows that the process has a uniformly convergent subsequence to a fluid limit. Furthermore, the fluid limit $m(t)$ can be written as the solution to a differential equation, which is constructed from the one-step drift of the process. This differential equation is

$$\frac{\mathrm{d}m(t)}{\mathrm{d}t} = \lambda - \mu 1_{(m(t)>0)},$$

and has solution $m(t) = m(0) + t(\lambda - \mu)^+$.

In the following we are interested in the stability of a stochastic process by analyzing its fluid limit, an approach that we deploy in Chapters 3 and 4. Let us first introduce the stability of the fluid model:

Definition 2.4.9 ([26])**.** A fluid limit model associated to a stochastic process is said to be *stable*, if there exists a constant T such that all associated fluid limits satisfy, $m(t) = 0$ for all $t \geq T|m(0)|$. A fluid limit model is said to be *unstable*, if for some initial state $m(t) \to \infty$ as $t \to \infty$, and*weakly unstable*, if there exists $t_0 > 0$ such that for each fluid limit solution with initial state $m(0) = 0$, it holds that $m(t_0) \neq 0$.

For the countable state space, the stability of the stochastic process follows from the stability of its fluid limit. However, the latter is no longer the case for non-countable state spaces. We will again give stability results first for countable state spaces and then for non-countable state spaces.

Countable state spaces

The stability of countable state spaces through its fluid limit is studied in [87]. The following theorem gives a stability result of the stochastic process by the study of its fluid limit.

Theorem 2.4.10 ([87])**.** *In a countable state space, the Markovian stochastic process is positive recurrent if there exists a constant T such that the associated fluid limit at time T,*

$$\lim_{r\to\infty} \sup \mathbb{E}_{rm}(|\bar{M}^r(T)|) = \lim_{r\to\infty} \sup \frac{\mathbb{E}_{rm}(|M^r(rT)|)}{r} = 0.$$

Even in the fluid limit, where the stochastic microscopic transitions are oblivious for the macroscopic dynamics of the system, showing that the fluid-scaled system converges to 0 can be challenging. For instance, in Section 3.4, we characterize a system where the process has a negative one-step drift, except for certain states where the drift is discontinuous. Many other systems in literature also present this feature, and in [47] the authors present a framework in order to analyze the fluid limit of a process with discontinuous drifts. The authors define a function F as the convex hull of the accumulation points of the fluid limit from different trajectories.

Proposition 2.4.11 ([47])**.** *Assume that $\lim_{R\to\infty} \mathbb{E}_x\left(|M(\tau) - M(0)|\mathbf{1}_{(|M(\tau)-M(0)|>R)}\right) = 0$ and that the drift of the process $f(m_0) = \mathbb{E}_x\left(M(\tau) - M(0)\right)$ is bounded. Let F be a set-valued function defined as*

$$F(m) := conv\left(acc_{r\to\infty} f(r \times m^r) \text{ with } \lim_{r\to\infty} m^r = m\right),$$

where $conv(A)$ is the convex hull of set A and $acc_{r\to\infty}\, m^r$ denotes the set of accumulation points of the sequence m^r when r goes to infinity. Then, the set of solutions of the differential inclusion $f(m) \in F(m)$ starting in x contains the fluid limits of M^r.

An illustration of how F is constructed is available in Figure 1 in [47, Section 2].

Non-countable state spaces

The stability of the fluid limit and its associated stochastic process for non-countable state spaces has been analyzed by several papers [26, 32, 31, 33, 73]. We note that, for a system with non-countable state space, the transition from the stability/instability of the fluid limit to the stability/instability is challenging to obtain.

In the following, we introduce sufficient conditions for fluid stability to imply stability of the original stochastic system. These conditions are differentiated by the scheduling policy implemented in the system.

We first focus on *Head-of-the-Line (HL)* policies, introduced in [26, 32]. Under HL policies the total amount of capacity (or "effort") already devoted to partially served jobs

does not increase without bound. Particularly, policies where only the first job in each server receive service, such as FCFS and Priority policies with FCFS, are HL policies. The Head-of-Line PS where the eldest job of each type receives service simultaneously at each server is also a HL policy. However, PS, as well as Last-Come-First-Serve (LCFS), are not HL policy.

For HL policies, the following theorem states that for any open queuing network where jobs have general service times, the stability of the fluid limit implies the stability of the stochastic process.

Theorem 2.4.12 ([32])**.** *In an open queuing network where an HL scheduling policy is implemented, stability of the fluid limit implies stability of the stochastic process.*

We note that the latter does not consider the PS scheduling policy, in which we are interested in this manuscript. In [73], the authors study bandwidth-sharing networks (with processor sharing policies), and show that under mild conditions, the stability of the fluid model implies positive Harris recurrence of the stochastic process. The conditions for that statement to hold include that the service time distribution is light-tailed. However, [32] and [73] conjecture that the statement might also be applicable for the system where PS is implemented and jobs have general service time distributions.

Theorem 2.4.13 ([73])**.** *Consider a multi-server system where PS is implemented in the servers and the service time distribution F of the jobs satisfy the following two conditions:*

1. *F has no atoms.*
2. *F is a light-tailed distribution in the following sense,*

$$\lim_{r\to\infty} \sup_{a\geq 0} \mathbf{E}[(X-a)1_{\{X-a>r\}}|X>a] = 0. \tag{2.2}$$

Then, the underlying stochastic system is positive Harris recurrent if its fluid limit is stable.

We note that Eq. (2.2) implies the following equation,

$$\sup_{a\geq 0} \mathbf{E}[(X-a)|X>a] \leq \Phi < \infty, \tag{2.3}$$

which is a usual light-tailed condition (see [38]). Hence, Eq. (2.2) includes large sets of distributions, such as phase-type distributions (which are dense in the set of all distributions on $\mathbb{R}^+$), exponential and hyper-exponential distributions, as well as distributions with bounded support.

The theorem below, originally in [33], considers a system with a weakly unstable fluid limit and shows that the stochastic process can not be stable.

Theorem 2.4.14 ([33]). *If the fluid model of an open queuing network is weakly unstable, then the open queuing network is unstable in the sense that, with probability 1,*

$$|M(t)| \to \infty \text{ as } t \to \infty.$$

2.5 Light-traffic approximation

The light-traffic approximation corresponds to the performance of the system when the load approaches 0, i.e. $\lambda \to 0$. This approximation was pioneered by [85, 105] and has been successfully applied in many papers such as [62, 86]. The main objective is to obtain the mean number of jobs, and mean sojourn time, a.k.a. response time, as the load approaches zero, for systems where these measures are challenging to find. In this thesis, we use light-traffic approximations to obtain insights on the performance of redundancy models, Chapter 3, and mobility models, Chapter 7.

Let us denote by $\bar{D}(\lambda, b)$ the mean sojourn time for a tagged job conditioned on its size being b, for $\lambda > 0$. Let us assume that the first n derivatives of $\bar{D}(\lambda, b)$ exist for $\lambda = 0$. We have the following approximation for $\bar{D}(\lambda, b) := \bar{D}^{LT}(\lambda, b) + O(\lambda^{n+1})$, as $\lambda \to 0$, where

$$\bar{D}^{LT}(\lambda, b) := \bar{D}^{(0)}(0, b) + \lambda \bar{D}^{(1)}(0, b) + \ldots + \frac{\lambda^n}{n!} \bar{D}^{(n)}(0, b), \tag{2.4}$$

is referred to as the light-traffic approximation of order n. Here $\bar{D}^{(0)}(0, b) = D^{(0)}(0, b)$ and $\bar{D}^{(i)}(0, b) = \frac{\partial^i D(\lambda, b)}{\partial \lambda^i}|_{\lambda=0}$, for $i = 1, \ldots, n$. The i-th term, $\bar{D}^{(i)}(0, b)$, is related to the mean sojourn time when in addition to the tagged job, i other jobs arrive to the system in the interval $(-\infty, \infty)$. The choice of the value n establishes a compromise between the accuracy of the approximation and the complexity of its derivation.

We restrict to the light-traffic approximation of order 1, that is, in Eq. (2.4) we set $n = 1$. That is, we calculate the sojourn time of the tagged job conditioned on, at most, having one other job present in the system. Let $\tilde{A}(t_0, t_1)$ denote the number of arrivals in the time interval $[t_0, t_1)$ in addition to the tagged job who is assumed to arrive at time 0. Then, the zeroth and first light-traffic derivatives satisfy

$$\bar{D}^{(0)}(0, b) := \mathbb{E}\left(\bar{D}(0, b) | \tilde{A}(-\infty, \infty) = 0\right) \tag{2.5}$$

and

$$\bar{D}^{(1)}(0, b) := \int_{-\infty}^{\infty} \left(\mathbb{E}\left(\bar{D}(0, b) | \tilde{A}(-\infty, \infty) = 1, \tau = t\right) - \mathbb{E}\left(\bar{D}(0, b) | \tilde{A}(-\infty, \infty) = 0\right)\right) \mathrm{d}t, \tag{2.6}$$

where τ is the arrival time of the other job.

The intuitive idea behind this is that the states where two or more jobs are present in the system are neglected, as these states will become negligible in the limit $\lambda \to 0$. Indeed, when λ is small, starting empty the system evolves as follows: for a long duration, of the order of $1/\lambda$, nothing happens. Then, an arrival occurs. The job stays in the system for a $O(1)$ duration during which no new arrival occurs, since it typically occurs after a duration of the order of $1/\lambda$. Thus, states with two or more jobs are exceptional and can be neglected.

Chapter 3

Stability of the redundancy-d system

In the present chapter, we characterize the stability condition of the redundancy-d model, that is, the K homogeneous server system where each arriving job dispatches d copies into d out of K servers chosen uniformly at random. We assume that jobs are canceled when a first copy completes service, i.e., the *c.o.c.* system. We consider that jobs have exponentially distributed service times with unit mean and either i.i.d. copies or identical copies.

We define the total traffic load by $\rho := \frac{\lambda}{\mu K}$, where μ is the capacity of a server. Note that without redundancy, i.e., $d = 1$, the system is stable if and only if $\rho < 1$ for any work-conserving policy employed in the servers. We further note that for both i.i.d. copies and identical copies, the stability region might reduce, but cannot increase. This follows since under exponentially distributed service times and homogeneous server capacities, the total departure rate is at most $K\mu$, while the total arrival rate is λ. Hence, $\rho < 1$ is a necessary stability condition for any value of d.

For the present model, Gardner et al. [43, 46] and Bonald and Comte [18] show that when FCFS is implemented and jobs have i.i.d. copies, $\rho < 1$ is a sufficient stability condition. That is, the stability region is not reduced due to adding redundant copies.

In this chapter we investigate whether $\rho < 1$ is also the stability condition for other scheduling policies than FCFS, and/or for non-i.i.d. copies. In Table 3.1 we summarize all the findings of this chapter.

In the case where jobs have i.i.d. copies, we prove that for both PS and ROS, the stability region is not reduced when adding redundant copies. This statement might lead the reader to think that $\rho < 1$ is the stability condition whenever copies are i.i.d. copies and a work-conserving policy is implemented. However, this is not the case, and we present a counterexample based on a priority policy.

For the case when jobs have identical copies, we observe that the stability condition strongly depends on the employed scheduling policy. When ROS is implemented,

Table 3.1 Summary of stability conditions

	FCFS	**PS**	**ROS**
i.i.d. copies	$\rho < 1$	$\rho < 1$ (Prop 3.2.3)	$\rho < 1$ (Prop 3.2.3)
identical copies	$\rho < \bar{\ell}/K$ (Prop 3.3.3)	$\rho < 1/d$ (Prop 3.4.1)	$\rho < 1$ (Prop 3.5.3)

redundancy does not impact the stability region, that is, it remains $\rho < 1$.

When FCFS is implemented, the stability condition is reduced to $\rho < \bar{\ell}/K$, where $\bar{\ell}$ denotes the mean number of jobs in service in an associated saturated system that we characterize. It holds that $\bar{\ell} < K - (d-1)$, which follows from the fact that the oldest job in the system is always served at all its compatible servers, and thus, the maximum number of jobs in service is $K - (d-1)$. Furthermore, we numerically observe that $\bar{\ell}/K$, and hence the stability region, decreases as the number of copies d increases.

Under PS, the system is stable if $\rho < 1/d$, which coincides with the stability region of a system where all d copies of each job are fully served. Hence, PS represents the worst possible reduction in stability.

The stability analysis under redundancy when copies are identical is challenging because of the synchronized departures. The latter generates a strong correlation among the departure processes in the servers. In order to prove the stability condition under FCFS and PS, we resort to upper and lower bounds for which the departure process is easier to characterize.

Through the light-traffic regime, we analyze the mean number of jobs, and observe that for sufficiently low loads, redundancy does improve the performance of the system. Numerical analysis also corroborates the latter statement.

In summary, the main takeaway message of this chapter is that when the servers are identical and jobs have identical copies, the stability region can be drastically reduced when adding redundant copies. Both stability and performance, strongly depend on the employed scheduling policy within the servers.

The rest of the chapter is organized as follows: In Section 3.1 we give a detailed description of the model. In Section 3.2 we present the stability results for i.i.d. copies. In Section 3.3, Section 3.4 and Section 3.5 we focus on identical copies and present the stability results for FCFS, PS and ROS, respectively. In Section 3.6 we provide the light-traffic analysis and obtain insights on the performance of the systems through simulations. All proofs are deferred to Appendix 3.8, for easy of readability.

3.1 Model description

We consider the redundancy-d model for a K parallel-server system with homogeneous capacities μ. We denote by S the set of all servers, $S = \{1, \ldots, K\}$. Under redundancy-d, each job will be assigned a type label, $c = \{s_1, \ldots, s_d\}$, with $s_1, \ldots, s_d \in S, s_i \neq s_j, i \neq j$, to indicate the d servers to which a copy is sent.

We denote by $\mathcal{C}$ the set of all types, that is, $\mathcal{C} := \{\{s_1, \ldots, s_d\} \subset S \ : \ s_i \neq s_j, \quad \forall i \neq j\}$ and $|\mathcal{C}| = \binom{K}{d}$. The probability that a job is of type c equals $p_c = 1/\binom{K}{d}$ for all $c \in \mathcal{C}$. We denote by $\mathcal{C}(s)$ the subset of types that are served at server s, that is, $\mathcal{C}(s) = \{c \in \mathcal{C} \ : \ s \in c\}$. The number of types served at server s equals the number of possible ways to choose $d-1$ servers out of the remaining $K-1$ servers, that is, $|\mathcal{C}(s)| = \binom{K-1}{d-1}$, and $\lambda \frac{\binom{K-1}{d-1}}{\binom{K}{d}} = \lambda d/K$ is the arrival rate per servers. Therefore, under the redundancy-d model all servers are equally loaded.

We assume that jobs have exponentially distributed service times with unit mean and that the copies of a job are either i.i.d. copies or identical copies.

We denote by $N_c(t)$ the number of type-c jobs at time t and $\vec{N}(t) = (N_c(t), c \in \mathcal{C})$. Furthermore, we denote by $M_s(t) := \sum_{c \in \mathcal{C}(s)} N_c(t)$, $s = 1, \ldots, K$, the number of copies in server s, and $\vec{M}(t) = (M_1(t), \ldots, M_K(t))$. For the i-th type-c job, let b_{cis} denote the realization of the service requirement of its copy in server s, $i = 1, \ldots, N_c(t)$, $s \in c$. Note that in case the copies are identical, then $b_{cis} = b_{ci}$ for all $s \in c$. We let $a_{cis}(t)$ denote the attained service in server s of the i-th type-c job at time t. We denote by $A_c(t) = (a_{cis}(t))_{is}$ a matrix on $\mathbb{R}_+$ of dimension $N_c(t) \times d$. Note that the number of type-c jobs increases by one at rate $\frac{\lambda}{\binom{K}{d}}$, which implies that a row composed of zeros is added to $A_c(t)$. When one element $a_{cis}(t)$ in matrix $A_c(t)$ reaches the required service b_{cis}, the corresponding job departs and all of its copies are removed from the system. Hence, row i in matrix $A_c(t)$ is removed. The rate at which the attained service $a_{cis}(t)$ increases is determined by the employed scheduling policy in that server.

Within a server, a scheduling policy determines how the capacity of the server is shared among the copies. Within this chapter, we mostly focus on three scheduling policies: *(i)* First-Come-First-Serve (FCFS), where copies within a server are served in order of arrival, *(ii)* Processor Sharing (PS), where each copy in server s receives capacity $1/M_s(t)$, and *(iii)* Random Order of Service (ROS), where an idle server chooses uniformly at random a new copy from its queue. All these three policies have in common that they schedule only based on $\{N_c(t), c \in \mathcal{C}\}_{t \geq 0}$. Furthermore, for the system under i.i.d. copies, the latter is also the Markovian descriptor of the system. However, for the system under identical copies, $\{N_c(t), A_c(t), c \in \mathcal{C}\}_{t \geq 0}$ is the Markovian descriptor, which has a non-countable state space. As to distinguish between the different policies, we will add a superscript $\{FCFS, PS, ROS\}$ to the process $\vec{N}(t)$.

3.2 Independent identically distributed copies

In this section, we analyze the stability of the redundancy-d model when copies of a job are i.i.d.. In Section 3.2.1, we prove that the stability condition under PS and ROS does not reduce due to adding redundant copies. We use fluid limit arguments in order to prove the stability condition. See Section 2.4.2 for more details on the stability condition and the fluid-limit approximation. However we do not extend this result to any arbitrary work-conserving policy, as we will show through a counterexample in Section 3.2.2. Appendix 3.8.A contains the proofs of all results obtained in this section.

3.2.1 PS and ROS

In this section, we study the policies PS and ROS and prove that their stability condition is $\rho < 1$ under the i.i.d. copies assumption. An intuitive explanation for this result is the following. Under both PS and ROS, the average capacity of server s dedicated to type-c jobs at time t is given by $N_c(t)/M_s(t)$. Since copies are i.i.d., the departure rate of type-c jobs is given by the sum of the departure rates in the d servers (in the set c) the job is sent to, that is, $\mu\left(\sum_{\tilde{s}\in c}\frac{N_c(t)}{M_{\tilde{s}}(t)}\right)$. Now, summing over all job types that have a copy in server s, we obtain as total departure rate from server s,

$$\sum_{c\in\mathcal{C}(s)}\sum_{\tilde{s}\in c}\left(\frac{\mu N_c(t)}{M_{\tilde{s}}(t)}\right). \tag{3.1}$$

For a given time t, let s_{max} be the server containing the largest number of copies, i.e., $M_{s_{max}}(t) \geq M_s(t)$, for all s. We have the following lower bound on the departure rate from server s_{max}:

$$\sum_{c\in\mathcal{C}(s_{max})}\sum_{\tilde{s}\in c}\left(\frac{\mu N_c(t)}{M_{\tilde{s}}(t)}\right) \geq \frac{\mu\sum_{c\in\mathcal{C}(s_{max})}\sum_{\tilde{s}\in c}N_c(t)}{M_{s_{max}}(t)} = \frac{\mu d\sum_{c\in\mathcal{C}(s_{max})}N_c(t)}{M_{s_{max}}(t)} = \mu d.$$

The arrival rate of copies to a server equals $\frac{d}{K}\lambda$. If $\rho < 1$, then $\lambda\frac{d}{K} < \mu d$, hence the total arrival rate to a server with the largest number of copies is smaller than its departure rate. This allows us to prove that the fluid limit of server $s_{\max}$ goes to zero in finite time, hence implies stability.

The fluid-scaled system is defined as follows, see Section 2.4.2.1 for more details: For $r > 0$, denote by $N_c^{IID,r}(t)$ the system where the initial state satisfies $N_c^{IID}(0) = rn_c(0)$, for all $c \in \mathcal{C}$. The superscript IID refers to the system under either PS or ROS in the system with i.i.d. copies. Using standard arguments, see [26], we can write for the

fluid-scaled number of jobs per type

$$\frac{N_c^{IID,r}(rt)}{r} = n_c(0) + \frac{1}{r}\tilde{A}_c(rt) - \frac{1}{r}\tilde{S}_c(T_c^{IID,r}(rt)), \tag{3.2}$$

where $\tilde{A}_c(t)$ and $\tilde{S}_c(t)$ are independent Poisson processes having rates $\frac{\lambda}{\binom{K}{d}}$ and μ, respectively, and $T_c^{IID,r}(t) = \sum_{s\in c} T_{s,c}^{IID,r}(t)$, where $T_{s,c}^{IID,r}(t)$ is the cumulative amount of capacity spend on serving type-c jobs in server $s \in c$ during the time interval $(0, t]$.

In the following result, we obtain the general characterization of a fluid limit.

Lemma 3.2.1. *For almost all sample paths ω and sequence r_k, there exists a subsequence r_{k_j} such that for all $c \in \mathcal{C}$ and $t \geq 0$,*

$$\lim_{j\to\infty} \frac{N_c^{IID,r_{k_j}}(r_{k_j}t)}{r_{k_j}} = n_c^{IID}(t),\ u.o.c \quad \text{and} \quad \lim_{j\to\infty} \frac{T_c^{IID,r_{k_j}}(r_{k_j}t)}{r_{k_j}} = \tau_c^{IID}(t),\ u.o.c., \tag{3.3}$$

with $(n_c^{IID}(\cdot), \tau_c^{IID}(\cdot))$ continuous functions. In addition,

$$n_c^{IID}(t) = n_c(0) + \frac{\lambda}{\binom{K}{d}}t - \mu\tau_c^{IID}(t),$$

where $n_c^{IID}(t) \geq 0$, $\tau_c^{IID}(0) = 0$, $\tau_c^{IID}(t) \leq t$, and $\tau_c^{IID}(\cdot)$ are non-decreasing and Lipschitz continuous functions for all $c \in \mathcal{C}$.

The following lemma gives a partial characterization of the fluid process.

Lemma 3.2.2. *The fluid limit $m_s^{IID}(t) := \sum_{c\in\mathcal{C}(s)} n_c^{IID}(t)$ satisfies:*

$$\frac{\mathrm{d}m_s^{IID}(t)}{\mathrm{d}t} \leq \lambda\frac{d}{K} - \mu d, \quad \text{if} \quad m_s^{IID}(t) = \max_{l\in S}\{m_l^{IID}(t)\} > 0.$$

In the case $\rho < 1$, the drift in the above expression is strictly negative. That is, the maximum of the fluid process $\vec{m}(t)$ is strictly decreasing. Hence, there is a finite time T when the fluid process is empty. From this, we can directly conclude that the system is stable, see Theorem 2.4.10.

Proposition 3.2.3. *Under either PS or ROS with i.i.d. copies, the system is stable when $\rho < 1$.*

Remark 3.2.4. Stability condition for general scheduling policies: We believe that the above result holds for any non-preferential scheduling policy that treats all job types equally, but we did not succeed in obtaining a unifying proof. Our approach to prove Proposition 3.2.3 can be readily extended to cover all policies whose fluid drift is *(i)* continuous and *(ii)* is equal or larger than μd for the server(s) with the largest

number of copies. Both PS and ROS satisfy this property, but not FCFS. Given the lack of generality of the class of policies that satisfy *(i)* and *(ii)*, we chose to restrict the presentation to PS and ROS. We provide a further discussion on the stability of redundancy models with exponentially distributed i.i.d. copies in Chapter 8.

In Chapter 6, we will see that when one considers a nested redundancy topology, there do exist priority policies that are maximally stable. More precisely, we show there that a version of the LRF policy provides stability when implemented in a nested system.

Remark 3.2.5. Stability condition for general service requirement distributions: In this chapter we focus on exponential distributed service requirements. The analysis of general service requirement distributions is a very challenging problem and it will require a different proof technique. For instance, FCFS with i.i.d. copies has been studied in [82] for a specific choice of highly variable service requirements. For an asymptotic regime, the authors show that the stability region *increases without bound* as the service requirement becomes more variable and/or the number of redundant copies increases. This is explained by the fact that each job has d independent copies, and hence, in the (unlikely) event that a copy has a relatively large size, the probability that this copy will be served will become very small as the number of redundant copies increases, or the sizes of the copies become more variable, since the completion of a small-sized copy will directly cancel this large copy. Therefore, the combination of variable job sizes and redundancy, increases the stability region. In the case of PS, [83] proves that for generally distributes service times, the necessary stability condition is given by $\lambda d E(\min(X_1, \ldots, X_d))/K < 1$, where $X_1, \ldots, X_d$ denote the service times of the d i.i.d. copies of a job. Additionally, the authors show that the stability condition for service time distributions that are NWU (NBU) is larger (smaller) than that for exponential service time distributions ($\rho < 1$).

3.2.2 Priority policy

Given Proposition 3.2.3, one might wonder whether any work-conserving policy would be maximally stable when copies are i.i.d.. Indeed, whenever all servers have copies to serve, the total departure rate of jobs equals $K\mu$. This is however not enough to conclude for stability. In Example 3.2.6. we give a counterexample.

Example 3.2.6. We consider the system with $K = 3$ and $d = 2$, hence there are three different types of jobs: $\mathcal{C} = \{\{1,2\},\{1,3\},\{2,3\}\}$. In server 1, FCFS is implemented. In server 2 and server 3, jobs of types $\{1,2\}$ and $\{1,3\}$ have preemptive priority over jobs of type $\{2,3\}$, respectively. Additionally, within a type, jobs are served in order of arrival.

In Figure 3.1 we have plotted the trajectory of the system when $\rho = 0.96 < 1$.

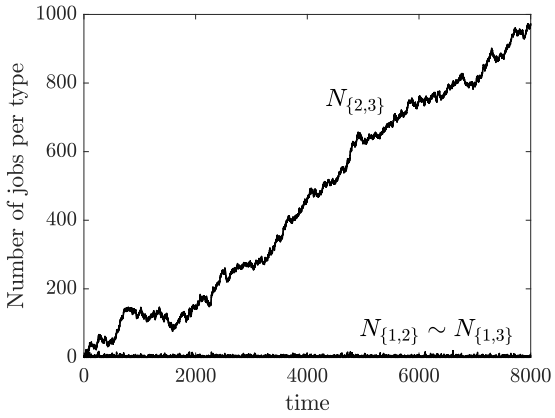


Figure 3.1 The trajectory of the number of jobs per type with time for the system with $\lambda = 2.9$.

One observes that the number of type-$\{2,3\}$ jobs in the system grows large, while the number of type-$\{1,2\}$ and type-$\{1,3\}$ jobs stay close to 0. Hence, the system is clearly unstable, even though $\rho < 1$. This is explained as follows: jobs of type-$\{1,2\}$ and jobs of type-$\{1,3\}$ are oblivious to the presence of jobs of type-$\{2,3\}$, due to the preemptive priority assumed in servers 2 and 3. Type-$\{1,2\}$ and type-$\{1,3\}$ jobs have only one server in common. Such a FCFS-redundancy system (M-model) has been analyzed in [46], where it was obtained that this system (and hence the number of type-$\{1,2\}$ jobs and type-$\{1,3\}$ jobs) is stable when $\rho = \frac{\lambda}{3\mu} < \frac{3}{2}$.

Type-$\{2,3\}$ jobs are served in server 2 (3) whenever there are no type-$\{1,2\}$ jobs (type-$\{1,3\}$ jobs) present in the system. Note that type-$\{1,2\}$ and type-$\{1,3\}$ jobs behave independent from type-$\{2,3\}$ jobs. Assuming type-$\{1,2\}$ and type-$\{1,3\}$ are in steady state, the drift of the number of type-$\{2,3\}$ jobs in the system is given by

$$\begin{aligned}\tfrac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}^{\vec{n}}[N_{\{2,3\}}(t)]\Big|_{t=0} = \;& \tfrac{\lambda}{3} - \mu P(N_{\{1,2\}}=0, N_{\{1,3\}}>0)\\ & - \mu P(N_{\{1,3\}}=0, N_{\{1,2\}}>0) - 2\mu P(N_{\{1,2\}}=0, N_{\{1,3\}}=0).\end{aligned}$$

By [46], we have that

$$\begin{aligned}P(N_{\{1,2\}}=0, N_{\{1,3\}}>0) = \;& P(N_{\{1,3\}}=0, N_{\{1,2\}}>0)\\ = \;& P(N_{\{1,3\}}=0, N_{\{1,2\}}=0)\left(\tfrac{2\mu}{2\mu-\lambda/3}-1\right),\end{aligned}$$

where $P(N_{\{1,3\}}=0, N_{\{1,2\}}=0) = \left(\frac{(2\mu-\lambda/3)^2(3\mu-2\lambda/3)}{4\mu^2(3\mu-2\lambda/3)+(\lambda/3)^2\mu}\right)$. After some algebra, we have

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}^{\vec{n}}[N_{\{2,3\}}(t)]\Big|_{t=0} = \frac{\lambda}{3} - 2\mu\left(\frac{(2\mu-\lambda/3)^2(3\mu-2\lambda/3)}{4\mu^2(3\mu-2\lambda/3)+(\lambda/3)^2\mu}\right)\left(\frac{2\mu}{2\mu-\lambda/3}\right).$$

It can be checked that the latter is strictly negative if and only if $\rho < 0.91$. From this one can conclude that the system is unstable when $\rho > 0.91$, using fluid scaling techniques. We however omit the proof.

3.3 The FCFS scheduling policy and identical copies

In this section we consider the redundancy-d model when copies of a job are identical and when FCFS is employed. We characterize the necessary and sufficient stability condition and show that the stability region is reduced when adding redundant copies. This as opposed to the i.i.d. copies case, for which the stability condition is independent of the number of copies d. Appendix 3.8.B contains the proofs of the results obtained in this section.

3.3.1 Characterization of stability condition

Due to the homogeneous capacities and identical copies assumption, if the copies in service in the K servers belong to k different jobs, the departure rate of the system is equal to $k\mu$. The latter is smaller than $K\mu$, even though K servers are busy. This follows from the observation below.

Observation 3.3.1. *At every instant of time when the system is not empty, the job that is longest in the system will be running on d servers.*

Since at every instant of time there is a subset of d servers giving service to all the copies of the same job and copies are identical, the total output rate of this subset of d servers is reduced to μ. Regarding the $K - d$ remaining servers, the order of arrivals of the jobs impacts the output rate of the remaining servers. As an example, when $K = 4$ and $d = 2$, the $K - d = 2$ remaining servers have as total output rate either μ (if copies of the same job are first in line in both servers) or 2μ. In total, this would give as total output rate either 2μ or 3μ. In both cases, it is strictly less than $K\mu = 4\mu$.

From the above, it is clear that the total departure rate is not order-independent, see condition (ii) in Definition 2.4.2. It depends on the order of arrivals of the jobs that are in service. Note that in the case of i.i.d. copies, the order independence property was key to obtain a product-form steady state distribution, see [18, 9]. In the case of identical copies (as considered here), the *lack* of the order independence assumption prevents us from obtaining a closed-form expression for the stability condition. Instead, in the main result of this section, we will characterize the stability condition through the average departure rate in a corresponding system with infinite backlog, referred to as the *saturated system*. Formally, the saturated system is defined as follows.

Definition 3.3.2. *Saturated system.* There is an infinite backlog of jobs waiting in the system, sampled uniformly over types. There are K servers and the scheduling policy within a server is FCFS. The d copies corresponding to a job are identical.

We denote the long-run time average number of distinct jobs served in the saturated system by $\bar{\ell}$. Hence, the total departure rate in a saturated system is $\bar{\ell}\mu$. Below we show that $\lambda < \bar{\ell}\mu$, or equivalently, $\rho < \bar{\ell}/K$, is a necessary and sufficient condition for the original FCFS system with identical copies to be stable. The characterization of the stability condition through a saturation system is reminiscent of the saturation rule obtained in [11] to prove stability of a large class of queuing systems. We can however not use their framework due to certain specifics of our model. Instead, in order to prove the stability condition, we resort to stochastic coupling, martingale arguments and fluid limits. The proof will be given in Section 3.3.2.

Proposition 3.3.3. *Under FCFS and identical copies, the system is stable if $\rho < \bar{\ell}/K$ and unstable if $\rho > \bar{\ell}/K$.*

We will prove in Section 3.4, that the stability condition under PS and identical copies is $\rho < 1/d$. Note that $\bar{\ell} \geq \lceil K/d \rceil$, since at least $\lceil K/d \rceil$ jobs are being served at a given time in the saturated system. This gives the following corollary.

Corollary 3.3.4. *The stability region under FCFS, $\rho < \bar{\ell}/K$, is larger than under PS, $\rho < 1/d$.*

In the remainder of this section, we characterize $\bar{\ell}$. In order to do so, we consider the Markovian state descriptor of the form $\vec{e} = (O_{\ell^*}, L_{\ell^*-1}, \ldots, O_2, L_1, O_1)$. Here, ℓ^* denotes the number of jobs that receive service in state $\vec{e}$ and O_j denotes the type of the j-th job in service. Furthermore, there are L_j jobs that arrived after job O_j and cannot be served since they are waiting for servers that are busy serving types $O_1, \ldots, O_j$. Note that the state descriptor $\vec{e}$ retains the order of the arriving jobs per type from right to left.

For a given state $\vec{e}$, we let $\ell^*(\vec{e})$ denote the number of jobs in service, i.e., ℓ^*. Let $\bar{E}$ denote the state space of the saturated system. The mean number of jobs in service can formally be written as

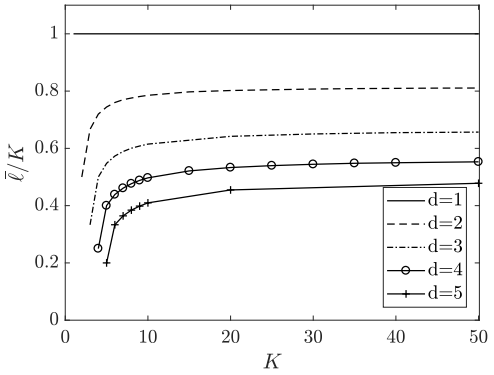
$$\bar{\ell} := \sum_{\vec{e} \in \bar{E}} \pi(\vec{e}) \ell^*(\vec{e}), \tag{3.4}$$

with $\pi(\vec{e})$ the steady-state distribution of the saturated system.

In general, no closed-form expression is known for $\bar{\ell}$. In Appendix 3.8.B, we write a general expression for the balance equations of the saturated system and state them explicitly for the case $d = K - 2$ (simplest non-trivial case, since then either two or three jobs are served in the saturated system). From this, we can obtain numerically

Figure 3.2 The table and figure show the values of $\bar{\ell}/K$ for different values of d and K.

$\bar{\ell}/K$	$K=2$	$K=3$	$K=4$	$K=5$	$K=6$	$K=7$	$K=8$
$d=1$	1	1	1	1	1	1	1
$d=2$	0.5	0.66	0.71	0.74	0.76	0.77	0.77
$d=3$		0.33	0.5	0.54	0.57	0.58	0.60
$d=4$			0.25	0.4	0.43	0.46	0.47
$d=5$				0.2	0.33	0.36	0.38
$d=6$					0.16	0.28	0.31
$d=7$						0.14	0.25



the value of $\bar{\ell}$. When $d \in \{1, K-1, K\}$, we can instead get closed-form expressions for $\bar{\ell}$. When $d = K-1$, there are d servers that process copies of one job, and the remaining $K - d = 1$ server serves one additional job, hence, $\bar{\ell} = 2$. When instead $d = 1$, there is no redundancy and each server serves one job in the saturated system, i.e., $\bar{\ell} = K$. When $d = K$, the system behaves as a single server with capacity μ, that is, $\bar{\ell} = 1$.

In Figure 3.2, we present $\bar{\ell}/K$ for different values of d and K: the table (*left*) shows $\bar{\ell}/K$ for small values of K and the figure (*right*) plots the value of $\bar{\ell}/K$ as K grows large. To obtain the value of $\bar{\ell}$ for $d \neq 1, K-2, K-1, K$, we simulated the saturated system, rather than solving the balance equations. We note from Figure 3.2 that $\bar{\ell}/K$ (and hence the stability region) increases when the number of servers (K) grows large. We make this formal in the proposition below, which is proved using stochastic coupling arguments.

Proposition 3.3.5. *For the saturated system, it holds that $\bar{\ell}/K$ is increasing in K.*

It would be interesting to determine $\lim_{K\to\infty} \bar{\ell}/K$, as this would represent the stability condition in a mean-field setting. The values in the figure at Figure 3.2 seem to indicate that $\lim\limits_{K\to\infty} \bar{\ell}/K = c$ with $c < 1$, that is, also in the mean-field limit, the stability region reduces as compared to $d = 1$. We observed that this value c coincides with the value obtained by the numerical method developed in [58].

From Figure 3.2 we further observe that $\bar{\ell}/K$ decreases when the number of redundant copies (d) increases. Unfortunately, we did not succeed in finding an argument to prove this property. In Chapter 8 we draw two conjecture after these two observations.

3.3.2 Proof of the stability condition

In this section we show that $\rho < \bar{\ell}/K$ is both a necessary and sufficient stability condition, that is, we prove Proposition 3.3.3. The dependency on the order of arrivals

of the total departure rate makes exact analysis hard. In order to prove the stability condition, we formulate two auxiliary systems that we can compare sample-path wise to the original system. These systems will have the property that for a sufficiently large period of time, a saturated system is observed, and hence, have as average departure rate $\bar{\ell}\mu$, which allows us to prove the stability condition. See Section 2.4.2 for more details on the stability condition. The full proofs can be found in Appendix 3.7.B.

3.3.2.1 Necessary stability condition

The auxiliary process $\vec{N}^{(T)}(t)$ is defined as follows. At time $t = 0$, we assume that $\tilde{A}_c(T)$ type-c jobs arrive, $\forall c \in \mathcal{C}$. During the interval $(0, T]$ there are no further arrivals. After time $t > T$, new type-c jobs arrive according to the original Poisson process with rate $\lambda/\binom{K}{d}$. In the $\vec{N}^{(T)}$-system, each server serves according to FCFS.

The reason why the auxiliary system is a lower bound can be explained as follows. Under the auxiliary system, all the jobs that in the original system are planed to arrive up to time $T > 0$, are already present at time $t = 0$. Therefore, under FCFS, these jobs will be served in the auxiliary system no later than in the original one.

To compare the auxiliary process with the original FCFS system, we need to introduce some notation. The attained service of the copy of the i-th type-c job in server s, $a_{cis}^{FCFS}(t)$, will be compared to the attained service of the same copy in the $\vec{N}^{(T)}$-system. For that, (with slight abuse of notation), we let $a_{cis}^{\vec{N}^{(T)}}(t)$ denote the attained service of *this same* copy, where we assume that in case this copy has already departed in the $\vec{N}^{(T)}$-system, then $a_{cis}^{\vec{N}^{(T)}}(t)$ is set equal to its service requirement b_{ci}. In the result below we show that sample-path wise, a job departs earlier in the $\vec{N}^{(T)}$-system than in the original system. In particular, this implies that if the original FCFS model is stable, then the $\vec{N}^{(T)}$-system is stable as well.

Lemma 3.3.6. *Assume* $N_c^{FCFS}(0) = N_c^{(T)}(0)$ *and* $a_{cis}^{FCFS}(0) = a_{cis}^{\vec{N}^{(T)}}(0)$*, for all* c, i, s*. Then,* $N_c^{(T)}(t) \leq N_c^{FCFS}(t) + (\tilde{A}_c(T) - \tilde{A}_c(t))^+$ *and* $a_{cis}^{FCFS}(t) \leq a_{cis}^{\vec{N}^{(T)}}(t)$*, for all* $i = 1, \ldots, N_c^{FCFS}(t)$*,* $c \in \mathcal{C}$*,* $s \in S$*.*

Let the random variable $\tau(T) > 0$ denote the moment that one of the servers becomes empty. In the time interval $[0, \tau(T)]$, the $\vec{N}^{(T)}$-system will behave as a saturated system. We will prove that as T grows large, $\tau(T)$ grows large, and due to the law of large numbers, the time-average number of jobs in service in the interval $[0, \tau(T)]$ will be equal to $\bar{\ell}$, as defined in Eq. (3.4). Since each job in service has a departure rate μ, this allows us to prove that if the $\vec{N}^{(T)}$-system is stable, then $\lambda < \bar{\ell}\mu$. Together with Lemma 3.3.6 this gives the following result.

Proposition 3.3.7. *Under FCFS and identical copies, the system is unstable if* $\rho > \bar{\ell}/K$*.*

3.3.2.2 Sufficient stability condition

In order to prove that $\rho < \bar{\ell}/K$ is a sufficient stability condition, we define the process $\vec{N}^{(0)}(t)$ as follows. In the time interval $[0, |\vec{N}^{(0)}(0)|/\mu]$, only those jobs that were already present at time 0 are allowed to be served (according to FCFS). From time $t, t \geq |\vec{N}^{(0)}(0)|/\mu$ onwards, all jobs present in the system can be served. We note that the present system is new compared to the $\vec{N}^{(T)}$-system in Section 3.3.2.1 and is not derived by fixing $T = 0$.

In $\vec{N}^{(0)}$-system, all the jobs that in the original system arrived between time $t = 0$ and $t = |\vec{N}^{(0)}(0)|/\mu$, can not be served until time $t = |\vec{N}^{(0)}(0)|/\mu$. From that moment on, all present jobs are again served in order of arrival. Therefore, under FCFS, these jobs will be served in the original system no later than in the $\vec{N}^{(0)}$-system.

We first establish a sample-path comparison with the original FCFS system, which allows us to conclude for stability of the original process. We let $a_{cis}^{\vec{N}^{(0)}}(t)$ denote the attained service of the i-th type-c job in the $\vec{N}^{(0)}$-system. The attained service of the i-th type-c job in server s in the $\vec{N}^{(0)}$-system will be compared to the attained service of the same copy in the FCFS system. In order to do so, with a slight abuse of notation, we let $a_{cis}^{FCFS}(t)$ denote the attained service of *this same* copy, where we assume that in case this copy has already departed in the FCFS-system, then it is set equal to its service requirement b_{ci}.

Lemma 3.3.8. *Assume $N_c^{FCFS}(0) = N_c^{(0)}(0)$ and $a_{cis}^{FCFS}(0) = a_{cis}^{\vec{N}^{(0)}}(0)$, for all c, i, s. Then, $N_c^{(0)}(t) \geq N_c^{FCFS}(t)$ and $a_{cis}^{\vec{N}^{(0)}}(t) \leq a_{cis}^{FCFS}(t)$, for all $i = 1, \ldots, N_c^{(0)}(t), c \in \mathcal{C}, s \in S$.*

For the stochastic process $\vec{N}^{(0)}(\cdot)$, we will see that the system is stable if $\rho < \bar{\ell}/K$. To do so, we will characterize the fluid limit. We will show that at the moment the auxiliary process can start serving jobs that were not present at time 0, the queue has built up, and during a considerable amount of time the system will behave as a saturated system. Hence, the average number of occupied servers equals $\bar{\ell}$, which allows us to prove that $\vec{N}^{(0)}(t)$ is stable if $\rho < \bar{\ell}/K$. Together with Lemma 3.3.8 this gives the following result.

Proposition 3.3.9. *Under FCFS and identical copies, the system is stable if $\rho < \bar{\ell}/K$.*

3.4 The PS scheduling policy and identical copies

In this section, we show that $\rho < 1/d$ is the stability condition for the redundancy-d model with identical copies when PS is employed in all the servers.

Proposition 3.4.1. *Under PS and identical copies, the system is stable if $\rho < \frac{1}{d}$ and unstable if $\rho > \frac{1}{d}$.*

Remark 3.4.2. Stability under general service time distributions: Our result, which appeared in [SR1], holds for exponential service requirements. In [83], Raaijmakers et al. extended this result. Using fluid limit arguments, the authors showed that $\rho < 1/d$ is a necessary stability condition when the service times follow a general distribution and copies have a general correlation structure. In Chapter 4, we extend the stability result of Proposition 3.4.1 to a general redundancy topology, and generally distributed service times with identical copies.

Before proceeding to the intuition (Section 3.4.1) and proof of Proposition 3.4.1 (Section 3.4.2), we first characterize the instantaneous departure of a job in the system.

Under PS, the attained service of the copy of the i-th type-c job in server s increases at speed $\mu/M_s^{PS}(t)$, that is, $\frac{\mathrm{d}a_{cis}^{PS}(t)}{\mathrm{d}t} = \frac{\mu}{M_s^{PS}(t)}$, $\forall c \in \mathcal{C}(s)$, $i = 1, \ldots, N_c(t)$. Note that a departure of a job is due to a departure in the server where it has the largest attained service. Denote by $s_{ci}^*(t)$ the server that contains the copy of the i-th type-c job with the largest attained service, that is, $s_{ci}^*(t) := \arg\max_{s \in c}\{a_{cis}^{PS}(t)\}$, for all $c \in \mathcal{C}$, $i = 1, \ldots, N_c(t)$. The instantaneous departure rate of the i-th type-c job under PS is hence $\frac{\mu}{M_{s_{ci}^*(t)}^{PS}(t)}$. In particular, the number of type-c jobs decreases at rate

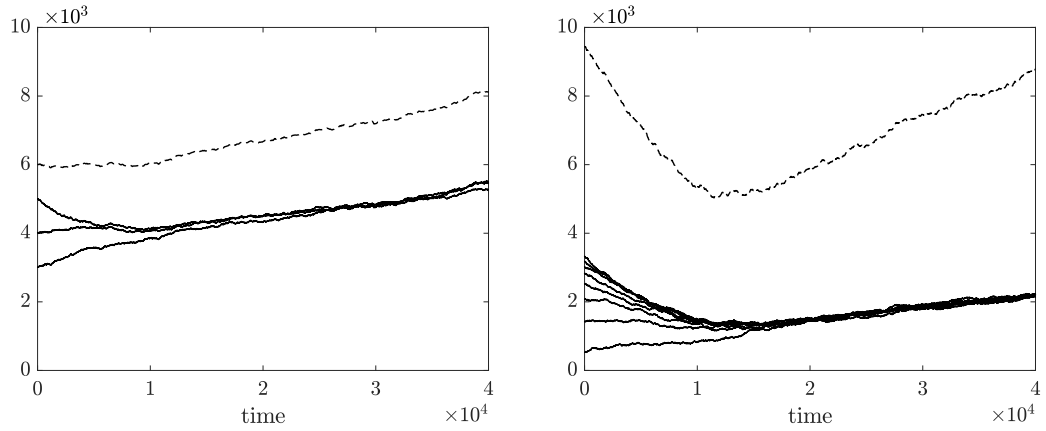
$$\sum_{i=1}^{N_c^{PS}(t)} \frac{\mu}{M_{s_{ci}^*(t)}^{PS}(t)}. \tag{3.5}$$

3.4.1 Intuition behind stability condition and its proof

To illustrate why $\rho < 1/d$ is the stability condition, we have plotted in Figure 3.3 the trajectories of the number of copies in each of the servers, $M_s^{PS}(t)$, for two settings, $K = 3$ and $K = 8$, with $d = 2$. In both cases, we assume the load is such that $\rho > 1/d$. We let the system start in a very large state, and plot the trajectories over a large time horizon.

In Figure 3.3, we observe the following effect. When the processes $M_s^{PS}(t)$ are unbalanced (as is the case for $t < 10^4$), the number of copies at the most loaded servers decrease. Consider one of the highly-loaded servers, referred to as server $\tilde{s}$. Now, a copy in service in server $\tilde{s}$ will leave because either it has obtained full service in server $\tilde{s}$, or a copy of the same job finished service in another *less-loaded* server. The rate at which a copy is served at such a less-loaded server, is higher than that in the high-loaded server $\tilde{s}$. Thus, the effective departure rate of copies from server $\tilde{s}$ will be higher than μ. Since the arrival rate of new copies to a given server equals $\lambda\frac{d}{K}$, this explains why the number of copies in server $\tilde{s}$ (a higher loaded server) can go down, even though $\lambda d/K > \mu$ ($\rho > 1/d$).

Hence, during a certain time, the system experiences a "good phase" in which



(a) $K = 3$, $d = 2$ and $\rho = 0.53$ (b) $K = 8$, $d = 2$ and $\rho = 0.52$.

Figure 3.3 Trajectory of the number of jobs in the system (dashed lines) and number of copies per server (solid lines) with respect to time under PS with identical copies.

higher-loaded servers decrease and the total queue length decreases as well. However, once the servers are more equally loaded, we observe that the total queue length starts to build up. To explain this, consider the symmetric case, i.e., $M_s^{PS}(t) = m$, for all s. Then, each copy of a job receives in each server the same fraction of capacity. Hence, the departure rate of copies from a server is μ (see Eq. (3.5)). Since $\lambda d/K > \mu$, the servers will build up from then on, and the total number of jobs will diverge.

In order to prove the stability condition, the challenge is to prove instability. We note that the total number of jobs cannot be taken as Lyapunov function: As we described above, inside some cone around the diagonal (symmetric states), the drift of the total number of copies in the system is strictly positive, while outside that cone, the drift of the total number of jobs is decreasing. We further observe from Figure 3.3 that the drift of the server with the minimum number of copies is strictly positive, while the drifts of the higher-loaded servers is first negative, until they join the minimum, from which point on they stay together and increase. This motivated us to study the drift of the server with the minimum number of copies. Though it is a complicated (non-monotone) function for the stochastic process, one can show that for the fluid limit, the drift of the server with the minimum number of copies is strictly positive. So, even if at a short time scale, the minimum cannot be taken as Lyapunov function, the minimum at a fluid scale does go up if $\rho > 1/d$. This is exactly what is used in order to prove unstability, see Lemma 3.4.5.

3.4.2 Proof of stability condition

Having identical copies makes exact analysis hard, as it requires to keep track of the attained service of the copies in each of the servers. In order to derive the necessary and sufficient stability condition, i.e. to prove Proposition 3.4.1, we describe two systems that lower and upper bound the original PS system. These systems will have the property that the departure rate no longer depends on the attained service. The latter allows us to prove using fluid limit arguments the necessary (sufficient) condition for stability for the lower bound (upper bound) system and hence, also for the original system. See Section 2.4.2 for more details on the stability condition and the fluid-limit approximation. The full proofs can be found in Appendix 3.7.C.

3.4.2.1 Necessary stability condition

For the original system, the departure rate of the number of type-c jobs depends on the attained service, see Eq. (3.5). More precisely, the departure rate of the i-th type-c job equals

$$\frac{\mu}{M^{PS}_{s^*_{ci}(t)}(t)}, \tag{3.6}$$

where we recall that $s^*_{ci}(t)$ denotes the server where a copy of this job has received most service so far. The Lower Bound (LB) system is defined as follows: We replace Eq. (3.6) by

$$\frac{\mu}{M^{PS}_{s^{min}_c(\vec{N}(t))}(t)},$$

where $s^{min}_c(\vec{N}(t)) := \arg\min_{s\in c}\{M_s(t)\}$ is the server with the least number of copies that contains a type-c job at time t (ties are broken at random). That is, in the LB-system, a type-c job receives service from the server in the set c with the minimum number of copies. We note that since the LB-system does no longer depend on the attained service, it is more amenable to get the stability condition.

The LB-system is described by $\{N^{LB}_c(t), c \in \mathcal{C}\}_{t\geq 0}$, living on the countable set $(\mathbb{N}\cup\{0\})^{\binom{K}{d}}$. Here, $N^{LB}_c(t)$ denotes the number of type-c jobs in the LB-system. The process $N^{LB}_c(t)$ increases by one at rate $\lambda/\binom{K}{d}$ (as is the case for the original process), and decreases by one at rate

$$\mu\frac{N^{LB}_c(t)}{M^{LB}_{s^{min}_c(\vec{N}^{LB}(t))}(t)}, \tag{3.7}$$

where $M^{LB}_s = \sum_{c\in\mathcal{C}(s)} N^{LB}_c$. Note that Eq. (3.7) coincides with Eq. (3.5), where now $s^*_{ci}(t)$ is replaced by $s^{min}_c(\vec{N}(t))$ (because for a given type, all jobs share the same server with the smallest number of copies). Below, we prove that this system gives a stochastic

lower bound for the original system.

Lemma 3.4.3. *Assume $N_c^{PS}(0) = N_c^{LB}(0)$, for all c. Then, $N_c^{PS}(t) \geq_{st} N_c^{LB}(t)$, for all $c \in \mathcal{C}$ and $t \geq 0$.*

In Lemma 3.4.4 below, we give an expression for the departure rate from a server s in the LB-system. Before doing so, we need to introduce some notation. For each server s, we define $D_s(\vec{N}^{LB}(t)) := \{l \in S : M_s^{LB}(t) \geq M_l^{LB}(t)\}$. We denote by $\mathcal{C}_l^s(\vec{N}(t)) := \{c \in \mathcal{C}(s) : l = s_c^{min}(\vec{N}(t))\}$, the subset of types that are served in server s and for whom server l is the server with the minimum number of copies that serve type c. Notice that $\mathcal{C}(s)$ is the disjoint union of the above elements, $\mathcal{C}(s) = \cup_{l \in D_s(\vec{N}(t))} \mathcal{C}_l^s(\vec{N}(t))$.

Lemma 3.4.4. *For the LB-system, when in state $\vec{N}^{LB}(t) = \vec{n}^{LB}$, the number of copies in server s, $M_s^{LB}(t)$, decreases by one at rate*

$$\mu \left(1 + \sum_{l \in D_s(\vec{n}^{LB})} \frac{(M_s^{LB}(t) - M_l^{LB}(t)) \sum_{c \in \mathcal{C}_l^s(\vec{n})} N_c^{LB}(t)}{M_s^{LB}(t) M_l^{LB}(t)} \right). \tag{3.8}$$

In particular, from Eq. (3.8) we clearly see the improvement brought by redundancy. For the server with the minimum number of copies, Eq. (3.8) simplifies to μ. This server is hence not receiving any help from the other (higher-loaded) servers. However, servers that do not have the minimum number of copies, do benefit from redundant copies, as their service rate is μ plus some additional positive fractions. This is due to the fact that in the LB-system, all types in server s that also have a copy in another server with less copies, will receive as effective service rate that what they would get in this latter server, and hence receive a higher capacity than what they would get in server s.

We study the fluid limit of the lower bound system in order to conclude the LB-system is transient when $\rho > 1/d$. The fluid-scaling consists in studying the rescaled sequence of systems indexed by parameter r, see Section 2.4.2.1 for more details. For $r > 0$, denote by $N_c^{LB,r}(t)$ the system where the initial state satisfies $N_c^{LB}(0) = rn_c(0)$, for all $c \in \mathcal{C}$. The associated number of copies per server is given by $M_s^{LB,r}(t) = \sum_{c \in \mathcal{C}(s)} N_c^{LB,r}(t)$, for all $s \in S$. For the fluid-scaled number of jobs per type we can write

$$\frac{N_c^{LB,r}(rt)}{r} = n_c(0) + \frac{1}{r}\tilde{A}_c(rt) - \frac{1}{r}\tilde{S}_c(T_c^{LB,r}(rt)), \tag{3.9}$$

where $T_c^{LB,r}(t)$ is defined as the cumulative amount of capacity spent on serving type-c jobs in server $s_c^{min}(\vec{N}^{LB,r}(\cdot))$ during the time interval $(0, t]$. The existence of fluid limits can be proved as before: The statement of Lemma 3.2.1 indeed directly translates to the process $\vec{N}^{LB,r}(t)$, and is therefore left out. In the following result, we obtain the general characterization of a fluid limit.

Lemma 3.4.5. *The fluid limit* $m_s^{LB}(t) := \sum_{c\in\mathcal{C}(s)} n_c^{LB}(t)$ *satisfies:*

$$\frac{\mathrm{d}m_s^{LB}(t)}{\mathrm{d}t} = \lambda\frac{d}{K} - \mu, \quad \textit{if} \quad m_s^{LB}(t) = \min_{l\in S}\{m_l^{LB}(t)\} > 0,$$

and

$$\frac{\mathrm{d}m_s^{LB}(t)}{\mathrm{d}t} \geq \lambda\frac{d}{K} - \mu, \quad \textit{if} \quad m_s^{LB}(t) = \min_{l\in S}\{m_l^{LB}(t)\} = 0.$$

Remark 3.4.6. *In case* $\lambda d/K - \mu > 0$*, this partial characterization of the fluid limit implies the following. Consider servers whose amount of fluid is the minimum, that is, consider servers belonging to the set* $U(t) := \{s \in S : m_s^{LB}(t) \leq m_{\tilde{s}}^{LB}(t), \forall \tilde{s}\}$*. By Lemma 3.4.5, the amount of fluid in these servers increases with a strictly positive rate* $\lambda d/K - \mu$*. Moreover, if at time* $t_0 > t$*, some server* $\tilde{s}$ *is added to this set, that is,* $U(t_0) = U(t) \cup \{\tilde{s}\}$*, this server will increase as well from that moment on with the same rate* $\lambda d/K - \mu$*.*

This uniform divergence of the fluid limit, together with bounds on the macroscopic drifts, allows us to show instability of the stochastic process $\vec{N}^{LB}(t)$ via a usual transience criterion for Markov chains whenever the fluid drift $\lambda d/K - \mu$ is strictly positive. Together with Lemma 3.4.3, this allows us to prove the following result.

Proposition 3.4.7. *Under PS and identical copies, the system is unstable if* $\rho > 1/d$*.*

3.4.2.2 Sufficient stability condition

For the original system, a job departs the system once a copy has received its service in one of the servers. We will now upper bound this, by considering the same system, but where a job departs from the system only if *all its copies* have completed service.

The Upper Bound (UB) system is defined as follows: we let $N_c^{UB}(t)$ denote the number of type-c jobs, and $A_c^{UB}(t) = (a_{cis}^{UB}(t))_{is}$, with $a_{cis}^{UB}(t)$ the attained service of the i-th type-c job in server s. Note that the number of copies in server s is given by $M_s^{UB}(t) = \sum_{c\in\mathcal{C}(s)} \sum_{i=1}^{N_c^{UB}(t)} \mathbf{1}_{(a_{cis}^{UB}(t)<b_{ci})}$, since a copy is only present in server s when $a_{cis}^{UB}(t)$ is strictly smaller than the service requirement b_{ci}. The i-th type-c job in server s is served at speed $\mu/M_s^{UB}(t)$, hence $\frac{\mathrm{d}a_{cis}^{UB}(t)}{\mathrm{d}t} = 1/M_s^{UB}(t)$. Now, the i-th type-c job departs from the system once $a_{cis}^{UB}(t) = b_{ci}$ for all servers $s \in c$, that is, when all the copies of a job are fully served.

To compare the UB-system with the original PS system, we need to compare the attained service of the i-th arrived job in both systems. For that, we denote by $\alpha_{i,s}^{UB}(t)$ and $\alpha_{i,s}^{PS}(t)$ the attained service of the i-th arrived job in server s, for UB and PS, respectively. With slight abuse of notation, we set $\alpha_{i,s}^{PS}(t)$ equal to β_i (the service requirement of the i-th arrived job) for all servers s, in case it has departed from the PS system.

Lemma 3.4.8. *Assume $\alpha_{i,s}^{PS}(0) = \alpha_{i,s}^{UB}(0)$, for all $i = 1, \ldots,$ and $s \in \mathcal{S}$. Then, $\alpha_{i,s}^{UB}(t) \leq_{st} \alpha_{i,s}^{PS}(t)$ for all $t \geq 0$, $i = 1, \ldots,$ and $s \in c$. In particular, $N_c^{UB}(t) \geq_{st} N_c^{PS}(t)$.*

In the UB-system, all copies need to be served until a job departs. Hence, each queue receives copies at rate $\lambda d/K$ and copies depart at rate μ, that is, its marginal distribution is that of an $M/M/1$ system with arrival rate $\lambda d/K$ and departure rate μ. The latter is positive recurrent if and only if $\rho < 1/d$. Since $\vec{N}^{UB}(t)$ serves as an upper bound for our model (Lemma 3.4.8), this implies that the original system is positive recurrent as well, as stated in the result below.

Proposition 3.4.9. *Under PS and identical copies, the system is stable if $\rho < 1/d$.*

3.5 The ROS scheduling policy and identical copies

In this section we study ROS with identical copies and show that the stability condition is $\rho < 1$. Appendix 3.8.D contains the proofs of the results obtained in this section.

3.5.1 Intuition behind stability condition and its proof

Under ROS with identical copies, an idle server chooses uniformly at random a new copy from its queue and serves it until the copy finishes service, or one of its identical copies finishes service in another server. Note that if k servers are serving different jobs, then the total departure rate of these k servers is μk. If however these k servers are serving a copy from the same job, then these k servers give together a total departure rate μ (since copies are identical), hence capacity is wasted.

From the above, we observe that $\mathbb{P}$(every copy in service belongs to a unique job) is an important measure to determine the stability condition under ROS. Note that this probability is strictly smaller than 1 when the queue length is small, hence capacity is wasted. However, as the queues grow large, this probability will converge to 1, showing that under the fluid scaling no capacity is wasted. This then allows us to conclude that the stability condition is not reduced when adding redundant copies, that is, $\rho < 1$ is the stability condition.

3.5.2 Proof of stability condition

In order to prove the stability, we investigate the fluid-scaled system. For $r > 0$, denote by $N_c^{ROS,r}(t)$ the system where the initial state satisfies $\vec{N}^{ROS}(0) = r\vec{n}(0)$. Using routing arguments, we can write

$$\frac{N_c^{ROS,r}(rt)}{r} = n_c(0) + \frac{1}{r}\tilde{A}_c(rt) - \frac{1}{r}\tilde{S}_c(T_c^{ROS,r}(rt)), \tag{3.10}$$

where $T_c^{ROS,r}(t)$ is defined as the cumulative amount of capacity spent on serving *a first copy* of type-c jobs in the interval $(0,t]$. For a given job, we refer with "first copy" to that copy (out of the d) that was first to enter into service.

The existence of the fluid limit can be proved. In fact, the statement of Lemma 3.2.1 and its proof directly carry over, and are therefore left out.

For a given server s, fixed time $t \geq 0$ and state $\vec{N}^{ROS}(t) = \vec{n}$, such that $\sum_{c\in\mathcal{C}(s)} n_c > 0$, we denote by $P_s(\vec{n})$ the probability that at time t this server is serving a copy that is not in service in any other server. Then, the following lemma is true.

Lemma 3.5.1. *For any server $s \in S$ and $\vec{N}^{ROS,r}(0) = r\vec{n}^r$, such that $\lim_{r\to\infty} \sum_{c\in\mathcal{C}(s)} rn_c^r > 0$, then*

$$\lim_{r\to\infty} P_s(r\vec{n}^r) = 1. \tag{3.11}$$

Thus, in the fluid limit, each server serves a copy of a job that is not being served at any other server in the system. We note, that this property holds since the scheduling policy is ROS, and does not depend neither on the redundancy topology nor the copies correlation nor the service time distribution in the servers.

The following lemma gives a partial characterization of the fluid process.

Lemma 3.5.2. *The fluid limit $m_s^{ROS}(t) := \sum_{c\in\mathcal{C}(s)} n_c^{ROS}(t)$ satisfies the following:*

$$\frac{\mathrm{d}m_s^{ROS}(t)}{\mathrm{d}t} \leq \lambda\frac{d}{K} - \mu d, \quad \textit{if} \quad m_s^{ROS}(t) = \max_{l\in S}\{m_l^{ROS}(t)\} > 0.$$

In case $\rho < 1$, the drift in the above expression is strictly negative. That is, the maximum of the fluid process $\vec{m}(t)$ is strictly decreasing. Hence, there is a finite time T when the fluid process is empty. From this, we can directly conclude stability (same steps as in the proof of Proposition 3.2.3).

Proposition 3.5.3. *Under ROS with identical copies, the process $\vec{N}^{ROS}(t)$ is ergodic when $\rho < 1$.*

3.6 Numerical analysis

We have implemented a simulator in order to assess numerically the impact of redundancy in a redundancy-d system. We run these simulations for a sufficiently large number of busy periods (10^6), so that, the variance and confidence intervals of the mean number of jobs in the system are sufficiently small.

We simulate the system under the same assumptions as considered in the theoretical results, that is, exponential service times and homogeneous servers. In order to assess the impact of our modeling assumptions, we also simulate the queuing model with other service time distributions, such as deterministic services, or degenerate hyperexponential

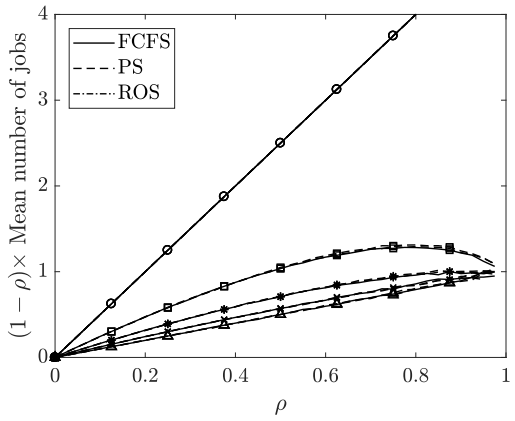


Figure 3.4 Mean number of jobs for the redundancy-d system ($K = 5$) with exponential service times and i.i.d. copies vs. the load for FCFS, PS and ROS service policies, and for $d = 1$ ($\circ$), $d = 2$ ($\square$), $d = 3$ ($*$), $d = 4$ ($\times$) and $d = 5$ ($\triangle$).

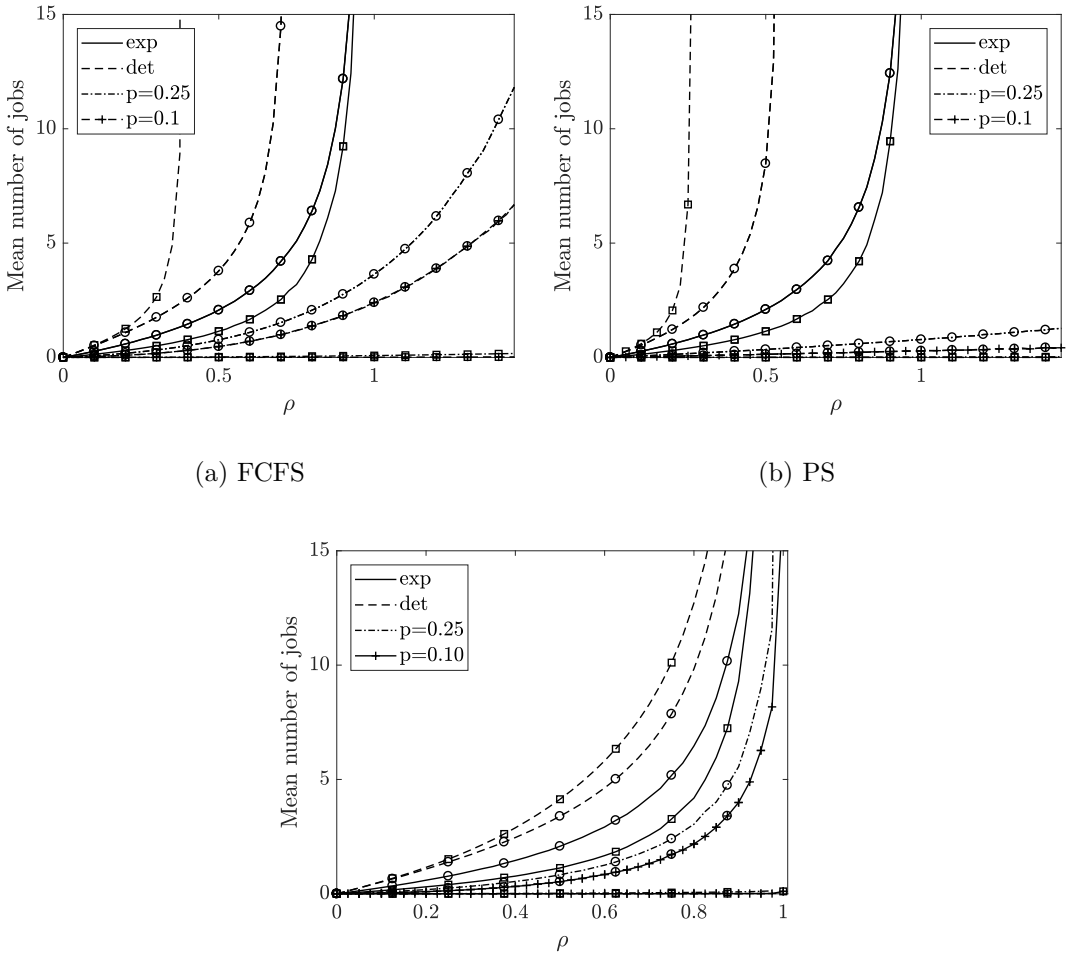
distributions. Under the latter distribution, with probability p the service requirement is exponentially distributed with parameter μp, and is 0 otherwise, hence, the mean service time equals $1/\mu$ (independent of p). The squared coefficient of variation however equals $\frac{2}{p} - 1$, which increases as p decreases. As a consequence, this distribution allows us to study the impact of the service time variability on the performance.

Without loss of generality, throughout this section we assume that the mean service requirement of a copy equals 1. In Section 3.6.1 we present the numerics for i.i.d. copies and in Section 3.6.2 for identical copies.

3.6.1 IID copies

In this section we consider that copies are i.i.d. copies. Under FCFS, PS and ROS, the system is stable whenever $\rho < 1$. In Figure 3.4 we plot the mean number of jobs (scaled by $1 - \rho$) under these policies for different values of d and exponentially distributed service times. For a given d, we observe that the plots under FCFS, PS and ROS are very similar. In addition, we observe that increasing the number of redundant i.i.d. copies, d, improves the performance.

In Figure 3.5 we plot the mean number of jobs under FCFS, PS, and ROS for exponential, deterministic, and degenerate hyperexponential ($p = 0.25$ and $p = 0.1$) service time distributions. We assume $K = 5$ servers and plot the performance for $d = 2$ and $d = 4$ copies. For either FCFS, PS or ROS, we can draw similar qualitative observations: *(i)* For variable service distributions, the performance improves as d increases while it is the other way around for deterministic copies, and *(ii)* for a given d, the performance improves as the variability of the service time distribution increases.



(a) FCFS

(b) PS

(c) ROS

Figure 3.5 Mean number of jobs for the redundancy-d system ($K = 5$) with $d = 2$ ($\circ$) and $d = 4$ ($\square$), for exponential, deterministic and degenerate hyperexponential ($p = 0.25$ and $p = 0.1$) service times (i.i.d. copies) vs. the load.

Only under FCFS and PS, with the degenerate hyperexponential distribution the system remains stable even if $\rho > 1$, while the stability region with deterministic service requirements seems to be reduced. In general, this can be intuitively explained by noting that when copies of a job are i.i.d. copies, the probability that a job departs due the completion of a rather large copy will become small as the variability in the copies increases. For PS, the increase in performance due to variability of service sizes is even more profound than with FCFS (see also Figure 3.5), as only jobs that have a positive service time for their d copies will enter service (which happens with probability p^d), while all other jobs are served instantaneously. We further note that (i) has been proved for PS in [83], where it was shown that for NWU (NBU) service times, the stability

region increases (decreases) as a function of d. Point (ii) has been proved for FCFS in an asymptotic region in [82].

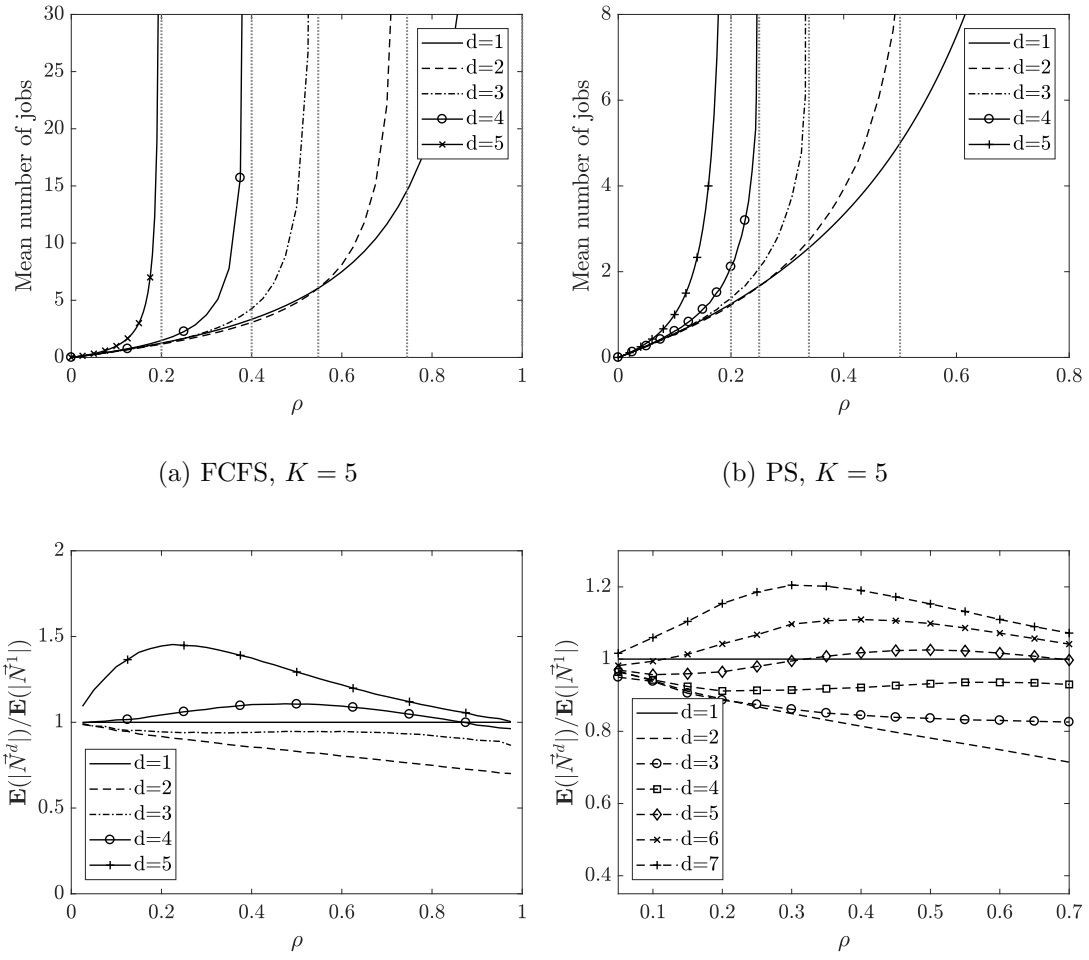
Under ROS, simulation results seem to indicate that the stability condition remains $\rho < 1$ for any service time distributions. Since copies are being randomly chosen for service, the system does not seem to profit from the variability of the service times of the i.i.d. copies. For example, in the case of degenerate hyperexponential distribution, when ρ is close to 1, with probability p a job will start being served in a server where its copy has strictly positive service requirement. Due to ROS, in a high congested system, the probability that another copy (possible of size 0) of this job will receive service in another server will be close to zero. Hence, with probability p a job needs exponential service time with parameter μp and with probability $1 - p$ its service time equals zero. On average it needs $1/\mu$, which explains why $\rho < 1$ can be the stability condition. Further note that for $d = 4$, it seems that the system remains stable when $\rho = 1$. This is however not the case. For ρ close to 1, the mean number of jobs in the system is close to zero, which can be explained as follows: A job has a strictly positive service requirement with probability p^d. When $d = 4$ and $p \in \{0.25, 0.1\}$, it holds that $p^4 < 10^{-2}$. Hence, it is very likely that for a very long time, zero jobs are present in the system (as all arriving jobs spend zero time in the system). This explains why for $\rho < 1$, the mean number of jobs stays very close to zero. We however believe that the stability condition is $\rho < 1$.

3.6.2 Identical copies

In this section we consider jobs with identical copies. We have proved that the stability condition strongly depends on the employed scheduling policy and on the number of copies d. In Section 3.6.2.1 we evaluate the system for different values of d and observe that the stability region reduces as d grows large. In Section 3.6.2.2 we characterize the performance and its dependence on d for a light-load regime, and observe that when the load is *small enough*, redundancy can improve the performance. In Section 3.6.2.3, we numerically study the impact of different service time distributions on the performance.

3.6.2.1 Exponential service time distributions

In Figure 3.6 we plot the mean number of jobs under (a) FCFS ($K = 5$), (b) PS ($K = 5$), (c) ROS ($K = 5$) and (d) ROS ($K = 8$) respectively, for different values of d. The vertical lines in Figure 3.6 (a) and (b) correspond to the stability regions (for different values of d) as derived in Proposition 3.3.3 (where $\bar{\ell}/K$ has been obtained via simulation of the saturated system) and Proposition 3.4.1. Indeed, we observe that the mean numbers of jobs under FCFS and PS have an asymptote at the point $\bar{\ell}/K$ and



(a) FCFS, $K = 5$

(b) PS, $K = 5$

(c) ROS, $K = 5$

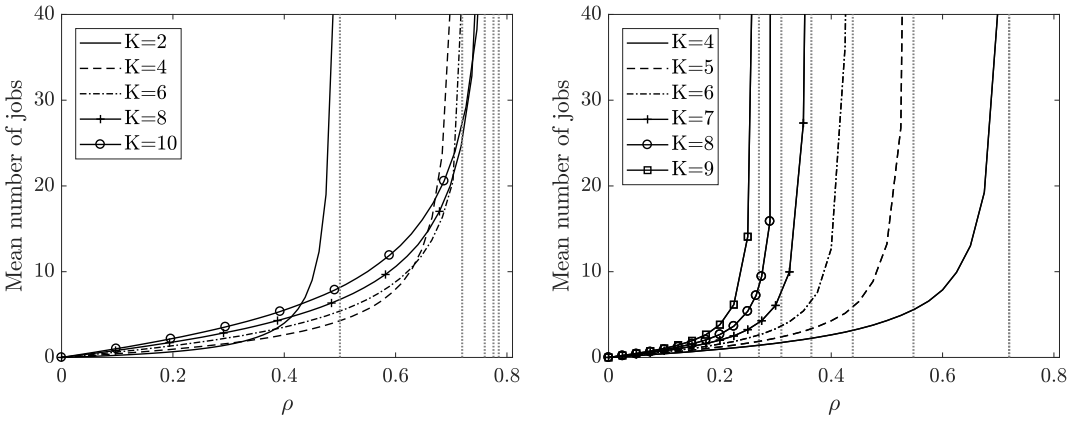
(d) ROS, $K = 8$

Figure 3.6 Mean number of jobs for the redundancy-d system with exponential service times and identical copies vs. the load.

$1/d$, respectively. An interesting observation we can draw from Figure 3.6 (a) and (b) is that for every d, the stability region under FCFS is larger than under PS, as proved in Corollary 3.3.4.

Under the ROS scheduling policy, the stability condition is $\rho < 1$ and does not reduce due to adding redundant copies, Proposition 3.5.3. In order to observe how the mean number of jobs performs with d, in Figure 3.6 (c) and (d) we plot the ratio between the mean number of jobs under d and the mean number of jobs under no redundancy ($d = 1$). If the ratio is below 1, this implies that redundancy (for the particular value of d) improves the performance.

In Figure 3.6 (c) and (d), we observe that for small values of d, redundancy out-



(a) $d = 2$ and $K = 2, \ldots, 9$ (b) $d = K - 2$ and $K = 4, \ldots, 10$

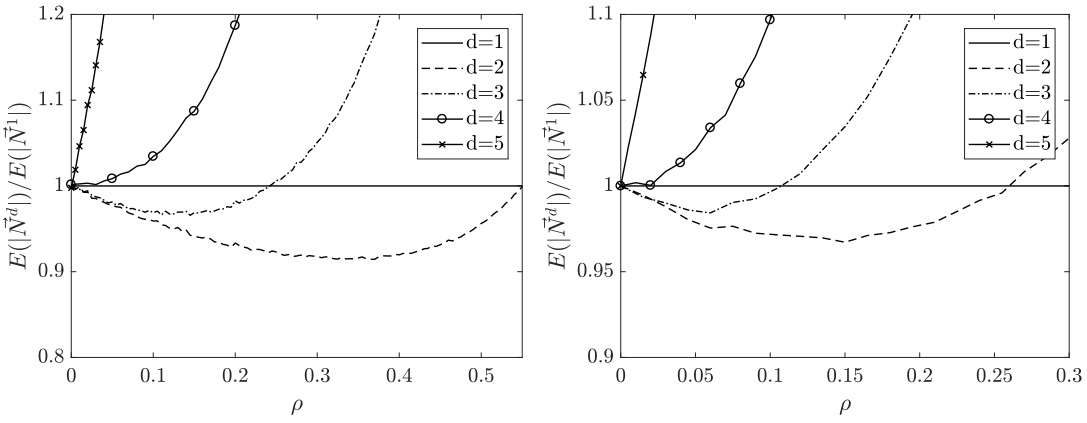
Figure 3.7 Mean number of jobs for the redundancy-d system under FCFS with exponential service times and identical copies vs. the load.

performs the non-redundant system, for any value of the load,. However for d values close K the performance degrades compared to that under redundancy-1. The latter suggests that there is a trade-off between the number of redundant copies at each server and the overload that the latter induces compared to the non-redundant system. Overall, we observe that when the scheduling policy is ROS, redundancy can improve the performance of the system for any value of ρ. This is very different from the FCFS and PS cases, in which the best performance is given by redundancy-1 as the value of ρ increases, see Figure 3.6 (a) and (b).

In Figure 3.7 we focus on FCFS. As before, the vertical lines correspond to the stability region, $\rho < \bar{\ell}/K$. In Figure 3.7 (a), we fix the number of copies to $d = 2$ and plot the performance for several values of the number of servers K. We note that the stability region increases in K (as also proved in Proposition 3.3.5) and that it converges as K grows large to a constant value. In Figure 3.7 (b), we instead set $d = K - 2$, so that the number of copies increases with K. Now, we observe that the stability condition reduces as the number of servers K increases. Hence, the negative impact due to having one more redundant copy, is more important than the benefit of having one more server. This is in agreement with the case $d = K - 1$, for which the the stability region ($\rho < \frac{2}{K}$) also decreases in K.

3.6.2.2 Light-traffic approximation

In this section we consider the steady-state performance for extremely low traffic load, i.e., the so-called light-traffic regime, see Section 2.5 for more details. The light-traffic



(a) FCFS (b) PS

Figure 3.8 Ratio of the mean delay with d identical copies and the mean delay with no redundant copies ($d = 1$), as a function of ρ. For the redundancy-d system ($K = 5$) with exponential service times and identical copies.

approximation corresponds to the first-order asymptotic expansion of the system as $\lambda \to 0$. More precisely, as $\lambda \to 0$ we seek to write $E(|\vec{N}^P(\infty)|) = \bar{N}^{LT,P}(\lambda) + o(\lambda^2)$, for a given scheduling policy P. We defer the details of the light-traffic analysis to Appendix 3.8.E, and we give here the main result of the approach in which we characterize $\bar{N}^{LT,FCFS}(\lambda)$, $\bar{N}^{LT,ROS}(\lambda)$ and $\bar{N}^{LT,PS}(\lambda)$.

Lemma 3.6.1. *The leading term of the light-traffic approximation for FCFS, ROS and PS with identical copies is given by* $\bar{N}^{LT,FCFS}(\lambda) = \bar{N}^{LT,ROS}(\lambda) = \frac{\lambda}{\mu} + \frac{3\lambda^2}{2\mu^2}\frac{1}{\binom{K}{d}}$, *and* $\bar{N}^{LT,PS}(\lambda) = \frac{\lambda}{\mu} + \frac{\lambda^2}{\mu^2}\frac{1}{\binom{K}{d}}$, *respectively.*

We note that for all three policies, the light-traffic term is minimized in $d^* := \lfloor K/2 \rfloor$. To explain this, we note that at very low loads, an arriving job will find at most one other job present. In particular this implies that this new arrival will wait for service if and only if it is of the same type as the job already present in the system. The probability of being of the same type is equal to $1/\binom{K}{d}$, which is minimized by setting d equal to d^*.

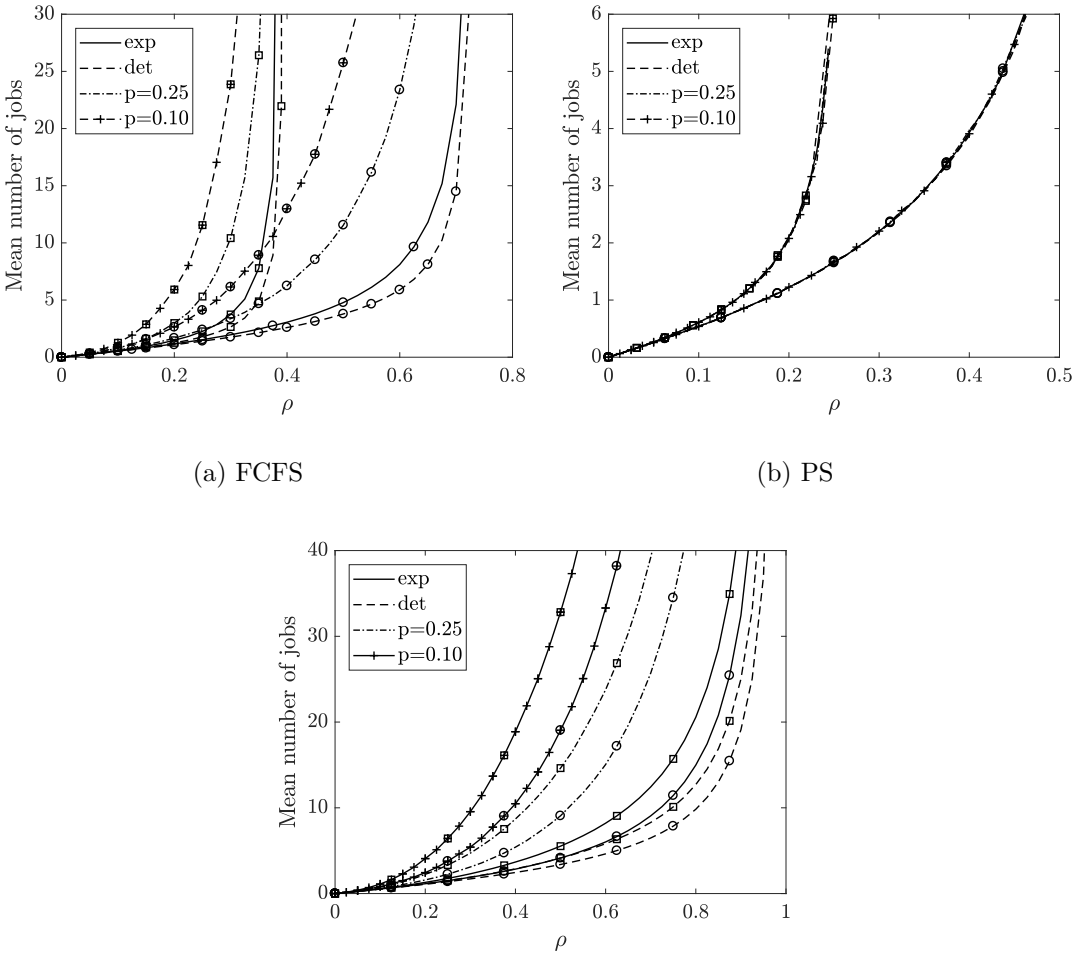
For ROS, we saw in Figure 3.6 that $d = 2$ minimized the mean number of jobs for any value of ρ. In Figure 3.8 (a) and (b) we consider FCFS and PS, respectively, for low load. We plot the ratio of the mean total number of jobs for the system with d identical copies with that of a system with no redundant copies ($d = 1$). If the ratio is below 1, this implies that redundancy (for the particular value of d) improves the performance. As predicted in Lemma 3.6.1, redundancy reduces the mean delay for ρ small enough, and the best performance is obtained in $d = 2$. For sufficiently large load, the minimum

delay is obtained with $d = 1$, see also Figure 3.6.

3.6.2.3 General service time distributions

In Figure 3.9, we compare the mean number of jobs for exponential, deterministic, and degenerate hyperexponential service time distributions. We consider $K = 5$ servers and $d = 2$ and $d = 4$ identical copies.

We observe in Figure 3.9 that for FCFS, PS and ROS, the performance degrades as d increases. This is in contrast to the i.i.d. copies case, where we observed the opposite effect, see Figure 3.5. This is due to the fact that with identical copies, capacity is



(a) FCFS

(b) PS

(c) ROS

Figure 3.9 Mean number of jobs for the redundancy-d system ($K = 5$) with $d = 2(\circ)$ and $d = 4(\square)$, and exponential, deterministic and degenerate hyperexponential ($p = 0.25$ and $p = 0.1$) service times (identical copies) vs. the load.

wasted on serving the exact same copy, while with i.i.d. copies, the system benefits from the difference in the requirement per copy.

For FCFS and ROS, we observe that, unlike in the i.i.d. copies case, the performance of the system degrades as the variability of the service time increases. In particular, for a given d, the best performance is obtained with deterministic service times. Moreover, for the degenerate hyperexponential service time distribution, the performance deteriorates as p decreases. For FCFS, these observations are in agreement with the results obtained by [58] for the mean field analysis.

From the numerics, it seems that for deterministic or degenerate hyperexponential service times, ROS is more stable than when FCFS and PS are implemented.

Another observation is that for PS the mean number of jobs in the systems seems to almost coincide for the different service time distribution, that is, the performance seems to be nearly insensitive to the service time distribution (beyond its mean value). When degenerate hyperexponential service times are considered, it is trivial that the performance coincides with that of exponential service requirements. This can be explained as follows: With identical copies, only a fraction p of the arrivals have a non-zero service requirement, and this is exponentially distributed with mean $1/(p\mu)$. Thus, the system with degenerate hyperexponential service requirements with parameter p is equivalent to the system with arrival rate λ and exponentially distributed service requirements with mean $1/\mu$ where time is parametrized with parameter p. Similar results are obtained in [55], where the authors show that for JSQ scheduling with PS, the mean number of jobs is nearly insensitive to the service time distribution.

3.7 Concluding remarks

In the present chapter, we have investigated the stability region and performance of the redundancy-d systems where servers are equally loaded, that is, the system where servers have homogeneous capacities, and each arriving job sends d copies into uniformly chosen d servers at random. We have shown that when jobs have i.i.d. copies, the stability condition of the system is not reduced due to adding redundant copies for both PS and ROS scheduling policies.

We observe that under the identical copies assumption, the stability condition strongly depends on the implemented scheduling policy. When ROS is implemented, the stability condition is independent of the redundancy degree d and does not reduce due to adding redundant copies. However, when FCFS and PS are implemented, the stability condition is dramatically reduced for the redundancy-d homogeneous system. In Chapter 4 we investigate the stability condition for redundancy models with general redundancy topologies and identical copies.

3.8 Appendix

A: Proofs of Section 3.2

Proof of Lemma 3.2.1: From the law of large numbers, we obtain that almost surely,

$$\lim_{r\to\infty}\frac{1}{r}\tilde{A}_c(rt)=\frac{\lambda}{\binom{K}{d}}t \text{ and } \lim_{r\to\infty}\frac{1}{r}\tilde{S}_c(s)ds=\mu t. \tag{3.12}$$

The cumulative amount of capacity spent on serving type-c jobs in server s, $T_{s,c}^{IID,r}(t)$ increases at rate $N_c^{IID}(t)/M_s^{IID}(t)\leq 1$. Hence, $\frac{1}{r}T_{s,c}^{IID,r}(rt)-\frac{1}{r}T_{s,c}^{IID,r}(ru)\leq(t-u)$ for every $t\geq u$, i.e., $T_{s,c}^{IID,r}(rt)/r$ is Lipschitz continuous. Therefore, by the Arzela-Ascoli theorem we obtain that for almost all sample path ω and any sequence r_k, there exists a subsequence r_{k_j} such that $\lim_{j\to\infty}\frac{T_c^{IID,r_{k_j}}(r_{k_j}t)}{r_{k_j}}=\tau_c^{IID}(t)$, u.o.c.. Together with Eq. (3.2) and Eq. (3.12), we obtain Eq. (3.3). □

Proof of Lemma 3.2.2: For ease of notation, we removed the superscript IID throughout the proof. Let $f(\vec{n})=(f_c(\vec{n}),c\in\mathcal{C})$, with $f_c(\vec{n}):\mathbb{R}_+^{|\mathcal{C}|}\to\mathbb{R}^{|\mathcal{C}|}$, denote the drift vector field of $\vec{N}(t)$ when starting in state $\vec{N}(0)=\vec{n}$, i.e., $f(\vec{n})=\frac{d}{dt}\mathbb{E}^{\vec{n}}[\vec{N}(t)]\Big|_{t=0}$. We can deduce from the results of [47, Proposition 5] that the fluid limit $\vec{n}(t)$ satisfies

$$\frac{\mathrm{d}\vec{n}(t)}{\mathrm{d}t}\in F(\vec{n}(t)), \tag{3.13}$$

where

$$F(\vec{n}):=conv\left(acc_{r\to\infty}\ f(r\vec{n}^r)\ \ with\ \ \lim_{r\to\infty}\vec{n}^r=\vec{n}\right). \tag{3.14}$$

Here, $acc_{r\to\infty}\ x^r$ denotes the set of accumulation points of the sequence x^r when r goes to infinity and $conv(A)$ is the convex hull of set A. An illustration of how F is constructed is available in Figure 1 in [47, Section 2].

Using Eq. (3.13) and Eq. (3.14), we can partly characterize the fluid process $m_s(t)=\sum_{c\in\mathcal{C}(s)}n_c(t)$. Denote by $\tilde{f}_s(\vec{n})=\sum_{c\in\mathcal{C}(s)}f_c(\vec{n})$ the one-step drift of $M_s(t)$. The arrival rate of copies to server s equals $\lambda\binom{K-1}{d-1}/\binom{K}{d}=\lambda d/K$. Recall from Eq. (3.1) that the total departure rate of copies from server s equals $\mu\left(\sum_{c\in\mathcal{C}(s)}\sum_{l\in c}\frac{n_c}{m_l}\right)$. Hence, in state $\vec{n}$, the drift of server s is equal to

$$\tilde{f}_s(\vec{n})=\lambda\frac{d}{K}-\mu\left(\sum_{c\in\mathcal{C}(s)}\sum_{l\in c}\frac{n_c}{m_l}\right). \tag{3.15}$$

Let $G_2(\vec{n}):=\{s\in S:m_s\geq m_l,\forall l\}$. Note that if $s\in G_2(\vec{n})$, then $\left(\sum_{c\in\mathcal{C}(s)}\sum_{l\in c}\frac{n_c}{m_l}\right)\geq\left(\sum_{c\in\mathcal{C}(s)}\sum_{l\in c}\frac{n_c}{m_s}\right)=\sum_{c\in\mathcal{C}(s)}d\frac{n_c}{m_s}=d$.

Now, let $\lim_{r\to\infty} \vec{n}^r = \vec{n}$, and $\vec{n} \neq \vec{0}$. Then, for $s \in G_2(\vec{n})$,

$$\lim_{r\to\infty} \tilde{f}_s(r\vec{n}^r) = \lambda d/K - \mu \left(\sum_{c\in\mathcal{C}(s)} \sum_{l\in c} \frac{n_c}{m_l} \right) \leq \lambda d/K - \mu d.$$

Together with Eq. (3.13), Eq. (3.14) and $\sum_{c\in\mathcal{C}(s)} \frac{\mathrm{d}n_c(t)}{\mathrm{d}t} = \frac{\mathrm{d}m_s(t)}{\mathrm{d}t}$, this concludes the proof. □

Proof of Proposition 3.2.3: Define $m^{IID}_{max}(t) := \max_{s\in S}\{m^{IID}_s(t)\}$ and fix $T = m^{IID}_{max}(0)/d(\mu - \frac{\lambda}{K})$. From Lemma 3.2.2, we know that at time T, $m_{max}(T) = 0$. Since for any $s \in S$, $m^{IID}_s(t) \leq m^{IID}_{max}(t)$, we conclude that at time T the fluid system is empty. From Theorem 2.4.10 we conclude that the process is ergodic. □

B: Comments and proofs of Section 3.3

Balance equations of the saturated system

For $\vec{e}_1, \vec{e}_2 \in \overline{E}$, we denote by $q(\vec{e}_1, \vec{e}_2)$ the transition probability from state $\vec{e}_1$ to state $\vec{e}_2$. Recall that in state $\vec{e} = (O_{\ell^*}, \ldots, O_2, L_1, O_1) \in \bar{E}$, exactly $\ell^*(\vec{e}) := \ell^*$ jobs are being served, each of them with departure rate μ. Hence, the balance equations of the saturated system are given by

$$\mu\ell^*(\vec{e}_1)\pi(\vec{e}_1) = \sum_{\vec{e}_2\in\bar{E}} q(\vec{e}_2, \vec{e}_1)\pi(\vec{e}_2),$$

with $\pi(\vec{e})$ the steady-state distribution.

We will write down the balance equations in the case $d = K - 2$. In that case, at any moment in time there is one job that is served in $d = K - 2$ servers. In the remaining two servers, either one job, or two jobs are served. Hence, the states of the saturated system are of the form $\vec{e} = (O_3, L_2, O_2, L_1, O_1)$ and $\vec{e} = (O_2, L_1, O_1)$. We denote by $\mathcal{C}(O_1) := \{c \in \mathcal{C} : c \cup O_1 = \{1, \ldots, K\}\}$ the subset of types that together with type O_1 make all servers busy. Hence, if the system is in state $\vec{e} = (c, L_1, O_1)$, $c \in \mathcal{C}(O_1)$, the total departure rate is 2μ. We denote by $\bar{\mathcal{C}}(O_1) := \mathcal{C} - O_1 \cup \mathcal{C}(O_1)$ the subset of types that together with O_1 do not use all servers. For $O_1, O_2 \in \mathcal{C}$, we denote by $\mathcal{C}(O_1, O_2) := \{c \in \mathcal{C} : c \cup O_1 \cup O_2 = \{1, \ldots, K\}\}$ the subset of types that together with O_1 and O_2 make all servers busy.

The balance equations are given by:

$$\begin{aligned}2\mu\pi(O_2,L_1,O_1) \;=\; & \mu\pi(O_2,L_1+1,O_1)+\mu\textstyle\sum_{c\in\mathcal{C}(O_1)}\sum_{j=0}^{L_1}(\frac{1}{|\mathcal{C}|})^{j+1}\pi(c,L_1-j,O_1)\\ &+\mu\textstyle\sum_{c\in\mathcal{C}(O_1)}(\frac{1}{|\mathcal{C}|})^{L_1+1}\pi(O_1,0,c)+\mu\sum_{c\in\bar{\mathcal{C}}(O_1)}\left(\frac{1}{\binom{K-1}{d}}\right)^{L_1}\pi(O_2,L_1,O_1,0,c)\\ &+\mu\textstyle\sum_{c\in\bar{\mathcal{C}}(O_1)}\sum_{j=0}^{L_1}\left(\frac{1}{\binom{K-1}{d}}\right)^{j}\pi(O_2,j,c,L_1-j,O_1),\end{aligned}$$

with $L_1\geq 0$, $O_1\in\mathcal{C}$, and $O_2\in\mathcal{C}(O_1)$. The term $(1/\binom{K-1}{d})^j$ in the fourth and fifth term on the right represents the probability that all j waiting jobs are of type O_1 (types O_1 and c occupy $K-1$ servers, hence $\binom{K-1}{d}$ is the number of possible types that can compose L_1). For a 3μ departure rate configuration state we have

$$\begin{aligned}3\mu\pi(O_3,L_2,O_2,L_1,O_1) &= \mu\pi(O_3,L_2,O_2,L_1+1,O_1)\\ &+\mu\textstyle\sum_{c\in\bar{\mathcal{C}}(O_1)\cap\mathcal{C}(O_1,O_3)}\sum_{j=0}^{L_1}\left(\frac{1}{\binom{K-1}{d}}\right)^{j+1}\pi(O_3,L_2+j+1,c,L_1-j,O_1)\\ &+\mu\textstyle\sum_{c\in\mathcal{C}(O_1,O_2)}\sum_{j=0}^{L_2}\left(\frac{\binom{K-1}{d}}{|\mathcal{C}|}\right)^{j}\frac{1}{|\mathcal{C}|}(c,L_2-j,O_2,L_1,O_1)\\ &+\mu\textstyle\sum_{c\in\bar{\mathcal{C}}(O_1)\cap\mathcal{C}(O_1,O_3)}\left(\frac{1}{\binom{K-1}{d}}\right)^{L_1+1}\pi(O_3,L_1+L_2+1,O_1,0,c)\\ &+\mu\textstyle\sum_{c\in\bar{\mathcal{C}}(O_1)\cap\mathcal{C}(O_1,O_2)}\left(\frac{\binom{K-1}{d}}{|\mathcal{C}|}\right)^{L_2}\frac{1}{|\mathcal{C}|}\left(\sum_{j=0}^{L_1}\left(\frac{1}{\binom{K-1}{d}}\right)^{j}\pi(O_2,j,c,L_1-j,O_1)\right.\\ &\left.+\left(\frac{1}{\binom{K-1}{d}}\right)^{L_1}\pi(O_2,L_1,O_1,0,c)\right)\\ &+\mu\textstyle\sum_{c\in\mathcal{C}(O_1)}\left(\frac{\binom{K-1}{d}}{|C|}\right)^{L_2}\left(\frac{1}{|\mathcal{C}|}\right)^{2}\left(\sum_{j=0}^{L_1}\left((\frac{1}{|\mathcal{C}|})^{j}\pi(c,L_1-j,O_1)\right)+(\frac{1}{|\mathcal{C}|})^{L_1}\pi(O_1,0,c)\right),\end{aligned}$$

with $O_1\in\mathcal{C}$, $O_2\in\bar{\mathcal{C}}(O_1)$, $O_3\in\mathcal{C}(O_1,O_2)$ and $L_1,L_2\geq 0$. Note that on the right-hand-side, the term $\frac{\binom{K-1}{d}}{|C|}$ is the probability that an arriving job is of type $c\in O_1\cup O_2$ (the number of types c with $c\in O_1\cup O_2$ is equal to $\binom{K-1}{d}$).

Some properties of $\bar{\ell}$: In the proof, we will make use of the following properties for the saturated system (as defined in Definition 3.3.2). Recall that the average total departure rate of the saturated system is given by $\bar{\ell}\mu$, where $\bar{\ell}$ is defined in Eq. (3.4) as the average number of jobs in service. For the saturated system, recall that $\ell^*(\vec{e})$ denotes the number of jobs that is served in state $\vec{e}$. Hence

$$\lim_{t\to\infty}\frac{1}{t}\int_0^t \ell^*(\vec{e}(u))\mathrm{d}u=\bar{\ell},\quad\text{almost surely.}\tag{3.16}$$

For the saturated system, let $\ell_c^*(\vec{e})$ equal 1 if a type-c job is served in state $\vec{e}$ and 0 otherwise. Note that $\sum_{c\in\mathcal{C}}\ell_c^*(\vec{e})=\ell^*(\vec{e})$. Hence, using the system symmetry, (or more precisely, the exchangeability of the server contents), and together with Eq. (3.16), we

obtain that

$$\lim_{t\to\infty}\frac{1}{t}\int_0^t \ell_c^*(\vec{e}(u))\mathrm{d}u = \frac{\bar{\ell}}{\binom{K}{d}}, \text{ almost surely.} \tag{3.17}$$

Proof of Proposition 3.3.5: We are given a saturated system with K servers, with a central queue where jobs wait in order of arrival. The system starts serving at time 0. Let $c^K(i)$ denote the type of the i-th job at time 0 in this central queue. Let $\alpha_{is}^K(t)$ denote the attained service of this job at time t in server $s \in c^K(i)$. Once the job i departs, the attained service $\alpha_{is}^K(t)$ is set equal to β_i, the service requirement of job i. Let $D_c^K(t)$ denote the number of departed type-c jobs in the interval $(0,t]$ and $D_s^K(t) := \sum_{c\in\mathcal{C}^K(s)} D_c^K(t)$ the number of departed jobs from server s, with $\mathcal{C}^K$ the set of types with K servers. We will prove that

$$D_s^K(t) \geq_{st} D_s^{K-1}(t), \text{ with } s \text{ an arbitrary server in each of the systems.} \tag{3.18}$$

Before proving this, we first show how Eq. (3.18) implies that $\bar{\ell}/K$ is increasing in K, as stated in Proposition 3.3.5. From Eq. (3.18) we have $\lim_{t\to\infty}\frac{1}{t}D_s^K(t) \geq \lim_{t\to\infty}\frac{1}{t}D_s^{K-1}(t)$, that is, the long-run departure rate from server s is increasing in the number of servers. Note that $\mu\sum_{c\in\mathcal{C}(s)}\ell_c^*(\vec{e}(t))$ is the instantaneous departure rate from server s, where $\ell_c^*(\vec{e}(t))$ equals 1 if a type-c job is served, and equals 0 otherwise. From Eq. (3.17), we have that the long-run departure rate from server s can equivalently be written as $\lim_{t\to\infty}\frac{1}{t}\mu\sum_{c\in\mathcal{C}(s)}\int_0^t \ell_c^*(\vec{e}(t))\mathrm{d}u = \frac{\bar{\ell}\mu}{\binom{K}{d}}\binom{K-1}{d-1} = \frac{d\bar{\ell}\mu}{K}$. Since the long-run departure rate is increasing in the number of servers, this implies that $\frac{\bar{\ell}}{K}$ is increasing in K and proves the statement of Proposition 3.3.5.

We are left with proving Eq. (3.18). In order to do so, we will couple the system with K servers to a system with $K-1$ servers as follows. We consider the central queue associated to the saturated system with K servers, which corresponds to an infinite backlog of jobs (at time 0) ordered according to arrival (from $-\infty$). In the $K-1$ server model, server K is removed. We couple the $K-1$ server model to the K server model, by creating the central queue for the $K-1$ system as follows. For each i-th job in the central queue that has a copy in server K, i.e., $K \in c^K(i)$, we choose uniformly at random another server among the remaining $K\backslash c^K(i)$ servers, denoted by $s^{K-1}(i)$. Hence, for any job with $K \in c^K(i)$, we set its type in the $K-1$ system as $c^{K-1}(i) = (c^K(i)\backslash K)\cup s^{K-1}(i)$. For all jobs with $K \notin c^K(i)$, we set $c^{K-1}(i) = c^K(i)$. Below we show that for all $t \geq 0$,

$$\alpha_{is}^K(t) \geq \alpha_{is}^{K-1}(t), \forall i = 1,\dots \text{ and } \forall s \in c^K(i)\backslash K. \tag{3.19}$$

and

$$\alpha_{iK}^{K}(t) \geq \alpha_{is^{K-1}(i)}^{K-1}(t). \tag{3.20}$$

From Eq. (3.19) and Eq. (3.20) we obtain that Eq. (3.18) holds: If a job i departs from a server s in the $K-1$ system, then (i) either also $s \in c^K(i)$, in which case this job has departed at a time $u \leq t$ in the K system (from Eq. (3.19)), (ii) or $s \neq c_i^K$, which implies that the type of the job is different in the K system and $K-1$ system, hence $s = s^{K-1}(i)$. Then, from Eq. (3.20) it follows that this job has departed at a time $u \leq t$ in the K system. To conclude, in both cases, job i has already departed in the K system before it departs in the $K-1$ system, hence, Eq. (3.18) holds.

The result in Eq. (3.19) and Eq. (3.20) will be proved by induction. It holds at time 0. Now assume that for all $u \leq t$ it holds that $\alpha_{is}^{K}(u) \geq \alpha_{is}^{K-1}(u), \forall i = 1, \ldots$ and $\forall s \in c^K(i)\backslash K$. and $\alpha_{iK}^{K}(u) \geq \alpha_{is^{K-1}(i)}^{K-1}(u)$. We prove that this remains true at time t^+.

In order for the inequality Eq. (3.19) to no longer be valid at time t^+, it needs to hold that either Eq. (3.19) or Eq. (3.20) hold with strict equality. We first assume the first case, that is, $\alpha_{is}^{K}(t) = \alpha_{is}^{K-1}(t)$, for some i and $s \in c^K(i)\backslash K$. If $\alpha_{is}^{K}(t) = \alpha_{is}^{K-1}(t) = 0$ and in the $K-1$ system it holds that $\alpha_{is}^{K-1}(t^+) > 0$, then this implies that one of the following occurs:

(1) in the K system, the server s is serving the i_1-th job, with $i_1 < i$, while in the $K-1$ system, server s starts serving job i at time t^+. However, since $\alpha_{i_1\tilde{s}}^{K}(t) \geq \alpha_{i_1\tilde{s}}^{K-1}(t)$, for all $\tilde{s} \in c^K(i)\backslash K$, this implies that job i_1 should not have a copy in server s in the $K-1$ system, since otherwise, job i_1 was also still in service in the $K-1$ system. However, due to the construction of the coupling and since $s \neq K$, such a job i_1 does not exist.

(2) in the $K-1$ system, this job i finishes its service in server $\tilde{s}$, that is, $\alpha_{i\tilde{s}}^{K-1}(t^+) = \beta_i$ and hence $\alpha_{is}^{K-1}(t^+) = \beta_i$. But since Eq. (3.19) and Eq. (3.20) hold at time t, this job is then also finished in the K system, and hence also $\alpha_{is}^{K}(t^+) = \beta_i$.

Now assume $\alpha_{is}^{K}(t) = \alpha_{is}^{K-1}(t) > 0$ for some i and $s \in c^K(i)\backslash K$. Then, both jobs are in service in server s, hence the inequality remains valid, unless the job departs in the $K-1$ system (and hence $\alpha_{is}^{K-1}(t^+) = \beta_i$), but not in the K system. This can however not happen, since $\alpha_{j\tilde{s}}^{K}(t) \geq \alpha_{j\tilde{s}}^{K-1}(t), \forall j\tilde{s} \in c^K(j)\backslash K$ and $\alpha_{jK}^{K}(t) \geq \alpha_{js^{K-1}(j)}^{K-1}(t)$. Hence, the inequality remains valid at time t^+.

To prove that $\alpha_{is}^{K}(t) = \alpha_{is}^{K-1}(t)$, implies $\alpha_{is}^{K}(t^+) = \alpha_{is}^{K-1}(t^+)$ follows exactly the same steps and is therefore left out. □

Proof of Lemma 3.3.6: Both systems are coupled as follows: At time $t = 0$, $N_c^{FCFS}(0) = 0$ and $N_c^{(T)}(0) = \tilde{A}_c(T)$, where $\tilde{A}_c(t)$ is the arrival process of type-c jobs. During the time interval $[0, T]$, we couple the original system and its modified

version by using the same arrivals and service times in the FCFS systems, as those that arrived in the $\vec{N}^{(T)}$-system at time 0.

The result will be proved by induction. It holds at time 0. Now assume that for all $u \leq t$ it holds that $N_c^{(T)}(u) \leq N_c^{FCFS}(u) + (\tilde{A}_c(T) - \tilde{A}_c(u))^+$ and $a_{cis}^{FCFS}(u) \leq a_{cis}^{\vec{N}^{(T)}}(u)$, for all $i = 1, \ldots, N_c^{FCFS}(t)$, $c \in \mathcal{C}$, $s \in S$. We prove that this remains true at time t^+.

For that, assume there is a c such that $N_c^{(T)}(t) = N_c^{FCFS}(t) + (\tilde{A}_c(T) - \tilde{A}_c(t))^+$. If $t < T$, only in the FCFS system we can have an arrival, in which case $N_c^{FCFS}(t^+) = N_c^{FCFS}(t) + 1$ and $(\tilde{A}_c(T) - \tilde{A}_c(t^+))^+ = (\tilde{A}_c(T) - \tilde{A}_c(t))^+ - 1$. Hence, the inequality remains valid. If $t \geq T$, then an arrival in the FCFS system is coupled to an arrival in the $\vec{N}^{(T)}$-system, hence $N_c^{(T)}(t^+) = N_c^{FCFS}(t^+)$ (and note that $(\tilde{A}_c(T) - \tilde{A}_c(t))^+ = 0$). Now, assume the i-th type-c job departs in the FCFS system (which can cause a violation of the inequality). Since $a_{ci\tilde{s}}^{FCFS}(t) \leq a_{ci\tilde{s}}^{\vec{N}^{(T)}}(t)$, for all $\tilde{s}$, it holds that the same job departs in the $\vec{N}^{(T)}$-system. Hence, in all cases, the inequality $N_c^{(T)}(t^+) \leq N_c^{FCFS}(t^+) + (\tilde{A}_c(T) - \tilde{A}_c(t^+))^+$ remains valid at time t^+.

Now assume there exists a c, i, s such that $a_{cis}^{FCFS}(t) = a_{cis}^{\vec{N}^{(T)}}(t)$. First assume $a_{cis}^{FCFS}(t) = a_{cis}^{\vec{N}^{(T)}}(t) > 0$. Because of FCFS, in both systems this copy has entered service in server s at the same instant of time. Hence, it cannot happen that $a_{cis}^{FCFS}(t^+) > a_{cis}^{\vec{N}^{(T)}}(t^+)$. If instead $a_{cis}^{FCFS}(t) = a_{cis}^{\vec{N}^{(T)}}(t) = 0$, the i-th type-c copy in server s is waiting in the queue in both systems. We need to prove that if this copy would enter service in server s at time t^+ in the FCFS system, it also enters service in the $\vec{N}^{(T)}$-system in server s. From the FCFS policy and $a_{\tilde{c}js}^{FCFS}(t) \leq a_{\tilde{c}js}^{\vec{N}^{(T)}}(t)$, for all $\tilde{c}, j \leq i$, this follows directly. □

Proof of Proposition 3.3.7: We will prove that if the $\vec{N}^{(T)}$-system is stable and T is sufficiently large, then $\lambda \leq \bar{\ell}\mu$. From Lemma 3.3.6, it follows that stability of the FCFS system, implies stability of the $\vec{N}^{(T)}$-system, and hence $\lambda \leq \bar{\ell}\mu$, which would conclude the proof.

We will now prove that if the $\vec{N}^{(T)}$-system is stable, then $\lambda \leq \bar{\ell}\mu$. We define the random variable $\tau(T)$ as the first moment in time a servers gets empty in the $\vec{N}^{(T)}$-system, i.e., $\tau(T) := \min\{u : M_s^{(T)}(u) = 0, \text{for some server } s\}$. Up till time $\tau(T)$, the $\vec{N}^{(T)}$-system is stochastically equivalent to the saturated system. Hence, using the Markovian description of the process $M_s^{(T)}$ and Dynkin's formula, we have that there exists a martingale $(Z_s(t))_{t\geq 0}$ such that

$$M_s^{(T)}(\tau(T)) = M_s^{(T)}(0) + \frac{d\lambda(T + (\tau(T) - T)^+)}{K} - \int_0^{\tau(T)} \mu \sum_{c \in C(s)} \ell_c(\vec{e}(u))\mathrm{d}u + Z_s(\tau(T)). \tag{3.21}$$

Since the increasing process associated to Z_s is bounded in mean by Ct, with $C > 0$, it follows that $\sup_t E(\frac{Z_s(t)}{t})^2 \leq \frac{Ct}{t^2}$, which in turn implies that $\frac{Z_s(\tau(T))}{\tau(T)} \to 0$ in L^2 and

hence the convergence holds almost surely.

Since by the law of large numbers for the Poisson process, $\liminf_{T\to\infty} \frac{\tau(T)}{T} \geq c > 0$, it follows that $\liminf_{T\to\infty} M_s^{(T)}(0)/\tau(T) = 0$ and $\liminf_{T\to\infty} \int_0^{\tau(T)} \mu \sum_{c\in C(s)} \ell_c(\vec{e}(u))\mathrm{d}u = \bar{\ell}\mu$ almost surely. Now, together with Eq. (3.21) and Eq. (3.17), it follows that

$$\lim_{T\to\infty} \frac{M_s^{(T)}(\tau(T))}{\tau(T)} = \lim_{T\to\infty} \frac{d\lambda}{K} \frac{T + (\tau(T) - T)^+}{\tau(T)} - \frac{d}{K}\bar{\ell}\mu, \quad \text{almost surely.} \tag{3.22}$$

By assumption, the $\vec{N}^{(T)}$-system is stable, hence, $\mathbb{E}(\tau(T)) < \infty$. We can take a sample path realization ω such that $\tau(T) < \infty$ and Eq. (3.22) holds. At least one of the following two cases holds:

- there exists a subsequence T_i of T, such that $\tau(T_i) \geq T_i$ and for each T_i, the server that empties first is the same server $\tilde{s}$. For any $\epsilon > 0$, we have for i large enough that
$$M_{\tilde{s}}^{(T_i)}(\tau(T_i)) \geq M_{\tilde{s}}^{(T)}(0) - \frac{d}{K}(\bar{\ell}\mu - \lambda)\tau(T_i) - o(\epsilon).$$
- there exists a subsequence T_i of T, such that $\tau(T_i) < T_i$ and for each T_i, the server that empties first is server $\tilde{s}$. For any $\epsilon > 0$, we have for i large enough that
$$M_{\tilde{s}}^{(T_i)}(\tau(T_i)) \geq M_{\tilde{s}}^{(T)}(0) - \frac{d}{K}(\bar{\ell}\mu - \lambda)\tau(T_i) - o(\epsilon),$$
since $\tau(T_i) < T_i$.

For both cases we have that since $M_{\tilde{s}}^{(T_i)}(\tau(T_i)) = 0$, this implies $\lambda < \bar{\ell}\mu$ (if $M_{\tilde{s}}^{(T)}(0) > 0$) and it implies $\lambda = \bar{\ell}\mu$ (if $M_{\tilde{s}}^{(T)}(0) = 0$). □

Proof of Lemma 3.3.8: We couple both systems as follows: at time zero, both systems start in the same initial state, that is, $N_c^{(0)}(0) = N_c^{FCFS}(0)$ and $a_{cis}^{\vec{N}^{(0)}}(0) = a_{cis}^{FCFS}(0)$, for all c, i, s. Arrivals and their service requirements are coupled.

The result will be proved by induction. It holds at time 0. Now assume that for all $u \leq t$ it holds that $N_c^{(0)}(u) \geq N_c^{FCFS}(u)$ and $a_{cis}^{\vec{N}^{(0)}}(u) \leq a_{cis}^{FCFS}(u)$ for all $i = 1, \ldots, N_c^{FCFS}(t)$, $c \in \mathcal{C}$, $s \in S$. Below, we prove that this remains true at time t^+.

Assume there is a c such that $N_c^{(0)}(t) = N_c^{FCFS}(t)$. The inequality can be violated at time t^+ if there is a type-c departure in $\vec{N}^{(0)}$, but not in FCFS. However, note that if the head-of-the-line type-c job in the $\vec{N}^{(0)}$-system departs, since $a_{c1\tilde{s}}^{\vec{N}^{(0)}}(t) \leq a_{c1\tilde{s}}^{FCFS}(t)$, this job would also depart from the FCFS system. Hence, $N_c^{(0)}(t^+) = N_c^{FCFS}(t^+)$ at time t^+.

Now assume there exists a c, i, s such that $a_{cis}^{\vec{N}^{(0)}}(t) = a_{cis}^{FCFS}(t)$. First assume $a_{cis}^{\vec{N}^{(0)}}(t) = a_{cis}^{FCFS}(t) > 0$. Because of FCFS, in both systems this copy has entered service in server s at the same instant of time. Hence, it cannot happen that $a_{cis}^{\vec{N}^{(0)}}(t^+) >$

$a_{cis}^{FCFS}(t^+)$. If instead $a_{cis}^{\vec{N}^{(0)}}(t) = a_{cis}^{FCFS}(t) = 0$, the i-th type-c copy in server s is waiting in the queue in both systems. We need to prove that if this copy would enter service in server s at time t^+ in the $\vec{N}^{(0)}$-system, it also enters service in the FCFS-system in server s. From the FCFS policy and $a_{\tilde{c}js}^{\vec{N}^{(0)}}(t) \leq a_{\tilde{c}js}^{FCFS}(t)$, for all $\tilde{c}, j \leq i$, this follows directly. □

Proof of Proposition 3.3.9: From Lemma 3.3.8 we have that $\vec{N}^{(0)}$-system is an upper bound for the original FCFS system. Hence, it will be enough to prove stability of the process $\vec{N}^{(0)}(t)$.

In order to prove stability of $\vec{N}^{(0)}(t)$, we study the fluid-scaled system. That is, for each r, we study $\vec{N}^{(0),r}(t)$, with $\vec{N}^{(0),r}(0) = r\vec{n}(0)$. Define $T_0 = \frac{|\vec{n}(0)|}{\mu}$. By definition of $\vec{N}^{(0),r}(t)$, in the interval $[0, rT_0]$, only those jobs present at time 0 are served (according to FCFS). From time $rT_0 = \frac{|\vec{N}^{(0),r}(0)|}{\mu}$ onwards, all jobs can be served.

We write $\vec{N}^{(0),r}(t) = \vec{N}_A^{(0),r}(t) + \vec{N}_B^{(0),r}(t)$, where $\vec{N}_A^{(0),r}(t)$ denotes the number of old jobs, that is, the number of jobs present at time t among those that were already present at time $t = 0$. We let $\vec{N}_B^{(0),r}(t) = \vec{N}^{(0),r}(t) - \vec{N}_A^{(0),r}(t)$ denote the number of new jobs present at time t. Similarly, we let $M_{s,B}^{(0)}(t)$ denote the number of new jobs that have a copy in server s.

We define

$$T_1 := T_0 \frac{\lambda}{\bar{\ell}\mu - \lambda} = |\vec{n}(0)| \frac{\lambda}{\mu(\bar{\ell}\mu - \lambda)}.$$

We show that for any fluid limit $\vec{n}(t)$ of $\vec{N}^{(0),r}(rt)$ it holds that it is zero at some time smaller than or equal to $T_0 + T_1$, that is $|\vec{n}(t)| = 0$, for some $t \leq T_1 + T_0$.

In the interval $[0, rT_0]$, the system serves only the jobs present at time 0. Let $\hat{\ell}_A^r(t)$ denote the number of such jobs in service at time t in the $\vec{N}^{(0),r}$-system. Hence, using the Markovian description of the process $|\vec{N}_A^{(0),r}(rt)|$ and Dynkin's formula, we have that there exists a martingale $(Z(t))_{t\geq 0}$ such that

$$\frac{|\vec{N}_A^{(0),r}(rt)|}{r} = |\vec{n}(0)| - \mu \frac{1}{r} \int_0^{rt} \hat{\ell}_A(u) \mathrm{d}u + \frac{Z(rt)}{r}, \tag{3.23}$$

for $t \in [0, T_0]$. Since the increasing process associated to Z is bounded in mean by Ct, with $C > 0$, it follows that $\sup_t E(\frac{Z(rt)}{r})^2 < \infty$, which in turn implies that $\frac{Z(rt)}{r} \to 0$ almost surely. Further note that $\hat{\ell}_A(u) \geq 1$ whenever $\vec{n} \neq \vec{0}$. Together, this gives that

$$\lim_{r\to\infty} \frac{|\vec{N}_A^{(0),r}(rt)|}{r} \leq \max(0, |\vec{n}(0)| - \mu t), \text{for } t \in [0, T_0].$$

Hence, for any fluid limit $\vec{n}_A(t)$, we have $|\vec{n}_A(t)| \leq \max(0, |\vec{n}(0)| - \mu t)$, so that $|\vec{n}_A(T_0)| = 0$.

In the remainder of the proof, we study fluid limits of the process $\vec{N}_B^{(0),r}(rt)/r$. We

define the random variable $T_1^r := \inf\{t > 0 : \text{there is an } s \text{ s.t. } M_{s,B}^{(0),r}(r(T_0+t)) = 0\}$ as the first moment after time T_0 that one of the servers gets empty in the fluid limit. By the law of large numbers, $\liminf_r T_1^r \geq c > 0$, almost surely. Hence, without loss of generality, we can focus on sample paths such that the latter is the case. For a given sample path we let

$$\tilde{T}_1 := \liminf_{r\to\infty} T_1^r.$$

Note that $\tilde{T}_1 \geq c$. We consider henceforth the subsequence r_j of any given sequence r, such that

$$T_1^{r_j} > \tilde{T}_1 - \epsilon, \ \forall r_j.$$

In particular, this implies that all servers are working on new jobs during the interval $[r_jT_0, r_jT_0 + r_j(\tilde{T}_1 - \epsilon)]$, for any r_j. Also note that all jobs $\vec{N}_B^{(0)}(T_1)$ are "freshly" sampled, and hence the system behaves as a saturated system during this time frame.

Using the Markovian description of the process $\vec{M}_B^{(0),r}(rt)$ and Dynkin's formula, we have that there exists a martingale $(Z_s(t))_{t\geq 0}$ such that

$$|M_{s,B}^{(0),r}(rt)| = \lambda\frac{dt}{K} - \mu\int_{rT_0}^{rt}\sum_{c\in\mathcal{C}(s)}\ell_c^*(\vec{e}(u))\mathrm{d}u + Z_s(rt), \tag{3.24}$$

where t is such that $T_0 \leq t \leq T_0 + \tilde{T}_1 - \epsilon$. We recall that $\ell_c^*(\vec{e}(u))$ equals 1 if a type-c job is in service, and equals zero otherwise. Since the increasing process associated to Z_s is bounded in mean by Ct, with $C > 0$, it follows that $\sup_t E(\frac{Z_s(rt)}{r})^2 \leq \frac{Ctr}{r^2} = \frac{Ct}{r}$, which in turn implies that $\frac{Z_s(rt)}{r} \to 0$ almost surely. Now together with Eq. (3.17), we conclude that for this subsequence r_j of r,

$$\lim_{j\to\infty}\frac{|M_{s,B}^{(0),r_j}(r_jt)|}{r_j} \quad = \quad \lambda\frac{dt}{K} - \bar{\ell}\mu\frac{d}{K}(t-T_0) = (\lambda-\bar{\ell}\mu)\frac{d}{K}t + T_0\bar{\ell}\mu\frac{d}{K},$$

where t is such that $T_0 \leq t \leq T_0 + \tilde{T}_1 - \epsilon$. Hence, the corresponding fluid limit satisfies

$$m_{s,B}(t) = (\lambda-\bar{\ell}\mu)\frac{d}{K}t + T_0\bar{\ell}\mu\frac{d}{K}, \tag{3.25}$$

where t is such that $T_0 \leq t \leq T_0 + \tilde{T}_1 - \epsilon$. In case $\tilde{T}_1 > T_1$, since $T_1 = T_0\frac{\lambda}{\bar{\ell}\mu-\lambda}$, one obtains from Eq. (3.25) that $m_{s,B}(T_0+T_1) = 0$ for all s. Now assume $\tilde{T}_1 \leq T_1$. By definition of T_1^r, it holds that $\prod_s M_{s,B}^{(0),r}(r(T_0+T_1^r)) = 0$ and hence $\prod_s m_{s,B}(T_0+\tilde{T}_1) = 0$. From Eq. (3.25) one has $m_{s,B}(T_0+\tilde{T}_1-\epsilon) = m_{\tilde{s},B}(T_0+\tilde{T}_1-\epsilon)$, for any $s, \tilde{s}$ and any $\epsilon > 0$. This, together with the fact that a fluid limit $\vec{n}_B(t)$ is a continuous function and $\prod_s m_{s,B}(T_0+\tilde{T}_1) = 0$, it follows that $m_{s,B}(T_0+\tilde{T}_1) = 0$ for all s and hence also $m_{s,B}(T_0+T_1) = 0$ since $\tilde{T}_1 \leq T_1$.

We conclude that at time $T_0 + T_1$, for any fluid limit $\vec{n}(\cdot)$ of $\vec{N}^{(0),r}(\cdot)$, it holds

that $\vec{n}(T_0 + T_1) = \vec{0}$. From Theorem 2.4.10, we conclude that the process $\vec{N}^{(0)}(t)$ is ergodic. □

C: Proofs of Section 3.4

Proof of Lemma 3.4.3: We couple the two systems as follows: at time zero, start in the same initial state. Arrivals are coupled in both systems. Below it will become clear how the departures are coupled under both systems.

Assume that at time $t \geq 0$, $N_c^{PS}(t) \geq N_c^{LB}(t)$ for all $c \in \mathcal{C}$. We prove that this remains valid at time t^+. We only need to analyse states such that $N_c^{PS}(t) = N_c^{LB}(t) = n_c$, for some $c \in \mathcal{C}$. Under this situation, note that $M^{PS}_{s^*_{ci}(t)}(t) \geq M^{PS}_{s_c^{min}(\vec{N}^{PS}(t))}(t) \geq M^{LB}_{s_c^{min}(\vec{N}^{PS}(t))}(t) \geq M^{LB}_{s_c^{min}(\vec{N}^{LB}(t))}(t)$ for all $i = 1, \ldots, N_c^{PS}(t)$. Hence, the departure rate of type-c jobs in the PS system, $\mu \sum_{i=1}^{N_c^{PS}(t)} \frac{1}{M^{PS}_{s^*_{ci}(t)}(t)}$, is smaller than or equal to that in the LB-system, $\mu \frac{N_c^{LB}(t)}{M^{LB}_{s_c^{min}(\vec{N}^{LB}(t))}(t)}$. We can therefore couple the systems such that if there is a type-c departure in the original PS model, then also a type-c departure occurs in the LB-system. Since arrivals are coupled in both systems, it follows directly that at time t^+, $N_c^{PS}(t^+) \geq N_c^{LB}(t^+)$. □

Proof of Lemma 3.4.4: For simplicity in notation, we remove the superscript LB throughout the proof. From Eq. (3.7), we have that the departure rate of $M_s(t)$ is given by

$$\sum_{c \in \mathcal{C}(s)} \frac{N_c(t)}{M_{s_c^{min}(\vec{n})}(t)}. \tag{3.26}$$

Recall that $c \in \mathcal{C}_l^s(\vec{n})$ if server l is the server with the minimum number of copies that serves a type-c job. Hence, if $c \in \mathcal{C}_l^s(\vec{n})$, then $s_c^{min}(\vec{n}) = l$. Since $\mathcal{C}(s) = \cup_{l \in D_s(\vec{n})} \mathcal{C}_l^s(\vec{n})$, Eq. (3.26) can be written as

$$\sum_{l \in D_s(\vec{n})} \frac{\sum_{c \in \mathcal{C}_l^s(\vec{n})} N_c(t)}{M_l(t)}. \tag{3.27}$$

Using that $\sum_{c \in \mathcal{C}_s^s(\vec{n})} N_c(t)$ can be written as $M_s(t) - \sum_{l \in D_s(\vec{n}), l \neq s} \sum_{c \in \mathcal{C}_l^s(\vec{n})} N_c(t)$, we obtain that Eq. (3.27) is equal to

$$\sum_{l \in D_s(\vec{n}), l \neq s} \frac{\sum_{c \in \mathcal{C}_l^s(\vec{n})} N_c(t)}{M_l(t)} + 1 - \sum_{l \in D_s(\vec{n}), l \neq s} \frac{\sum_{c \in \mathcal{C}_l^s(\vec{n})} N_c(t)}{M_s(t)}$$
$$= 1 + \sum_{l \in D_s(\vec{n})} \frac{(M_s(t) - M_l(t)) \sum_{c \in \mathcal{C}_l^s(\vec{n})} N_c(t)}{M_s(t) M_l(t)}.$$

Proof of Lemma 3.4.5: For ease of notation, we removed the superscript LB throughout the proof.

Let $f(\vec{n}) = (f_c(\vec{n}), c \in \mathcal{C})$, with $f_c(\vec{n}) : \mathbb{R}_+^{|\mathcal{C}|} \to \mathbb{R}^{|\mathcal{C}|}$, denote the drift vector field of $\vec{N}(t)$ when starting in state $\vec{N}(0) = \vec{n}$, i.e. $f(\vec{n}) = \frac{d}{dt}\mathbb{E}^{\vec{n}}[\vec{N}(t)]_{t=0}$. Recall that the fluid limit can be characterized as in Eq. (3.13) and Eq. (3.14). We want to partly characterize the fluid process $m_s(t) = \sum_{c\in\mathcal{C}(s)} n_c(t)$. We denote by $\tilde{f}_s(\vec{n}) = \sum_{c\in\mathcal{C}(s)} f_c(\vec{n})$ the drift of $M_s(t)$.

From Lemma 3.4.4, we can write the drift of $M_s(\cdot)$, starting in state $\vec{N}(0) = \vec{n}$, as

$$\tilde{f}_s(\vec{n}) = \lambda\frac{d}{K} - \mu\mathbf{1}_{(m_s>0)} - \mu\mathbf{1}_{(m_s>0)}\left(\sum_{l\in D_s(\vec{n})} \frac{(m_s - m_l)\sum_{c\in\mathcal{C}_l^s(\vec{n})} n_c}{m_s m_l}\right), \tag{3.28}$$

where $D_s(\vec{n}) = \{l \in S : m_s \geq m_l\}$ is the set of servers that have less than or equal number of copies, compared to server s, in state $\vec{n}$.

Let $G_1(\vec{n}) := \{s \in S : m_s \leq m_l, \forall l\}$. If $s \in G_1(\vec{n})$ and $\lim_{r\to\infty}\sum_{c\in\mathcal{C}(s)} n_c^r = \lim_{r\to\infty} m_s^r > 0$, it follows from Eq. (3.28) that

$$\lim_{r\to\infty} \tilde{f}_s(r\vec{n}^r) = \lambda d/K - \mu.$$

If instead $s \in G_1(\vec{n})$ and $\lim_{r\to\infty} m_s^r = 0$, then

$$conv\left(acc_{r\to\infty}\ \tilde{f}_s(r\vec{n}^r)\ \ with\ \ \lim_{r\to\infty}\vec{n}^r = \vec{n}\right) = conv(\lambda d/K - \mu, \lambda d/K).$$

Combining Eq. (3.13) and $\sum_{c\in\mathcal{C}(s)} \frac{\mathrm{d}n_c(t)}{\mathrm{d}t} = \frac{\mathrm{d}m_s(t)}{\mathrm{d}t}$, we conclude the proof. □

Proof of Proposition 3.4.7: From Lemma 3.4.3 we have that if the lower-bound system is unstable, then also the original PS system. Hence, to prove Proposition 3.4.7, it will be enough to prove that $\vec{N}^{LB}(t)$ is unstable if $\rho > 1/d$. This is done in the remainder of the proof.

For ease of notation, we remove the superscript LB throughout the proof. To prove that the system is transient, below we will show that there is a subsequence of t such that the system $\vec{N}(t)$ converges towards $+\infty$.

Define $m_{min}(t) := \min_{s\in S}\{m_s(t)\}$ and fix $T = (|\vec{n}| + \delta)/(\lambda d/K - \mu)$, for some $\delta > 0$. From Lemma 3.4.5, we know that at time T, $m_{min}(T) \geq |\vec{n}| + \delta$, when $\vec{n}(0) = \vec{n}$. Hence, as well,

$$|\vec{n}(T)| \geq m_{min}(T) \geq |\vec{n}| + \delta. \tag{3.29}$$

For almost all sample paths, and any subsequence r_k of r, there exists a further subsequence r_{k_j} such that $\lim_{j\to\infty} \frac{|\vec{N}^{r_{k_j}}(r_{k_j}T)|}{r_{k_j}} = |\vec{n}(T)| \geq |\vec{n}| + \delta$, with $\vec{N}^r(0) = r\vec{n}$ and $\vec{n}(t)$ a fluid limit (the inequality follows from Eq. (3.29)). Hence, when considering the

liminf subsequence, this gives, for all $\vec{n}$,

$$\liminf_{r\to\infty} \left| \frac{\vec{N}^r(rT)}{r} \right| \geq |\vec{n}| + \delta,$$

where $\vec{N}^r(0) = r\vec{n}$. From Fatou's lemma, this implies

$$\liminf_{r\to\infty} \mathbb{E}\left(\frac{\vec{N}^r(rT)}{r} \right) \geq |\vec{n}| + \delta.$$

Hence, there exists $r_0(\vec{n}) \geq 1$, such that $\vec{N}^{r_0(\vec{n})}(0) = r_0(\vec{n})\vec{n}$ and

$$\mathbb{E}\left(\vec{N}^{r_0(\vec{n})}(r_0(\vec{n})T) \right) \geq r_0(\vec{n}) \left(|\vec{n}| + \delta - \epsilon \right), \tag{3.30}$$

for some ϵ, with $0 < \epsilon < \delta$. Now, for any $\vec{n}$, define the discrete time stochastic process $(\vec{Y}_l, \vec{Z}_l)$, $l \geq 0$:

$$\begin{aligned}
&\vec{Z}_0 = \vec{n}, \\
&\vec{Y}_{l+1} = \vec{N}^{r_0(\vec{Z}_l)}(r_0(\vec{Z}_l)T), \text{ where } \vec{N}^{r_0(\vec{Z}_l)}(0) = r_0(\vec{Z}_l)\vec{Z}_l, \\
&\vec{Z}_{l+1} = \vec{Y}_{l+1} r_0(\vec{Z}_l), l \geq 0.
\end{aligned}$$

Observe that:

1. $(\vec{Y}_l, \vec{Z}_l)$ is Markov, since $\vec{N}$ is a Markov process.
2. It follows from Eq. (3.30) that $\mathbb{E}\Big(|\vec{Z}_{l+1}| \Big| \vec{Z}_l \Big) - |\vec{Z}_l| \geq \delta - \epsilon > 0$, $l \geq 0$.
3. Using Dynkins formula for the continuous time process $\vec{N}(t)$, we see that

$$\begin{aligned}
\mathbb{E}\Big(\mathbb{E}\left(|\vec{Z}_{l+1}| \right) - |\vec{Z}_l| \Big) &= \mathbb{E}\Big(|\vec{Z}_l| + \frac{1}{r_0(\vec{Z}_l)} \int_0^{r_0(\vec{Z}_l)T} a(\vec{N}_s) ds - |\vec{Z}_l| \Big) \\
&= \mathbb{E}\Big(\frac{1}{r_0(\vec{Z}_l)} \int_0^{r_0(\vec{Z}_l)T} a(\vec{N}_s) ds \Big),
\end{aligned}$$

where $a(\cdot)$ is the drift of the norm function. Note that given the model (bounded rates of arrival and departures) this drift is a bounded function (say by γ), which implies that

$$\mathbb{E}\left(|\vec{Z}_{l+1}| - |\vec{Z}_l| \right) \leq \mathbb{E}\left(\mathbb{E}\left(\frac{\gamma r_0(\vec{Z}_l)T}{r_0(\vec{Z}_l)} \Big| \vec{Z}_l \right) \right) = \gamma T < \infty.$$

Using a typical transience criterion for Markov chains, (see for instance Proposition 8.9 in [87]), we obtain that Z_l cannot be stable. This in turn directly implies that $\vec{N}(t)$

converges along one subsequence of t towards $+\infty$, which implies that it cannot be stable. □

Proof of Lemma 3.4.8: We couple both systems as follows: at time zero, we start in the same initial state. Arrivals and their service requirements are coupled in both systems.

The result will be proved by induction. It holds at time 0. Now assume that for all $u \leq t$ it holds that $\alpha_{i,s}^{UB}(u) \leq \alpha_{i,s}^{PS}(u)$ for all $i = 1, \ldots,$ and $s \in S$. Below we prove that this remains true at time t^+.

Let $c(i)$ denote the type of the i-th arrived job and let $\tilde{A}(t)$ denote the number of arrivals until time t. Assume there is an $i \leq \tilde{A}(t)$ and $s \in c(i)$ such that $\alpha_{i,s}^{UB}(t) = \alpha_{i,s}^{PS}(t)$. Note that for all c,

$$N_c^{UB}(t) = \sum_{j=1}^{\tilde{A}(t)} \mathbf{1}_{\{c=c(j)\}} \mathbf{1}_{\{\exists \tilde{s} \in c(j), \text{ s.t. } \alpha_{j,\tilde{s}}^{UB}(t) < \beta_j\}}$$

and

$$N_c^{PS}(t) = \sum_{j=1}^{\tilde{A}(t)} \mathbf{1}_{\{c=c(j)\}} \mathbf{1}_{\{\forall \tilde{s} \in c(j), \alpha_{j,\tilde{s}}^{PS}(t) < \beta_j\}}.$$

Since $\alpha_{j,\tilde{s}}^{UB}(t) \leq \alpha_{j,\tilde{s}}^{PS}(t)$, for all $j, \tilde{s}$, it follows that $N_c^{UB}(t) \geq N_c^{PS}(t)$, for all c, hence $M_{\tilde{s}}^{UB}(t) \geq M_{\tilde{s}}^{PS}(t)$ for all servers $\tilde{s}$. In particular, this implies that $\frac{\mathrm{d}\alpha_{i,s}^{UB}(t)}{\mathrm{d}t} = \frac{1}{M_s^{UB}(t)} \leq \frac{1}{M_s^{PS}(t)} = \frac{\mathrm{d}a_{i,s}^{PS}(t)}{\mathrm{d}t}$, for $s \in c(i)$, which together with $\alpha_{i,s}^{UB}(t) = \alpha_{i,s}^{PS}(t)$ gives that $\alpha_{i,s}^{UB}(t^+) \leq \alpha_{i,s}^{PS}(t^+)$ holds at time t^+. □

D: Proofs of Section 3.5

Proof of Lemma 3.5.1: For ease of notation, we remove the superscript ROS throughout the proof. Assume at time 0 we are in state $\vec{N}(0) = \vec{N}$. We first derive $P_s(\vec{N})$, the probability that at time $t = 0$ a given server s is serving a copy that is not in service in any other server. In order to derive that, we consider $P_s(\vec{N}|c)$ defined as the probability that server s is serving a type-c job, $s \in c$, and this job is not in service in any other server.

Let $-\tilde{T}_s < 0$ denote the time that server s started working on the copy which it is serving at time 0. When the server becomes idle, it chooses a copy uniformly at random. Hence, the probability that a copy from a type-c job is being served in server s is given by $\frac{N_c(-\tilde{T}_s)}{M_s(-\tilde{T}_s)}$. Using the law of total probability, we have

$$P_s(\vec{N}) = \sum_{c \in \mathcal{C}(s)} \frac{N_c(-\tilde{T}_s^r)}{M_s(-\tilde{T}_s^r)} P_s(\vec{N}|c). \tag{3.31}$$

To calculate $P_s(\vec{N}|c)$, note that $\frac{N_c(-\tilde{T}_l^r)-1}{M_l(-\tilde{T}_l^r)}$ is the probability that server l is *not* serving the type-c copy that is now in service in server s, with $l, s \in c$. Hence,

$$P_s(\vec{N}|c) = \Pi_{l\in c, l\neq s} \frac{M_l(-\tilde{T}_l^r) - 1}{M_l(-\tilde{T}_l^r)}, \ s \in c. \tag{3.32}$$

We now characterize the fluid limits, which we recall can be characterized as in Eq. (3.13) and Eq. (3.14). Let $f(\vec{n}) = (f_c(\vec{n}), c \in \mathcal{C})$, with $f_c(\vec{n}) : \mathbb{R}_+^{|\mathcal{C}|} \to \mathbb{R}^{|\mathcal{C}|}$, denote the drift vector field of $\vec{N}(t)$ when starting in state $\vec{N}(0) = \vec{n}$, i.e., $f(\vec{n}) = \frac{d}{dt}\mathbb{E}^{\vec{n}}[\vec{N}(t)]_{t=0}$. Hence, we study the fluid drift in points $r\vec{n}^r$, where $\lim_{r\to\infty} r\vec{n}^r = \vec{n}$. That is, $\vec{N}(0) = r\vec{n}^r$.

Since the transition rates μ and λ are of order $O(1)$, it follows directly that $\tilde{T}_s^r$ and $\vec{N}(-\tilde{T}_s^r) - \vec{N}(0)$ are of order $O(1)$ as well, so that

$$\lim_{r\to\infty} \frac{N_c(-\tilde{T}_s^r)}{M_s(-\tilde{T}_s^r)} = \lim_{r\to\infty} \frac{N_c(0)}{M_s(0)} = \frac{n_c(0)}{m_s(0)} \text{ and } \lim_{r\to\infty} \frac{M_l(-\tilde{T}_l^r) - 1}{M_l(-\tilde{T}_l^r)} = 1. \tag{3.33}$$

It hence follows from Eq. (3.31) and Eq. (3.32) that if $\lim_{r\to\infty} \sum_{c\in\mathcal{C}(s)} rn_c^r > 0$ then, $\lim_{r\to\infty} P_s(r\vec{n}^r) = 1$. □

Proof of Lemma 3.5.2: For ease of notation, we remove the superscript *ROS* throughout the proof. We denote by $\tilde{f}_s(\vec{n}) = \sum_{c\in\mathcal{C}(s)} f_c(\vec{n})$ the one-step drift of $M_s(t)$. When starting in state $\vec{N}(0) = r\vec{n}^r$, the latter is in the limit equal to

$$\lim_{r\to\infty} \tilde{f}_s(r\vec{n}^r) = \lambda\frac{d}{K} - \mu\left(\sum_{c\in\mathcal{C}(s)}\sum_{l\in c}\frac{n_c}{m_l}\lim_{r\to\infty}(P_l(r\vec{n})) - \lim_{r\to\infty}(g_{c,l,s}(r\vec{n}^r)(1 - P_l(r\vec{n})))\right) \tag{3.34}$$

with $g_{c,l,s} = O(1)$. Note that the first term multiplied by μ in Eq. (3.34) represents departures of type-c jobs, $c \in \mathcal{C}(s)$, who were served in one unique server. Here $\frac{n_c}{m_l}$ represents the probability (in the limit) that a copy from type c is being served in server s, see Eq. (3.33). The second term multiplied by μ in Eq. (3.34) represents departures due to a type-c job that is being served in more than one server simultaneously. Together with Lemma 3.5.1, Eq. (3.11), we obtain

$$\lim_{r\to\infty} \tilde{f}_s(r\vec{n}^r) = \lambda\frac{d}{K} - \mu\sum_{c\in\mathcal{C}(s)}\sum_{l\in c}\frac{n_c}{m_l}. \tag{3.35}$$

Now, note that Eq. (3.35) is equal to Eq. (3.15) (the fluid drift for the ROS model with i.i.d. copies). Hence, the proof now follows as in the proof of Lemma 3.2.2. □

E: Proofs of Section 3.6.2.2

Light-traffic approximation

In the present we calculate the light-traffic approximation of order 1, where the light-traffic approximation was introduced in Section 2.5 and the $n = 1$ order approximation is given in Eq. (2.4). We note that for the ease of notation we drop the dependency on P of $\bar{D}^{(n)}(0, b)$. Hence, we will calculate the sojourn time of the tagged job conditioned on, at most, having one other job present in the system. Let $\tilde{A}(t_0, t_1)$ denote the number of arrivals in the time interval $[t_0, t_1)$ in addition to the tagged job who is assumed to arrive at time 0. The zeroth and first light-traffic derivatives satisfy, Eq. (2.5) and Eq. (2.6) (in Section 2.5):

$$\bar{D}^{(0)}(0, b) := \mathbb{E}\left(\bar{D}(0, b)|\tilde{A}(-\infty, \infty) = 0\right) = b \tag{3.36}$$

and

$$\begin{aligned}\bar{D}^{(1)}(0, b) \quad := \quad & \frac{1}{\binom{K}{d}} \int_{-\infty}^{\infty} \Big[\mathbb{E}\left(\bar{D}(0, b)|\tilde{A}(-\infty, \infty) = 1, \tau = t, c_1 = c\right) \\ & -\mathbb{E}\left(\bar{D}(0, b)|\tilde{A}(-\infty, \infty) = 0\right)\Big] \, \mathrm{d}t \end{aligned} \tag{3.37}$$

where τ is the arrival time of the other job. There Eq. (3.36) is straightforward, since only the tagged job is present, and all copies of this job are equal to b. There, Eq. (3.37) is obtained applying the following reasoning in Eq. (2.6): when in addition to the tagged job, another job is present, the delay of the tagged job will depend on the type of the job already in the system, denoted by c_1. If both jobs are of a different type, the new job will start being served immediately, and hence the first term in the integral of $\overline{D}^{(1)}(0, b) = b$, that is, the first light-traffic derivative is equal to zero. On the other hand, if both jobs have the same type, which happens with probability $\frac{1}{\binom{K}{d}}$, the job that is already in the system will have an impact on the sojourn time of the tagged job. We note that the precise value of the impact will depend on P, which we quantify later on in the proof of Lemma 3.6.1. We note that in Eq. (3.37), the first term is conditioned on the job being of the same type as the tagged job.

We note that if the scheduling policy does not depend on d, then neither does $\mathbb{E}(\bar{D}(0, b)|\tilde{A}(-\infty, \infty) = 1, \tau = t, c_1 = c)$, hence the light-traffic approximation of order 1 is minimized when d is set equal to $d^* = \lfloor K/2 \rfloor$.

Proof of Lemma 3.6.1: In order to obtain an expression for $\bar{N}^{LT,P}(\lambda)$, we will calculate $\bar{D}^{LT,P}(\lambda, b)$, uncondition on b and then apply Little's law. By Eq. (3.36) and Eq. (3.37), calculating $\bar{D}^{LT,P}(\lambda, b)$ reduces to calculating $\mathbb{E}\left(\bar{D}(0, b)|\tilde{A}(-\infty, \infty) = 1, \tau = t, c_1 = c\right)$. Below we do so for exponentially distributed service requirements and with P equal to PS, FCFS, or ROS.

First consider FCFS. If in addition to the tagged job, another job arrives in the interval $(-\infty, \infty)$, which is of the same type $c_1 = c$ and has service time $B_1 = b_1$, then the sojourn time of the tagged job will be given by

$$\mathbb{E}\left(\bar{D}(0,b)|\tilde{A}(-\infty,\infty) = 1, \tau = t, B_1 = b_1, c_1 = c\right) = \begin{cases} b, & \text{if } t \leq -b_1, \\ t + b_1 + b, & \text{if } -b_1 \leq t \leq 0, \\ b, & \text{if } t \geq 0. \end{cases}$$

For example, the second equation is the case where the other job arrives before the tagged job, and has still $b_1 + t$ remaining service left. Hence, the tagged job has to wait $b_1 + t$, so that its sojourn time equals $b + b_1 + t$. To calculate $\bar{D}^{(1)}(0,b)$, we subtract from the above $\bar{D}^{(0)}(0,b) = b$ and we multiply with $\frac{1}{\binom{K}{d}}$, integrate over t, and uncondition on the service requirements b_1. Further unconditioning on b gives $\frac{3\lambda}{2\mu^2}\frac{1}{\binom{K}{d}}$. On the other hand, unconditioning $\bar{D}^{(0)}(0,b)$ over b readily yields $1/\mu$. Summing both terms, we get $\bar{D}^{LT,FCFS}(\lambda) = \frac{1}{\mu} + \frac{3\lambda}{2\mu^2}\frac{1}{\binom{K}{d}}$, and multiplying by λ (Little's law) we obtain the expression for $\bar{N}^{LT,FCFS}(\lambda)$.

The analysis of FCFS carries directly over to ROS, since at most two jobs are considered to be present in the system, in which case jobs under ROS will be served in order of arrival.

We now consider PS. We consider the case where in addition to the tagged job, another job arrives in the system in the interval $(-\infty, \infty)$. Let b_1 denote the service of this other job and c_1 its type. We have

$$\mathbb{E}\left(\bar{D}(0,b)|\tilde{A}(-\infty,\infty) = 1, \tau = t, B_1 = b_1 c_1 = c\right) = \begin{cases} b & \text{if } t \leq -b_1, \\ b + b_1 + t & \text{if } -b_1 \leq t \leq -b_1 + b \text{ and } b \leq b_1, \\ 2b & \text{if } -b_1 + b \leq t \leq 0 \text{ and } b \leq b_1, \\ b + b_1 + t & \text{if } -b_1 \leq t \leq 0 \text{ and } b \geq b_1, \\ b + b_1 & \text{if } 0 \leq t \leq b - b_1 \text{ and } b \geq b_1, \\ 2b - t & \text{if } b - b_1 \leq t \leq b \text{ and } b \geq b_1, \\ 2b - t & \text{if } 0 \leq t \leq b \text{ and } b \leq b_1, \\ b & \text{if } t \geq b. \end{cases}$$

The expression above takes into account all the possible events. For example, the first equation is the case when the other job arrives and leaves before the tagged job arrives. The second equation is the case where the other job arrives before the tagged job and leaves first. In that case, the sojourn time experienced by the tagged job is b plus the capacity spend on serving the other job $b_1 - (-t)$. The third equation is the case where

the other job arrives before the tagged job and leaves after the tagged job. In that case, the tagged job has shared during its whole stay the server, hence its sojourn time equals $2b$.

To calculate $\bar{D}^{(1)}(0,b)$, we combine all the cases, subtract $\bar{D}^{(0)}(0,b) = b$ and multiply by $\frac{1}{\binom{K}{d}}$, integrate over t and uncondition over b_1. Then, further unconditioning on b we get the expression $\frac{\lambda}{\mu^2}\frac{1}{\binom{K}{d}}$. As in the case of FCFS, summing now with $1/\mu$, we get $\bar{D}^{LT,PS}(\lambda) = \frac{1}{\mu} + \frac{\lambda}{\mu^2}\frac{1}{\binom{K}{d}}$. Multiplying by λ yields $\bar{N}^{LT,PS}(\lambda)$. □

Chapter 4

Stability with a general redundancy topology

In the present chapter, we characterize the stability condition for redundancy systems with a general topology and heterogeneous server capacities, where jobs have identical copies. That is, we generalize the results in Chapter 3 for the redundancy-d system where jobs have exponentially distributed service times, to a general redundancy topology and general service time distributions. Chapter 3, serves as a basis to understand the performance when the system is heterogeneous.

More precisely, we investigate the stability condition when servers implement PS. We characterize the stability condition by a recursion, and observe that this condition coincides with that of a system where each job type only dispatches copies into its least-loaded servers, and those copies need to be fully served. We also discuss the stability conditions when either FCFS or ROS is implemented. The latter are open problems and are further discussed in Chapter 8.

Through a numerical analysis, we investigate the performance of the redundancy systems under the three scheduling policies. We consider both low and high variable service time distributions and observe that under FCFS and ROS the performance degrades when the service time distribution is highly variable, whereas under PS the performance seems to be nearly insensitive to the service time distribution.

The rest of the chapter is organized as follows: In Section 4.1 we give a detailed description of the model. In Section 4.2 we present the stability results for PS with identical copies for a general redundancy topology. In Section 4.3 we provide a discussion of the stability condition under FCFS and ROS. Section 4.4 provides insights on the performance of redundancy systems through simulations. All proofs are deferred to Appendix 4.6, for the sake of readability. Several proof techniques in the present chapter are based on those from Chapter 3.

We further refer to Chapter 5 where we take advantage of the results in this chapter in order to investigate when adding redundancy can be beneficial from the stability point of view. That is, we characterize when redundancy can improve the stability region, and hence the performance under high loads, with respect to Bernoulli routing system.

4.1 Model description

We consider a K parallel-server system with heterogeneous capacities μ_s, for $s \in S$, where S denotes the set of all servers, $S = \{1, \ldots, K\}$. Each job will be assigned a type label, $c = \{s_1, \ldots, s_n\}$, with $s_1, \ldots, s_n \in S, s_i \neq s_j, i \neq j$ and $n \leq K$, to indicate the n servers to which a copy is sent. We let $\mathcal{C}$ be the set of all the types, that is, $\mathcal{C}$ determines the redundancy topology of the system. We denote by $|\mathcal{C}|$ the total number of types. An arriving job is of type $c \in \mathcal{C}$ with probability p_c, where $\sum_{c\in\mathcal{C}} p_c = 1$. We denote by $\mathcal{C}(s)$ the subset of types that are served at server s, that is, $\mathcal{C}(s) = \{c \in \mathcal{C} \ : \ s \in c\}$.

In this chapter, we assume that the service time distribution is a random variable X with unit mean and distribution function F that has no atoms and is light-tailed. This assumption is needed, see [73] and [81], in order to apply fluid limits arguments in the context of processor sharing networks. To be more specific, we refer to F as light-tailed if the following is satisfied:

$$\lim_{r\to\infty} \sup_{a\geq 0} \mathbf{E}[(X-a)1_{\{X-a>r\}}|X > a] = 0. \tag{4.1}$$

It can be seen (as observed in [81]) that Eq. (4.1) also implies

$$\sup_{a\geq 0} \mathbf{E}[(X-a)|X > a] \leq \Phi < \infty, \tag{4.2}$$

which is a usual light-tailed condition (see [38]). Hence, Eq. (4.1) and Eq. (4.2), though exclude heavy-tailed distributions like Pareto, include large sets of distributions as phase-type (which are dense in the set of all distributions on $\mathbb{R}^+$), exponential and hyper-exponential distributions, as well as distributions with bounded support.

We denote by $N_c(t)$ the number of type-c jobs that are present in the redundancy system at time t and $\vec{N}(t) = (N_c(t), c \in \mathcal{C})$. Furthermore, we denote the number of copies per server by $M_s(t) := \sum_{c\in\mathcal{C}(s)} N_c(t)$, $s \in S$, and $\vec{M}(t) = (M_1(t), \ldots, M_K(t))$. For the j-th type-c job, let b_{cj} denote the service requirement of this job, for $j = 1, \ldots, N_c(t)$, $c \in \mathcal{C}$.

Let $a_{cjs}(t)$ denote the attained service in server s of the j-th type-c job at time t. We denote by $A_c(t) = (a_{cjs}(t))_{js}$ a matrix on $\mathbb{R}_+$ of dimension $N_c(t) \times |c|$. Note that the number of type-c jobs increases by one at rate λp_c, which implies that a row

composed of zeros is added to $A_c(t)$. When one element $a_{cjs}(t)$ in matrix $A_c(t)$ reaches the required service b_{cj}, the corresponding job departs and all of its copies are removed from the system. Hence, row j in matrix $A_c(t)$ is removed.

In the present chapter, for a given system where servers implement policy P, we characterize $\lambda^{R,P}$ as the value of λ such that the redundancy model is stable if $\lambda < \lambda^{R,P}$ and unstable if $\lambda > \lambda^{R,P}$. We study scheduling policies FCFS, PS and ROS. As to distinguish between the different policies, we will add a superscript $\{FCFS, PS, ROS\}$ to the process $\vec{N}(t)$.

4.2 The PS scheduling policy

In order to characterize the stability condition of the system under PS and identical copies, we define the *load* of a server in a subsystem:

Definition 4.2.1. For any given set of servers $\tilde{S} \subseteq S$ and its associated set of job types $\tilde{\mathcal{C}} = \{c \in \mathcal{C} \ : \ c \subseteq \tilde{S}\}$, the *load* of server $s \in \tilde{S}$ in this so-called $\tilde{S}$-subsystem is defined by

$$\frac{\lambda \sum_{c \in \tilde{\mathcal{C}}(s)} p_c}{\mu_s},$$

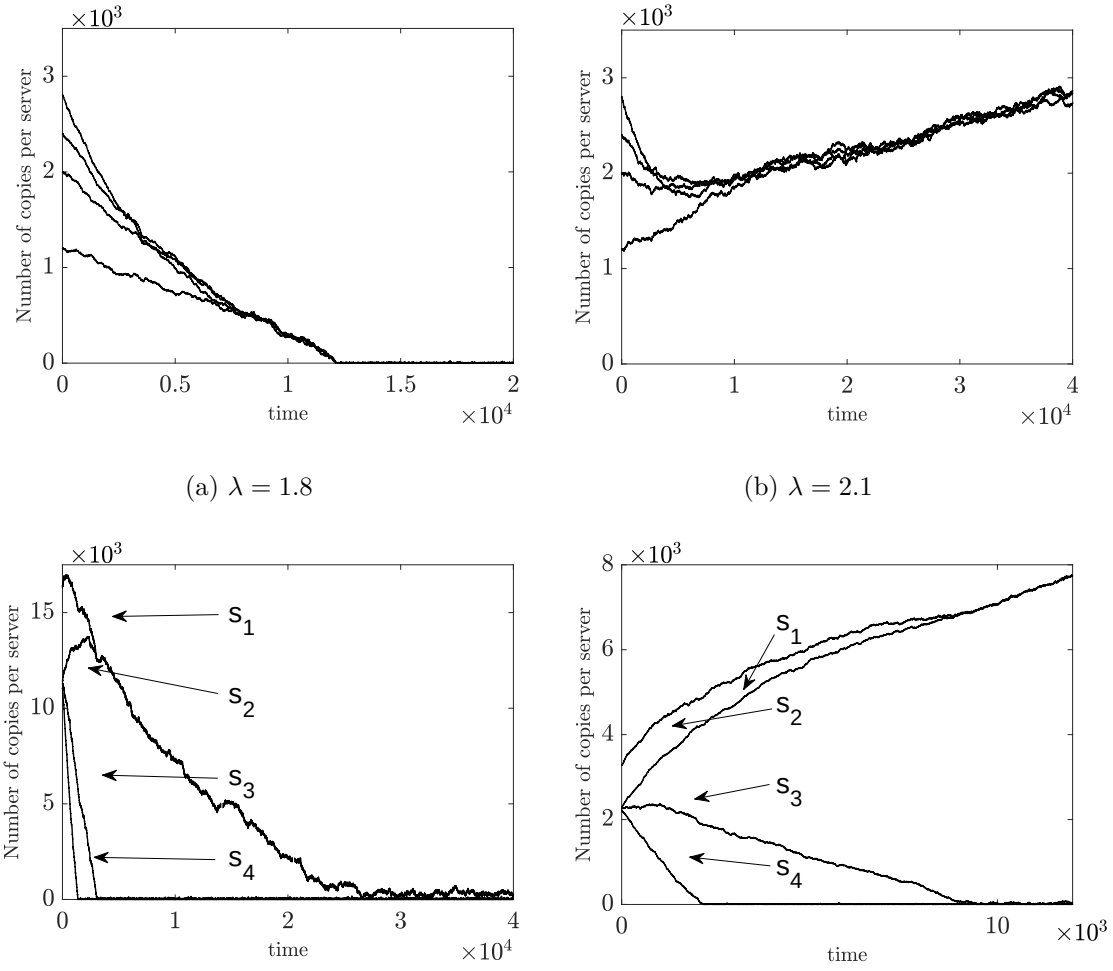
where $\tilde{\mathcal{C}}(s) = \tilde{\mathcal{C}} \cap \mathcal{C}(s)$ is the subset of types in $\tilde{\mathcal{C}}$ that are served in server s.

4.2.1 An illustrative example

In the following, we first illustrate through a numerical example some of the key aspects of our proof, and in particular the essential role played by the load defined in Definition 4.2.1. In Figure 4.1 we plot the trajectories of the number of copies per server with respect to time for a $K = 4$ server where each job dispatches two copies, that is, $\mathcal{C} = \{\{1,2\},\{1,3\},\{1,4\},\{2,3\},\{2,4\},\{3,4\}\}$. Our proof techniques will rely on fluid limits, and therefore we chose large initial points. Figures 4.1 (a) and (b) show the trajectories for the redundancy-2 model (that is, $p_c = 1/\binom{K}{d}$ for all $c \in \mathcal{C}$ and $\mu_k = \mu$ for $k = 1, \ldots, 4$) for $\lambda = 1.8$ and $\lambda = 2.1$, respectively. Figures 4.1 (c) and (d) consider a heterogeneous system (parameters see the legend) for $\lambda = 7.5$ and $\lambda = 9$, respectively.

The homogeneous example (Figure 4.1 (a) and (b)) falls within the scope of Section 3.4. There we show that the stability condition is $\lambda < \frac{\mu K}{d}$. In Figure 4.1 (a) and (b), the value for λ is chosen such that they represent a stable and an unstable system, respectively.

The behavior of the heterogeneous case is rather different. The parameters corresponding to Figures 4.1 (c) and (d) are such that the system is stable in (c), but not in (d). In Figure 4.1 (c) we see that the trajectories of all queue lengths are not always decreasing, including the maximum queue length. In Figure 4.1 (d), we observe that



(a) $\lambda = 1.8$

(b) $\lambda = 2.1$

(c) $\lambda = 7.5$

(d) $\lambda = 9$

Figure 4.1 Trajectory of the number of copies per server with respect to time for a $K = 4$ server system where each job dispatches two copies with exponentially distributed job sizes. Figures (a) and (b) consider the redundancy-2 model. Figures (c) and (d) consider heterogeneous server capacities $\vec{\mu} = (1, 2, 4, 5)$ and arrival rates per type $\vec{p} = (0.25, 0.1, 0.1, 0.2, 0.2, 0.15)$ for types $\mathcal{C}$.

the number of copies in servers 3 and 4 are decreasing, whereas those of servers 1 and 2 are increasing.

When studying stability for the heterogeneous setting, one needs to reason recursively. First, assume that each server s needs to handle its full load, i.e., $\lambda \frac{\sum_{c \in \mathcal{C}(s)} p_c}{\mu_s}$. Hence, one can simply compare the servers loads, $\lambda \sum_{c \in \mathcal{C}(s)} p_c / \mu_s$, to see which server is the least-loaded server and could hence potentially empty first. In this example, server 4 is the least-loaded server in the system, and, in fluid scale, will reach zero in finite time, and remain zero, since $\lambda \sum_{c \in \mathcal{C}(4)} p_c / \mu_4 = \lambda(p_{\{1,4\}} + p_{\{2,4\}} + p_{\{3,4\}})/5 = 0.675$ is smaller

than 1.

Whenever, at fluid scale, server 4 is still positive, the other servers might either increase or decrease. However, the key insight is that once the queue length of server 4 reaches 0, the fluid behavior of the other classes no longer depend on the jobs that also have server 4 as compatible server. That is, we are sure that all jobs that have server 4 as compatible server, will be fully served in server 4, since server 4 is in fluid scale empty and all the other servers are overloaded. Therefore, jobs with server 4 as compatible server can be ignored, and we are left with a subsystem formed by servers $\{1, 2, 3\}$ and without the job types served by server 4. Now again, we consider the load in the subsystem $\{1, 2, 3\}$ in order to determine the least-loaded server. This time, server 3 has the minimum load, which is $\lambda(p_{\{1,3\}} + p_{\{2,3\}})/4 = 0.75$. This is less than 1, it is a sufficient condition for server 3 to empty.

Similarly, once server 3 is empty, we consider the subsystem with servers 1 and 2 only. Hence, there is only one type of jobs, $\{1, 2\}$. Now server 2 is the least-loaded server and its load is $\lambda p_{\{1,2\}}/2 = 0.937$. Thus, the load in the server is less than 1, which implies that server 2 (and hence server 1, because there is only one job type) will be stable too. Indeed, in Figures 4.1 (c) we also observe that as soon as the number of copies in server 3 is relatively small compared to that of server 1 and server 2, the number of copies in both server 1 and server 2 decreases.

We can now explain the evolution observed in Figure 4.1 (d) when $\lambda = 9$. The evolution for servers 4 and 3 can be argued as before: both their loads are larger than $\lambda = 9$, hence they empty in finite time. However, the load of the subsystem with servers 1 and 2, which is 1.125, is strictly larger than 1. We thus observe that, unlike in the homogeneous case, in the heterogeneous case some servers might be stable, while others (here server 1 and 2) are unstable.

Proposition 4.2.3 formalizes the above intuitive explanation, by showing that the stability of the system can be derived recursively.

The load at every subsystem allows us now to reinterpret the homogeneous case depicted in Figure 4.1 (a) and (b). In this case, the load in the S system of all the servers is the same, which implies *(i)* that either all servers will be stable, or all unstable, and *(ii)* from the stability viewpoint is as if all copies received service until completion.

4.2.2 Stability condition

4.2.2.1 General redundancy topology

In this section we discuss the stability condition of the general redundancy topology with PS. In order to do so, we first define several sets of subsystems, similar to as what we did in the illustrative example of Section 4.2.1.

The first subsystem includes all servers, that is $S_1 = S$. We let $\mathcal{C}_1 = \mathcal{C}$. We denote by $\mathcal{L}_1$ the set of least-loaded servers in the system $S_1 = S$. Thus,

$$\mathcal{L}_1 = \left\{ s \in S_1 \ : \ s = \arg\min_{\tilde{s} \in S_1} \left\{ \frac{\lambda \sum_{c \in \mathcal{C}} p_c}{\mu_{\tilde{s}}} \right\} \right\}.$$

For $i = 2, \ldots, K$, we define recursively

$$\begin{aligned} S_i &:= S \backslash \cup_{l=1}^{i-1} \mathcal{L}_l, \\ \mathcal{C}_i &:= \{c \in \mathcal{C} \ : \ c \subset S_i\}, \\ \mathcal{C}_i(s) &:= \mathcal{C}_i \cap \mathcal{C}(s), \\ \mathcal{L}_i &:= \left\{ s \in S_i \ : \ s = \arg\min_{\tilde{s} \in S_i} \left\{ \frac{\lambda \sum_{c \in \mathcal{C}_i(\tilde{s})} p_c}{\mu_{\tilde{s}}} \right\} \right\}. \end{aligned}$$

The S_i-subsystem will refer to the system consisting of the servers in S_i, with only jobs of types in the set $\mathcal{C}_i$. The $\mathcal{C}_i(s)$ is the subset of types that are served in server s in the S_i-subsystem. The $\mathcal{L}_i$ represents the set of servers s with least-loaded servers in the S_i-subsystem. Finally, we denote by $i^* := \arg\max_{i=1,\ldots,K}\{\mathcal{C}_i \ : \ \mathcal{C}_i \neq \emptyset\}$ the last index i for which the subsystem S_i is not empty of job types.

Remark 4.2.2. *We illustrate the above definitions by applying them to the particular example considered in Section 4.2.1. The first subsystem consists of servers* $S_1 = S = \{1, 2, 3, 4\}$ *and all job types, see Figure 4.2 (a). The loads in the* S_1 *subsystem are:* $\{3.375, 2.4, 0.825, 0.675\}$*, and thus* $\mathcal{L}_1 = \{4\}$*. The second subsystem is formed by* $S_2 = \{1, 2, 3\}$ *and job types that are compatible with server 4 can be ignored, that is,* $\mathcal{C}_2 = \{\{1, 2\}, \{1, 3\}, \{2, 3\}\}$*, see Figure 4.2 (b). The loads for servers in the* S_2 *subsystem are given by* $\{2.625, 1.65, 0.75\}$*, and thus* $\mathcal{L}_2 = \{3\}$*. The third subsystem consists of servers* $S_3 = \{1, 2\}$ *and job types that are compatible with servers 3 or 4 can be ignored, that is,* $\mathcal{C}_3 = \{\{1, 2\}\}$*, see Figure 4.2 (c). The loads for servers in the* S_3 *subsystem are*

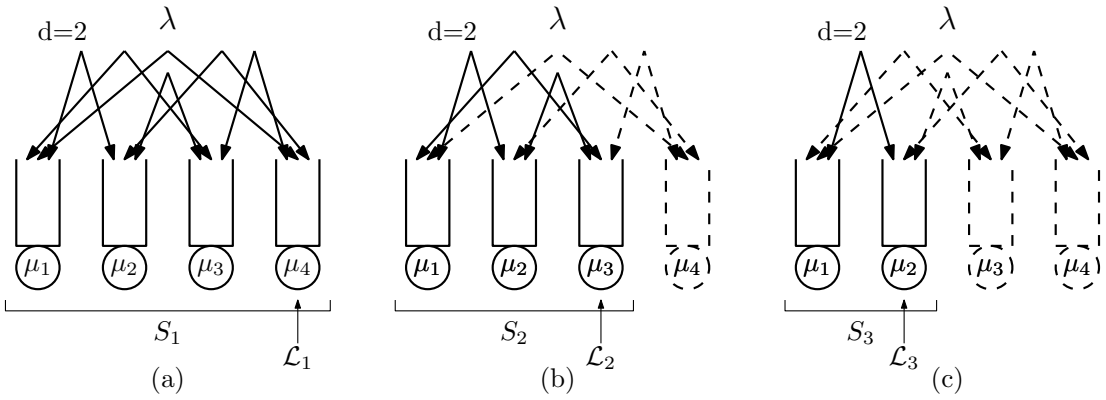


Figure 4.2 The $K = 4$ server system where each job dispatches two copies according to $\vec{p}$. In (a) subsystem S_1, in (b) subsystem S_2 and in (c) subsystem S_3.

given by $\{1.875, 0.937\}$. *Hence,* $\mathcal{L}_3 = \{2\}$. *Then,* $S_4 = \{1\}$, *but* $\mathcal{C}_4 = \emptyset$, *so that* $i^* = 3$.

In the following proposition we characterize the stability condition for servers in terms of the load corresponding to each subsystem. It states that the least-loaded servers in the S_i-subsystem can be stable if and only if all servers in subsystems $S_1, \ldots, S_{i-1}$ are stable as well. The proof can be found in Section 4.2.3.

Proposition 4.2.3. *For a given* $i \leq i^*$, *servers* $s \in \mathcal{L}_i$ *are stable if* $\lambda \sum_{c \in \mathcal{C}_i(s)} p_c < \mu_s$, *for all* $l = 1, \ldots, i$. *Servers* $s \in \mathcal{L}_i$ *are unstable if there is an* $l = 1, \ldots, i$ *such that* $\lambda \sum_{c \in \mathcal{C}_i(s)} p_c > \mu_s$.

Corollary 4.2.4. *The redundancy system is stable if* $\lambda \sum_{c \in \mathcal{C}_i(s)} p_c < \mu_s$, *for all* $s \in \mathcal{L}_i$ *and* $i = 1, \ldots, i^*$. *The redundancy system is unstable if there exists an* $\iota \in \{1, \ldots, i^*\}$ *such that* $\lambda \sum_{c \in \mathcal{C}_\iota(s)} p_c > \mu_s$ *and* $s \in \mathcal{L}_\iota$.

From the above corollary, we directly observe that the stability condition for the redundancy system coincides with the stability condition corresponding to K individual servers where each type-c job is only dispatched to its least-loaded servers.

We define by $\lambda^{R,PS}$ the value of λ such that the redundancy model is stable if $\lambda < \lambda^{R,PS}$ and unstable is $\lambda > \lambda^{R,PS}$. From the corollary, we hence obtain that the stability region under redundancy is given by

$$\lambda^{R,PS} = \min_{i=1,\ldots,i^*} \left\{ \frac{\mu_s}{\sum_{c \in \mathcal{C}_i(s)} p_c} \right\}. \tag{4.3}$$

We note that the set of stability conditions is not necessarily ordered with respect to i, for $i = 1, \ldots, i^*$.

Furthermore, we note that the stability condition is upper bounded by the condition provided by the S_1 subsystem, that is, $\lambda^{R,PS} \leq \max_{s \in S}\{\mu_s / \sum_{c \in \mathcal{C}(s)} p_c\}$.

4.2.2.2 Particular redundancy topologies

In this subsection we discuss the stability condition for some particular the redundancy topologies redundancy-d and nested.

Redundancy-d. In Section 3.4, the redundancy-d model was proved to be stable if $\lambda d < \mu K$. From Corollary 4.2.4 we obtain that this remains the stability condition for light-loaded service time distributions. We note that each server has arrival rate $\lambda d/K$ and capacity μ, thus the load at every server is $d/(K\mu)$. This implies that $\mathcal{L}_1 = S$, $\mathcal{C}_1 = 1$ and $i^* = 1$. From Corollary 4.2.4, we obtain that the system is stable if $\lambda d < \mu K$.

For the system with redundancy-d topology and heterogeneous server capacities, we have the following:

Corollary 4.2.5. *Assume the redundancy-d topology with heterogeneous server capacities $\mu_1 < \ldots < \mu_K$. The system is stable if for all $i = d, \ldots, K$, $\lambda \frac{\binom{i-1}{d-1}}{\binom{K}{d}} < \mu_i$. The system is unstable if there exists $i \in \{d, \ldots, K\}$ such that $\lambda \frac{\binom{i-1}{d-1}}{\binom{K}{d}} > \mu_i$.*

In the case of homogeneous server capacities, it is easy to deduce that the stability condition, $\lambda d < \mu K$, decreases as d increases. However, in the heterogeneous server capacities case, both the numerator and denominator are non-monotone functions of d, and as a consequence it is not straightforward how the stability condition depends on d. This dependence on d will be numerically studied in Section 5.2.1.

N-model. The simplest nested model is the N-model. We recall that this is a $K = 2$ server system with capacities $\vec{\mu} = (\mu_1, \mu_2)$ and types $\mathcal{C} = \{\{2\}, \{1, 2\}\}$, see Figure 1.1 (b). A job is of type $\{2\}$ with probability p and of type $\{1, 2\}$ with probability $1 - p$. The stability condition is $\lambda < \lambda^{R,PS}$ where:

$$\lambda^{R,PS} = \begin{cases} \mu_2, & 0 \le p \le \left(\frac{\mu_2 - \mu_1}{\mu_2}\right)^+ \\ \mu_1/(1-p), & \left(\frac{\mu_2 - \mu_1}{\mu_2}\right)^+ \le p \le \frac{\mu_2}{\mu_1 + \mu_2} \\ \mu_2/p, & \frac{\mu_2}{\mu_1 + \mu_2} < p \le 1. \end{cases}$$

The above is obtained as follows: The load of the system is $\mu_1/(1-p)$ and μ_2, respectively for server 1 and server 2. First assume $\mu_1/(1-p) > \mu_2$. Then $\mathcal{L}_1 = \{1\}$ and the second subsystem is composed of server $S_2 = \{2\}$ and $\mathcal{C}_2 = \{\{2\}\}$, with arrival rate λp to server 2. Hence the load of server 2 in the S_2-subsystem is μ_2/p. From Corollary 4.2.4, it follows that $\lambda^{R,PS} = \min\{\mu_1/(1-p), \mu_2/p\}$. On the other hand, if $\mu_1/(1-p) < \mu_2$, then $\mathcal{L}_1 = \{2\}$, and $S_2 = \{1\}$, but $\mathcal{C}_2 = \emptyset$. Thus, $\lambda^{R,PS} = \mu_2$. Lastly, if $\mu_1/(1-p) = \mu_2$, $\mathcal{L}_1 = \{1, 2\}$, thus $S_2 = \emptyset$ and $\mathcal{C}_2 = \emptyset$. Hence, $\lambda^{R,PS} = \mu_2$.

We observe that the stability condition $\lambda^{R,PS}$, is a continuous function reaching the maximum value $\lambda^{R,PS} = \mu_1 + \mu_2$ at $p = \mu_2/(\mu_1 + \mu_2)$. It thus follows that for $p = \mu_2/(\mu_1 + \mu_2)$, redundancy achieves the maximum stability condition.

W-model. The W-model is a $K = 2$ server system with capacities $\vec{\mu} = (\mu_1, \mu_2)$ and types $\mathcal{C} = \{\{1\}, \{2\}, \{1, 2\}\}$, see Figure 1.1 (c). A job is of type $\{1\}$ with probability $p_{\{1\}}$, type $\{2\}$ with probability $p_{\{2\}}$ and of type $\{1, 2\}$ with probability $p_{\{1,2\}}$. W.l.o.g., assume $(1 - p_{\{2\}})/\mu_1 \ge (1 - p_{\{1\}})/\mu_2$, that is, the load on server 1 is larger or equal to that on server 2. The stability condition is then given by:

$$\lambda^{R,PS} = \begin{cases} \mu_2/(1 - p_{\{1\}}), & p_{\{1\}} \le \frac{\mu_1}{\mu_1 + \mu_2} \\ \mu_1/p_{\{1\}}, & p_{\{1\}} \ge \frac{\mu_1}{\mu_1 + \mu_2}, \end{cases}$$

if $(1-p_{\{2\}})/\mu_1 > (1-p_{\{1\}})/\mu_2$. And,

$$\lambda^{R,PS} = \mu_2/(1-p_{\{1\}})$$

if $(1-p_{\{2\}})/\mu_1 = (1-p_{\{1\}})/\mu_2$. Similar to the N-model, the above can be obtained from Corollary 4.2.4. When $p_{\{1\}} = \mu_1/(\mu_1+\mu_2)$, maximum stability $\lambda^{R,PS} = \mu_1+\mu_2$ is obtained.

4.2.3 Proof of the stability condition

In this section, we prove that the condition in Proposition 4.2.3 is sufficient and necessary for the respective subsystem to be stable. As we observe in Section 4.2.1, there are two main issues concerning the evolution of a redundancy system with a general topology and heterogeneous server capacities. First of all, the number of copies in a particular server decreases, only if a certain subset of servers is already in steady state. Secondly, for a particular server $s \in S$, the instantaneous departure of that server might be larger than μ_s due to copies leaving in servers other than s. This makes the dynamics of the system complex. In order to prove Proposition 4.2.3, we therefore construct upper and lower bounds of our system for which the dynamics are easier to characterize. By using fluid limit arguments, we prove that the upper bound (lower bound) is stable (unstable) directly, which implies that the original system is also stable (unstable). This will be done in Proposition 4.2.12 and Proposition 4.2.19. See Section 2.4.2 for more details on the stability condition and the fluid-limit approximation. All proofs of this section can be found in Appendix 4.6.A.

We first introduce some notation: We denote by $E_c(t) = \max\{j \ : \ U_{cj} < t\}$ the number of type-c jobs that arrived during the time interval $(0,t)$ and by U_{cj} the instant of time at which the jth type-c job arrived to the system. We recall that b_{cj} denotes its service realization. Additionally, since copies are identical, all the copies of that job have the same service realization. We denote by b'_{cms} the residual job size of the mth eldest type-c job in server s that is already in service at time 0. We further let $\phi_s(\vec{M}^{PS}(t))$ be the capacity that each of the copies in server s obtains when in state $\vec{M}^{PS}(t)$, which under PS is given by $\phi_s(\vec{M}^{PS}(t)) := \frac{\mu_s}{M_s^{PS}(t)}$ for server $s \in S$ and time $t \geq 0$. The cumulative service that a copy in server s gets during the time interval (v,t) is then

$$\eta_s(v,t) := \int_{x=v}^{t} \phi_s(\vec{M}^{PS}(x))\mathrm{d}x.$$

4.2.3.1 Sufficient stability condition

Before we define the Upper Bound (UB) system we introduce some notation. We define the set of least-loaded servers for type-c jobs, for all $c \in \mathcal{C}$. That is,

$$\mathcal{R}(c) := \{s : \exists i, \text{ s.t. } c \in C_i(s) \text{ and } s \in \mathcal{L}_i\},$$

for all $c \in \mathcal{C}$. Note that there is a unique subsystem S_i for which this happens, i.e., $\mathcal{R}(c) \subseteq \mathcal{L}_i$ for exactly one i. We note that for a type-c job, if c contains at least a server that was removed in the ith iteration, then $\mathcal{R}(c) \subseteq \mathcal{L}_i$. We further let $\mathcal{R} := \cup_{c\in\mathcal{C}}\mathcal{R}(c)$.

We define the UB-system as follows. Upon arrival, each job is with probability p_c of type c and sends identical copies to all servers $s \in c$. In the UB-system, a type-c job departs the system **only when all copies in the set of servers $\mathcal{R}(c)$ are fully served.** We recall that the set $\mathcal{R}(c)$ denotes the set of servers where a type-c job is the least-loaded server. When this happens, the remaining copies that are still in service (necessarily not in a server in $\mathcal{R}(c)$) are immediately removed from the system. We denote by $N_c^{UB}(t)$ the number of type-c jobs present in the UB-system at time t.

We note that the UB-system is closely related to the one in which copies of type-c jobs are only sent to servers in $\mathcal{R}(c)$. However, the latter system is of no use for our purposes as it is neither an upper bound nor a lower bound of the original system.

We first prove that UB-system provides an upper bound on the original system. To do so, we show that every job departs earlier in the original system than in the UB-system. In the statement, we assume that in case a job has already departed in the original system, but not in the UB-system, then its attained service in all its servers in the original system is set equal to its service requirement b_{cj}.

Proposition 4.2.6. *Assume $N_c^{PS}(0) = N_c^{UB}(0)$ and $a_{cjs}^{PS}(0) = a_{cjs}^{UB}(0)$, for all c, j, s. Then, $N_c^{PS}(t) \leq N_c^{UB}(t)$ and $a_{cjs}^{PS}(t) \geq a_{cjs}^{UB}(t)$, for all c, j, s and $t \geq 0$.*

We now prove that $\lambda \sum_{s\in\mathcal{C}_l(s)} < \mu_s$, for all $l = 1, \ldots, i$, implies stability of the servers in the set $\mathcal{L}_i$, that is, the first implication of Proposition 4.2.3. We do this by analyzing the UB-system for which stability of the servers $\mathcal{L}_i$ follows intuitively as follows: Given a server $s \in \mathcal{L}_1$ and any type $c \in C(s)$, it holds that $\mathcal{R}(c) \subseteq \mathcal{L}_1(c)$. Hence, a server in $\mathcal{L}_1$ will need to fully serve all arriving copies. Therefore each server s, with $s \in \mathcal{L}_1$, behaves as an M/G/1 PS queue, which is stable if and only if its arrival rate of copies, $\lambda \sum_{c\in C_1(s)} p_c$, is strictly smaller than its departure rate, μ_s. Assume now that for all $l = 1, \ldots, i-1$ the subsystems S_l are stable and we want to show that servers in $\mathcal{L}_i$ are stable as well. First of all, note that in the fluid limit, all types c that do not exist in the S_i-subsystem, i.e., $c \notin C_i(s)$, will after a finite amount of time equal (and remain) zero, since they are stable. For the remaining types c that have copies in server $s \in \mathcal{L}_i$, i.e., $s \in c$ with $s \in \mathcal{L}_i$, it will hold that their least-loaded servers are $\mathcal{R}(c) \subseteq \mathcal{L}_i$. Due

to the characteristics of the upper bound system, all copies sent to these servers will need to be served. Hence, a server $s \in \mathcal{L}_i$ behaves in the fluid limit as an M/G/1 PS queue with arrival rate $\lambda \sum_{c \in C_i(s)} p_c$ and departure rate μ_s. In particular, such a queue is stable if and only if $\lambda \sum_{c \in C_i(s)} p_c < \mu_s$.

We now prove the stability of the UB-system. For that, we first describe the dynamics of the number of type-c jobs in the UB-system, denoted by $N_c^{UB}(t)$. We recall that a type-c job departs only when all the copies in the set of servers $\mathcal{R}(c)$ are completely served. We let $\eta^{min}_{\mathcal{R}(c)}(v,t) = \min_{\tilde{s} \in \mathcal{R}(c)}\{\eta_{\tilde{s}}(v,t)\}$ be the minimum cumulative amount of capacity received by a copy in one of its servers $\mathcal{R}(c)$ during the interval (v,t). Therefore,

$$\begin{aligned} N_c^{UB}(t) &= \sum_{m=1}^{N_c^{UB}(0)} 1\left(\{\exists \tilde{s} \in \mathcal{R}(c) \ : \ b'_{cm\tilde{s}} > \eta_{\tilde{s}}(0,t)\}\right) \\ &\quad + \sum_{j=1}^{E_c(t)} 1\left(b_{cj} > \eta^{min}_{\mathcal{R}(c)}(U_{cj},t)\right). \end{aligned}$$

We denote the number of type-c copies in server s by $M_{s,c}^{UB}(t)$. We note that for a type-c job in server s there are two possibilities: (server $s \in \mathcal{L}_i$),

- if $s \in \mathcal{R}(c)$, the copy of the type-c job leaves the server as soon as it is completely served. The cumulative amount of capacity that the copy receives during (v,t) is $\eta_s(v,t)$.
- If $s \notin \mathcal{R}(c)$, the copy of the type-c job in server s leaves the system either if it is completely served or if all copies of this type-c job in the servers $\mathcal{R}(c)$ are served. We note that for any $\tilde{s} \in \mathcal{R}(c)$, $\tilde{s} \in \mathcal{L}_l$, with $l < i$.

Hence, the number of type-c jobs in server $s \in \mathcal{L}_i$ is given by the following expression. If $s \in \mathcal{R}(c)$,

$$M_{s,c}^{UB}(t) = \sum_{m=1}^{M_{s,c}^{UB}(0)} 1\left(b'_{cms} > \eta_s(0,t)\right) + \sum_{j=1}^{E_c(t)} 1\left(b_{cj} > \eta_s(U_{cj},t)\right)$$

and if $s \notin \mathcal{R}(c)$,

$$\begin{aligned} M_{s,c}^{UB}(t) &= \sum_{m=1}^{M_{s,c}^{UB}(0)} 1\left(\{\exists \tilde{s} \in \mathcal{R}(c) \ : \ b'_{cm\tilde{s}} > \eta_{\tilde{s}}(0,t)\} \cap b'_{cms} > \eta_s(0,t)\right) \\ &\quad + \sum_{j=1}^{E_c(t)} 1\left(b_{cj} > \eta_{\mathcal{R}(c),s}(U_{cj},t)\right), \end{aligned}$$

where $\eta_{\mathcal{R}(c),s}(v,t) = \max\{\eta^{min}_{\mathcal{R}(c)}(v,t), \eta_s(v,t)\}$. The first terms in both equations cor-

respond to the type-c jobs that were already in the system by time $t = 0$, the second terms correspond to the type-c jobs that arrived during the time interval $(0, t)$.

In the following we obtain the number of copies per server. Before doing so, we need to introduce some additional notation. Let $\mathcal{D}^l(s) = \{c \in \mathcal{C}(s) \ : \ \mathcal{R}(c) \subseteq \mathcal{L}_l(c)\}$ be the set of types in server s for which the set of least-loaded servers for type-c jobs is $\mathcal{R}(c) \subseteq \mathcal{L}_l(c)$. If $s \in \mathcal{L}_i$, then, by definition, $\mathcal{D}^l(s) \neq \emptyset$ if $l \leq i$ and $\{\mathcal{D}^l(s)\}_{l=1}^{i}$ forms a partition of $\mathcal{C}(s)$. Furthermore, $D^i(s) = \mathcal{C}_i(s)$, for all $s \in \mathcal{L}_i$. Therefore, for a server $s \in \mathcal{L}_i$, the number of copies in the server is given by the following expression:

$$M_s^{UB}(s) = \sum_{c \in \mathcal{C}(s)} M_{s,c}^{UB}(t) = \sum_{l=1}^{i-1} \sum_{c \in \mathcal{D}^l(s)} M_{s,c}^{UB}(t) + \sum_{c \in \mathcal{C}_i(s)} M_{s,c}^{UB}(t).$$

The first term of the RHS of the equation corresponds to the type-c jobs in server s that have $\mathcal{R}(c) \subseteq \mathcal{L}_l(c)$. The second term of the RHS corresponds to type-c jobs in server s that have $\mathcal{R}(c) \subseteq \mathcal{L}_i(c)$. Particularly, we note that in the UB-system, $M_s^{UB}(t) \leq \sum_{c \in \mathcal{C}(s)} N_c^{UB}(t)$, since copies might have left, while the job is still present.

In order to prove the stability condition, we investigate the fluid-scaled system. The fluid-scaling consists in studying the rescaled sequence of systems indexed by parameter r, see Section 2.4.2.1 for more details. For $r > 0$, we denote by $M_{c,s}^{UB,r}(t)$ the system where the initial state satisfies $M_{s,c}^{UB}(0) = r m_{s,c}^{UB}(0)$, for all $c \in \mathcal{C}$ and $s \in S$. We define,

$$\bar{M}_{s,c}^{UB,r}(t) = \frac{M_{s,c}^{UB,r}(rt)}{r}, \quad \text{and} \quad \bar{M}_s^{UB,r}(t) = \frac{M_s^{UB,r}(rt)}{r}$$

In the following, we characterize the fluid model.

Definition 4.2.7. *Non-negative continuous functions $m_s^{UB}(\cdot)$ are a fluid model solution if they satisfy the functional equations*

$$\begin{aligned} m_s^{UB}(t) \ = \ & \sum_{l=1}^{i-1} \sum_{c \in \mathcal{D}^l(s)} \Big[m_{s,c}^{UB}(0) \left(1 - G\left(\bar{\eta}_{\mathcal{R}(c),s}(0,t)\right)\right) \\ & \qquad + \lambda p_c \left(\int_{x=0}^{t} 1 - F\left(\bar{\eta}_{\mathcal{R}(c),s}(x,t)\right) dx \right) \Big] \\ & + \sum_{c \in \mathcal{C}_i(s)} \Big[m_{s,c}^{UB}(0) \left(1 - G(\bar{\eta}_s(0,t))\right) \\ & \qquad + \lambda p_c \int_{x=0}^{t} (1 - F(\bar{\eta}_s(x,t))) dx \Big], \end{aligned} \tag{4.4}$$

for $s \in \mathcal{L}_i$ and $i = 1, \ldots, i^*$, where $G(\cdot)$ is the distribution of the remaining service requirements, $F(\cdot)$ the service time distribution of arriving jobs, and

$$\bar{\eta}_s(v,t) = \int_{x=v}^{t} \phi_s(\vec{m}^{UB}(x)) dx,$$

$$\bar{\eta}^{min}_{\mathcal{R}(c)}(v,t) = \min_{\tilde{s}\in\mathcal{R}(c)} \{\bar{\eta}_{\tilde{s}}(v,t)\},$$

$$\bar{\eta}_{\mathcal{R}(c),s}(v,t) = \max\{\bar{\eta}^{min}_{\mathcal{R}(c)}(v,t), \bar{\eta}_s(v,t)\}.$$

The existence and convergence of the fluid limit to the fluid model can now be proved.

Proposition 4.2.8. *The limit point of any convergent subsequence of* $(\bar{M}^{UB,r}_s(t), s \in S)_{t\geq 0}$ *is almost surely a solution of the fluid model Eq. (4.4).*

We now give a further characterization of the fluid model Eq. (4.4).

Proposition 4.2.9. *Let* $i \leq i^*$ *and assume* $\lambda \sum_{c\in\mathcal{C}_l(s)} p_c < \mu_s$ *for all* $l \leq i-1$ *and* $s \in \mathcal{L}_l$. *Then, there is a time* $T \geq 0$, *such that for* $t \geq T$ *and for* $s \in \cup_{l=1}^{i-1}\mathcal{L}_l$, $m^{UB}_s(t) = 0$ *and for* $s \in \mathcal{L}_i$

$$m^{UB}_s(t) = \sum_{c\in\mathcal{C}_i(s)} \left[m^{UB}_{s,c}(0)\,(1 - G(\bar{\eta}_s(0,t))) + \lambda p_c \int_{x=0}^{t} (1 - F(\bar{\eta}_{s,i}(x,t)))dx \right], \quad (4.5)$$

with

$$\bar{\eta}_{s,i}(v,t) := \int_{x=v}^{t} \phi_{s,i}(\vec{m}(x))dx,$$

and $\phi_{s,i}(\vec{m}(x)) := \frac{\mu_s}{\sum_{c\in\mathcal{C}_i(s)} m_{s,c}(x)}$.

Below we prove that the UB-system is Harris recurrent. Note that the concept of Harris recurrence is needed here since the state space is not countable, (as we need to keep track of residual service times). Hence, we need to prove that there exists a petite set C for which $P(\tau_C < \infty) = 1$ where τ_C is the stopping time of C, see Section 2.4 for the corresponding definitions. To do so, we first establish the fluid stability, that is, the fluid model is 0 in finite time. The latter is useful, as we can use the result in Theorem 2.4.13 (originally in [73]) that establishes that under some suitable conditions, fluid stability implies Harris recurrency, see the lemma below.

Lemma 4.2.10. *If the fluid limit is fluid stable, then the stochastic system is Harris recurrent.*

Eq. (4.5) coincides with the fluid limit of an $M/G/1$ PS system with arrival rate $\lambda \sum_{c\in\mathcal{C}_i(s)} p_c$ and server speed μ_s. If $\lambda \sum_{c\in\mathcal{C}_l(s)} p_c < \mu_s$, for all $l = 1,\ldots,i$, Eq. (4.5) equals zero in finite time. Hence, from Lemma 4.2.10 we conclude that for servers $s \in \mathcal{L}_i$, the associated stochastic number of copies in server s is Harris recurrent, as stated in the proposition below.

Proposition 4.2.11. *For* $i \leq i^*$, *the set of servers* $s \in \mathcal{L}_i$ *in the UB-system is stable if* $\lambda \sum_{c\in\mathcal{C}_i(s)} p_c < \mu_s$, *for all* $l = 1,\ldots,i$.

Together with Proposition 4.2.6, we obtain the following result for the original system.

Proposition 4.2.12. *For a given $i \leq i^*$, servers $s \in \mathcal{L}_i$ are stable if $\lambda \sum_{c \in \mathcal{C}_l(s)} p_c < \mu_s$, for all $l = 1, \ldots, i$.*

Remark 4.2.13. *In Section 3.4 we show that for the redundancy-d model, the system where all the copies need to be served is an upper bound. We note that this upper bound coincides with the upper bound presented in this chapter (in that case $\mathcal{L}_1 = S$). Nevertheless, the proof approach is different. In Section 3.4, see also [83], the proof followed directly, as each server in the upper bound system behaved as an M/G/1 PS queue. In the general redundancy topology studied here, the latter is no longer true for the system as a whole. Instead, it does apply recursively when considering the fluid regime: In order to see that a given server $s \in \mathcal{L}_i$ behaves as a PS queue in the fluid regime, one first needs to argue that all the servers with less load than that server, that is, the servers in $\mathcal{L}_1, \ldots, \mathcal{L}_{i-1}$, become 0 under fluid scaling.*

Remark 4.2.14. *In order to make a link with the result in Proposition 3.4.9, let us consider the fluid limit of Proposition 4.2.11 applied to the redundancy-d model where jobs have exponentially distributed service times (with unit mean). We recall that under redundancy-d, we have that $i^* = 1$ and thus, the UB-system is the system where all the copies need to be served. Then, the fluid limit expression in Proposition 4.2.11 simplifies to*

$$m_s^{UB}(t) = m_s^{UB}(0) + t(\lambda \frac{d}{K} - \mu).$$

We note above equation coincides with the fluid limit of an $M/M/1$ PS system, and is stable if and only if $\lambda < K\mu/d$. Which coincides with the stability condition of the UB-system defined in Section 3.4.2.2.

Remark 4.2.15. *We note that the service time distribution F has no atoms and is light-tailed (Eq. (4.1)). This assumption on the service time distribution, is an assumption needed in order to prove Lemma 4.2.10 (see Appendix 4.6.A for more details).*

4.2.3.2 Necessary stability condition

In this section we prove the necessary stability condition of Proposition 4.2.3. Let us first define

$$\iota := \min \left\{ l = 1, \ldots, i^* : \lambda \sum_{c \in \mathcal{C}_i(s)} p_c > \mu_s \right\},$$

and by $\gamma := \frac{\mu_s}{\sum_{c \in \mathcal{C}_\iota(s)} p_c}$.

We note that for any $i < \iota$, $\lambda \sum_{c \in \mathcal{C}_i(s)} p_c < \mu_s$. Hence, the servers in $\mathcal{L}_i$, with $i < \iota$ are stable, see Proposition 4.2.11. We are left to prove that the servers in S_ι cannot be stable. In order to do so, we construct a lower bound system.

In the S_ι subsystem, the loads are such that for all $s \in S_\iota$, $(\sum_{c \in \mathcal{C}_\iota(s)} p_c)/\mu_s \leq 1/\gamma$. We will construct the Lower Bound (LB) system in which the resulting load at each server is γ for all servers $s \in S_\iota$. We use the superscript LB in the notation to refer to this system, which is defined as follows. First of all, we only want to focus on the $\mathcal{S}_\iota$ system, hence, we set the arrival rate $p_c^{LB} = 0$ for types $c \in \mathcal{C} \backslash \mathcal{C}_\iota$, whereas the arrival rate for types $c \in \mathcal{C}_\iota$ remain unchanged, i.e., $p_c^{LB} = p_c$. The capacity of servers $s \in S_\iota$ in the LB-system is set to

$$\mu_s^{LB} := \mu_{\tilde{s}} \frac{\sum_{c \in \mathcal{C}_\iota(s)} p_c}{\sum_{c \in \mathcal{C}_\iota(\tilde{s})} p_c} = \gamma \cdot \Big(\sum_{c \in \mathcal{C}_\iota(s)} p_c \Big),$$

where $\tilde{s} \in \mathcal{L}_\iota$. Additionally, in the LB-system, we assume that each copy of a type-c job receives the same amount of capacity, which is equal to the highest value of $\mu_s^{LB}/M_s^{LB}(t)$, $s \in c$. We therefore define the service rate for a job of type c by

$$\phi_c^{LB}(\vec{N}^{LB}(t)) := \max_{s \in c} \left\{ \frac{\mu_s^{LB}}{M_s^{LB}(t)} \right\}, \tag{4.6}$$

where $c \in \mathcal{C}_\iota$ (instead of $\phi_s(\cdot)$ for a copy in server s in the original system). The cumulative amount of capacity that a type-c job receives is

$$\eta_c^{LB}(v,t) := \int_{x=v}^{t} \phi_c^{LB}(\vec{N}^{LB}(x))dx, \text{ for } c \in \mathcal{C}_\iota.$$

The number of type-c jobs in the system is given by

$$N_c^{LB}(t) = 0, \quad \text{for} \quad c \in \mathcal{C} \backslash \mathcal{C}_\iota,$$

$$N_c^{LB}(t) = \left[\sum_{m=1}^{N_c^{LB}(0)} 1\left(b'_{cms} > \eta_c^{LB}(0,t)\right) + \sum_{j=1}^{E_c(t)} 1\left(b_{cj} > \eta_c^{LB}(U_{cj},t)\right) \right], \quad \text{for} \quad c \in \mathcal{C}_\iota$$

We first prove that LB-system is a lower bound for the original system.

Proposition 4.2.16. *Assume $N_c^{PS}(0) = N_c^{LB}(0)$, for all c. Then, $N_c^{PS}(t) \geq_{st} N_c^{LB}(t)$, for all $c \in \mathcal{C}$ and $t \geq 0$.*

In order to show that the LB-system is unstable, we investigate the fluid-scaled system. For $r > 0$, denote by $N_c^{LB,r}(t)$ the system where the initial state satisfies

$N_c^{LB}(0) = rn_c^{LB}(0)$, for all $c \in \mathcal{C}$. We write for the fluid-scaled number of jobs per type

$$\bar{N}_c^{LB,r}(t) = \frac{N_c^{LB,r}(rt)}{r}.$$

In the following we give the characterization of the fluid model.

Definition 4.2.17. *Non-negative continuous functions $n_c^{LB}(\cdot)$ are a fluid model solution if they satisfy the functional equations*

$$n_c^{LB}(t) = 0, \quad c \in \mathcal{C}\backslash\mathcal{C}_\iota,$$

$$n_c^{LB}(t) = n_c^{LB}(0)\left(1 - G\left(\bar{\eta}_c^{LB}(0,t)\right)\right) + \lambda p_c \left(\int_{x=0}^{t} 1 - F\left(\bar{\eta}_c^{LB}(x,t)\right) dx\right), \quad c \in \mathcal{C}_\iota$$

where $G(\cdot)$ is the distribution of the remaining service requirements of initial jobs, $F(\cdot)$ the service time distribution of arriving jobs and

$$\bar{\eta}_c^{LB}(v,t) = \int_{x=v}^{t} \phi_c^{LB}(\vec{n}^{LB}(x))dx, \quad \textit{with} \quad c \in \mathcal{C}_\iota.$$

The existence and convergence of the fluid-scaled number of jobs $(\bar{N}_c^{LB,r}(t), c \in \mathcal{C})$ to the fluid model $(n_c^{LB}(t), c \in \mathcal{C})$ can be proved as before. The statement of Proposition 4.2.8, indeed directly translates to the process $(\bar{N}_c^{LB,r}(t), c \in \mathcal{C})$, since $\eta_c^{LB}(v,t)$ is both decreasing and continuous in v for all $c \in C$. Therefore, it is left out.

Next, we characterize the fluid model solution $\vec{n}^{LB(t)}$ in terms of the number of copies per server $m_s^{LB}(t) = \sum_{c \in \mathcal{C}(s)} n_c^{LB}(t)$. We show that if the initial condition for all servers is such that $m_s^{LB}(0)/\mu_s^{LB} = \alpha(0)$ for all $s \in S_\iota$, then $m_s^{LB}(t)/\mu_s^{LB} = \alpha(t)$ for all $s \in S_\iota$, where $\alpha(t)$ is given below.

Lemma 4.2.18. *Let us assume that the initial condition is such that $n_c^{LB}(0) = 0$ for all $c \in \mathcal{C}\backslash\mathcal{C}_\iota$ and for $c \in \mathcal{C}_\iota$, $n_c^{LB}(0)$ are such that $m_s^{LB}(0)/\mu_s^{LB} = \alpha(0)$ for all $s \in S_\iota$. Let*

$$\alpha(t) = \alpha(0)(1 - G(\bar{\eta}_\alpha^{LB}(0,t))) + \frac{\lambda}{\gamma}\int_{x=0}^{t}(1 - F(\bar{\eta}_\alpha^{LB}(x,t)))dx, \tag{4.7}$$

where $\bar{\eta}_\alpha^{LB}(v,t) = \int_{x=v}^{t} \phi_\alpha^{LB}(\alpha(x))dx$, with $\phi_\alpha^{LB}(\alpha(t)) = \frac{1}{\alpha(t)}$. Then, $n_c^{LB}(t) = 0$ for all $t \geq 0$ and $c \in \mathcal{C}\backslash\mathcal{C}_\iota$, and

$$m_s^{LB}(t)/\mu_s^{LB} = \alpha(t),$$

for all $t \geq 0$ and $s \in S_\iota$.

We note that Eq. (4.7) corresponds to the fluid limit of an $M/G/1$ system with PS, arrival rate λ/γ and server speed 1. Assuming $\lambda > \gamma$, it follows that the fluid limit $m_s^{LB}(t), s \in \mathcal{S}_\iota$ diverges. Hence, together with Proposition 4.2.16, the fluid limit of the original system $m_s(t)$ diverges for all $s \in \mathcal{S}_\iota$ as well, that is, the original system

is weakly unstable, see Definition 2.4.9. In the proposition below, we show that this implies that the stochastic process can not be stable either, by using similar arguments as in [33, Theorem 3.2].

Proposition 4.2.19. *For a given $i \leq i^*$, servers $s \in S_i$ are unstable if there is an $l = 1, \ldots, i$ such that $\lambda \sum_{s \in \mathcal{C}_l(s)} p_c > \mu_s$.*

Remark 4.2.20. *In the special case of the redundancy-d model, Section 3.4 used a lower bound that consisted in modifying the service rate obtained per job type, as in Eq. (4.6). This lower bound coincides with the lower bound presented in this chapter, since under the redundancy-d model, it holds that $\mu_s^{LB} = \mu_s = \mu$. The difficulty when studying a redundancy system with a general topology lies in the fact that the load received in each server is different. In order to show that the fluid limit of the server with the minimum number of copies is increasing (in the lower bound), we need to adequately modify the server capacities in order to make sure that the load at each of the servers is equal.*

Remark 4.2.21. *Let us focus on the fluid limit expression in Lemma 4.2.18. For the redundancy-d model with exponentially distributed service times (with unit mean), this lemma shows the diagonal property we explain in Section 3.4, where we showed that inside some cone around the diagonal (symmetric states), the fluid drift of the number of copies per server stay together for all the servers and is strictly positive. For this particular system, Lemma 4.2.18 shows that if the initial condition of the LB-system is such that $m_s^{LB}(0)/\mu = \alpha(0)$ for all $c \in \mathcal{C}$, then*

$$\alpha(t) = \alpha(0) + \lambda \frac{d}{K} - \mu, \tag{4.8}$$

and $m_s^{LB}(t) = \alpha(t)$, for all $t \geq 0$ and $s \in S$. The latter is closely related to Remark 3.4.6, where we show that if the system starts unbalanced, eventually the number of copies per server will get together and stay together for the rest of the time.

4.3 The FCFS and ROS scheduling policies

4.3.1 The FCFS scheduling policy

In this section, we consider redundancy systems with a general topology and heterogeneous server capacities where jobs have exponentially distributed service times and identical copies. The stability condition when FCFS is implemented in the servers is still an open problem. Under these assumptions we observe that the total instantaneous departure rate depends not only on the types in service, as it was the case for the redundancy-d model in Section 3.3, but also on the attained service of the corresponding copies in service as this determines from which server a copy can depart. Hence, the

state descriptor of the process needs to capture not only the job types in their order of arrival, but also the attained service times of each of the copies in service. The latter implies that the analysis of the stability condition is hard and that the techniques used in Section 3.3 are no longer valid.

In the following we discus the stability condition for a simple redundancy topology, the N-model. In the N-model, we note that type-$\{1,2\}$ jobs enter no later in server 1 than in server 2. Therefore, we can say that if $\mu_1 \geq \mu_2$, then type-$\{1,2\}$ jobs will get fully served in server 1, and hence intuitively, eventually jobs of type $\{2\}$ get the full capacity of server 2.

From the latter intuition, we can derive that the stability condition is given by the following conditions. However, we did not succeed in having a proof for the conjecture that the stability condition is as follows.

Conjecture 4.3.1. *Consider the N-model with $\mu_1 \geq \mu_2$, exponential service times and identical copies. This system is stable if $\lambda < \lambda^{R,FCFS}$ and unstable otherwise, where*

$$\lambda^{R,FCFS} = \begin{cases} \frac{\mu_1}{1-p}, & 0 \leq p \leq \frac{\mu_2}{\mu_1+\mu_2} \\ \frac{\mu_2}{p}, & \frac{\mu_2}{\mu_1+\mu_2} \leq p \leq 1. \end{cases}$$

When $p = \mu_2/(\mu_1+\mu_2)$, the above condition reduces to $\lambda < \mu_1 + \mu_2$, which is the maximum stability condition for exponentially distributed job sizes. We note that it coincides with the stability condition under PS when $\mu_1 \geq \mu_2$, derived in Section 4.2.2.2.

We note that for this simple model, we are able to guess the long-run departure rate of each type. However, no such guess will be possible when $\mu_1 < \mu_2$ or a general redundancy topology is considered. The stability condition of this redundancy system is further discussed in Chapter 8.

4.3.2 The ROS scheduling policy

The stability condition under the ROS scheduling policies when copies are identical is still an open problem. We provide our conjecture below, which we further discuss in Chapter 8.

Conjecture 4.3.2. *Consider the redundancy system with general topology and heterogeneous server capacities, where jobs have exponentially distributed service times and identical copies. The system under ROS is stable if for all $C \subseteq \mathcal{C}$,*

$$\lambda \sum_{c \in C} p_c < \sum_{s \in S(C)} \mu_s,$$

where $S(C) = \cup_{c \in C}\{s \in c\}$. *The system is unstable if there exists* $\tilde{C} \subseteq \mathcal{C}$ *such that*

$$\lambda \sum_{c \in \tilde{C}} p_c > \sum_{s \in S(\tilde{C})} \mu_s.$$

4.4 Numerical analysis

We have implemented a simulator in order to assess the impact of the scheduling policies FCFS, PS and ROS in redundancy models. Our simulations consider a large number of busy periods (10^6), so that the variance and confidence intervals of the mean number of jobs in the system are sufficiently small.

Within this section, we compare the performance of those service policies under different service time distributions. First, for exponentially distributed service times, we compare the performance with respect to the scheduling policy implemented in the servers. Second, for a given policy, we investigate how the service time distribution affects the performance of the system.

We further refer to Chapter 5 where the performance under redundancy scheduling is compared to load balancers that do not apply redundancy.

4.4.1 Exponential service time distributions

In order to analyze the impact of the scheduling policy on the performance of the system when jobs have identical copies, in Figure 4.3 we consider an N-model when either FCFS, or PS, or ROS is implemented. We assume capacities $\vec{\mu} = (1.25, 1)$ and

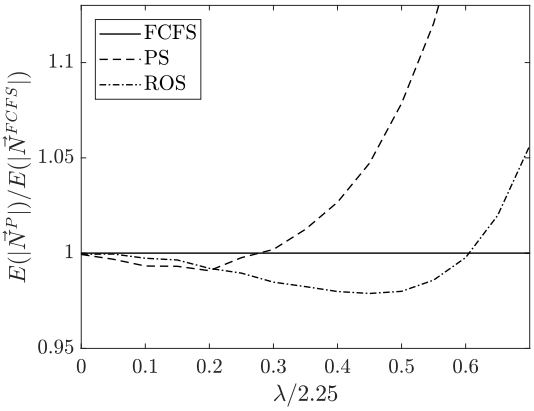
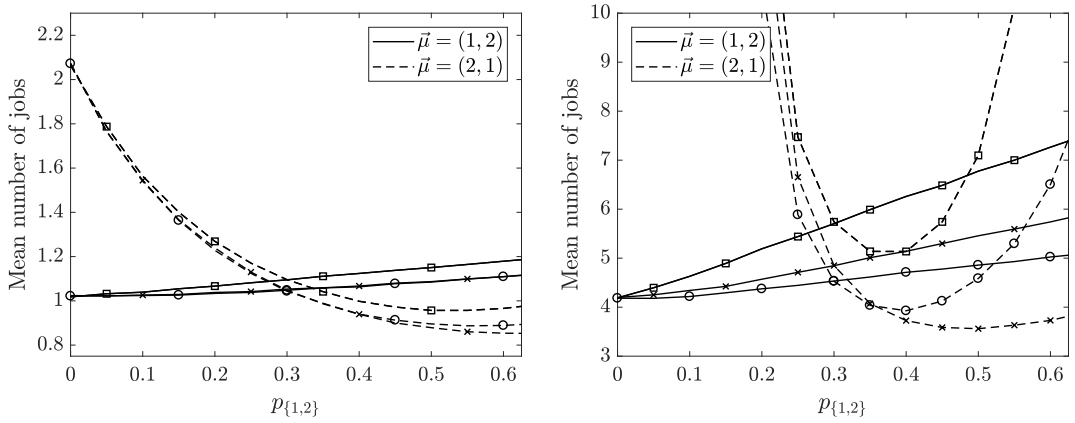


Figure 4.3 Mean number of jobs under FCFS, PS, and ROS with respect to that under FCFS. For the N-model with $\vec{\mu} = (1.25, 1)$, $\vec{p} = (1, 1.25)/2.25$, and with respect to $\lambda/2.25$.

$\vec{p} = (1, 1.25)/2.25$. We note that for this choice of μ and $\vec{p}$, the stability condition under PS is given by $\lambda < 2.25$, which is the maximum stability condition under exponentially distributed service times, see Section 4.2.2.2. Additionally, we conjecture that that is also the stability condition under both FCFS and ROS, see Section 4.3. In Figure 4.3, we plot the ratio between the mean number of jobs in the system under either FCFS, PS, or ROS, with respect to the mean number of jobs under FCFS. We observe that under low loads, PS and ROS outperform FCFS, whereas as the load increases FCFS outperforms both PS and ROS.

In Figure 4.4 (a) and (b) we consider a W-model and compare the performance under the different policies PS, FCFS and ROS. We take exponentially distributed service times with either $\vec{\mu} = (1, 2)$ or $\vec{\mu} = (2, 1)$. We plot the mean number of jobs with respect to $p_{\{1,2\}}$, with $p_{\{1\}} = 0.35$ and $p_{\{2\}} = 1 - p_{\{1\}} - p_{\{1,2\}}$. The only redundant job type is $\{1, 2\}$, thus as $p_{\{1,2\}}$ increases, we can observe how increasing the fraction of redundant jobs affects the performance. We also note that when $p_{\{1,2\}}$ increases, the load in server 1 increases as well, whereas the load in server 2 stays constant. In Figure 4.4 (a) we set $\lambda = 1$, and in Figure 4.4 (b) we set $\lambda = 2$.

In the case of $\vec{\mu} = (1, 2)$, we observe that FCFS always outperforms ROS. Intuitively we can explain this as follows. Since $p_{\{1\}}$ is kept fixed, as $p_{\{1,2\}}$ increases, the load in server 1 increases. With FCFS, it is more likely that both servers work on the same copy, and hence that the fast server 2 "helps" the slow server 1 (with high load). With ROS however, both servers tend to work on different copies, and the loaded slow server 1 will take a long time serving copies that could have been served faster in the fast server 2.

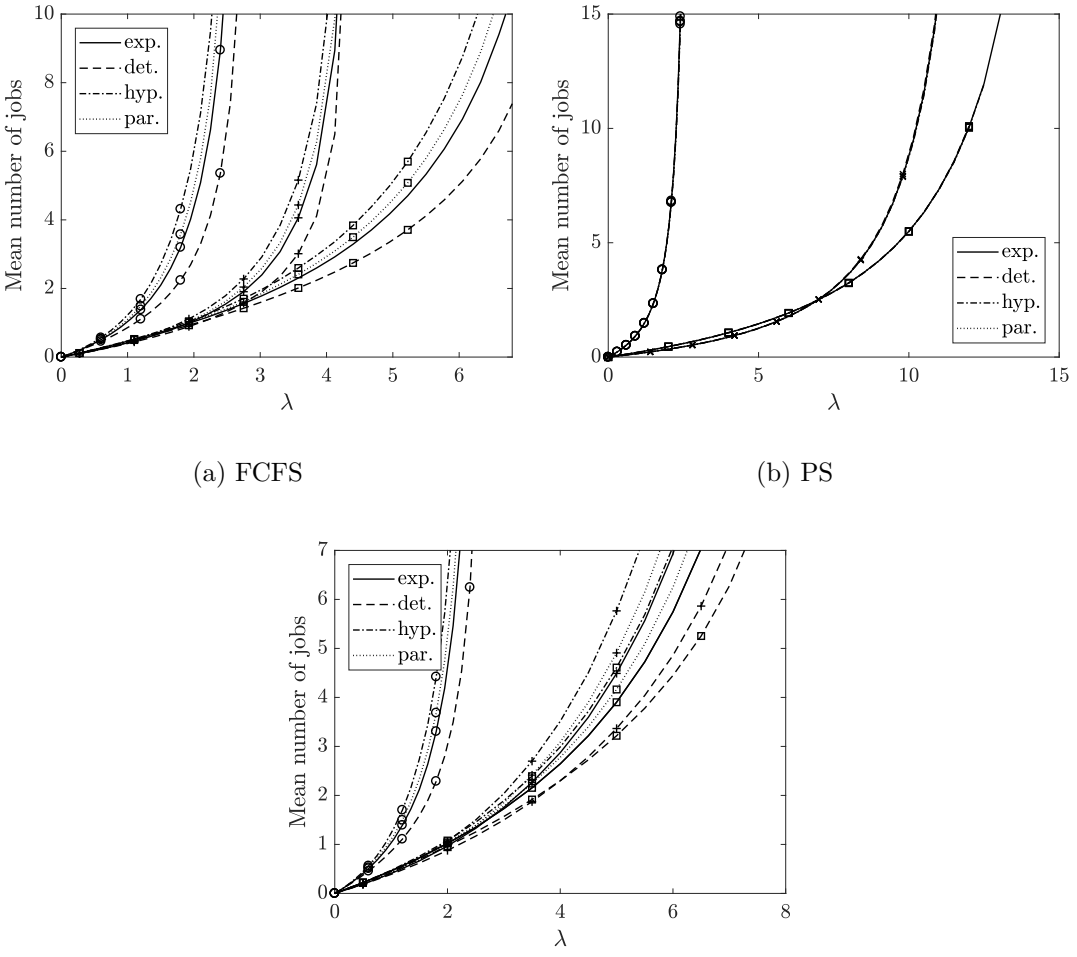


(a) $\lambda = 1$ (b) $\lambda = 2$

Figure 4.4 Mean number of jobs for the W-model where FCFS ($\circ$), PS ($\square$) and ROS ($\times$) is implemented in the servers, with respect to $p_{\{1,2\}}$ and exponentially distributed service times. We set $p_{\{1\}} = 0.35$ and $p_{\{2\}} = 1 - p_{\{1\}} - p_{\{1,2\}}$.

On the other hand, with $\vec{\mu} = (2, 1)$ and sufficiently large $p_{\{1,2\}}$, ROS outperforms FCFS. In this case, the loaded server 1 is the fast server, and hence having both servers working on the same copy becomes ineffecient, which explains that the performance under ROS becomes better. As a rule of thumb, it seems that for a redundancy model, if slow servers are highly loaded, then FCFS is preferable, but if fast servers are highly loaded, then ROS is preferable.

From Figures 4.4 (a) and (b) we further observe that for all values of $p_{\{1,2\}}$, FCFS and ROS outperform PS, and that the gap increases when λ increases.



(a) FCFS

(b) PS

(c) ROS

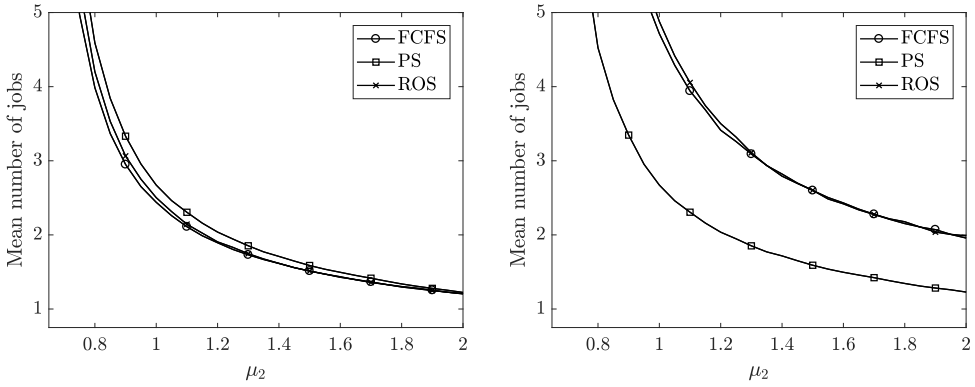
Figure 4.5 Mean number of jobs in the system with respect to λ for non-exponential service times and redundancy topologies W ($\circ$), and redundancy-2 ($\square$), and redundancy-4 ($\times$) with $K = 5$. For the W-model, we set $\vec{\mu} = (1, 2)$, and for the redundancy-d with $K = 5$ a) FCFS and c) ROS $\vec{\mu} = (1, 1.5, 2, 2.5, 3)$ and b) PS $\vec{\mu} = (1, 2, 4, 6, 8)$.

4.4.2 General service time distributions

In Figure 4.5 we investigate the performance of redundancy with (a) FCFS, (b) PS and (c) ROS for several non-exponential distributions. In particular, we consider the following distributions for the service times: deterministic, hyperexponential, and bounded Pareto, as well as the exponential distribution. With the hyperexponential distribution, job sizes are exponentially distributed with parameter ν_1 (ν_2) with probability q ($1-q$). For Pareto the density function is $\frac{1-(k/x)^\alpha}{(1-(k/\tilde{q})^\alpha)}$, for $k \leq x \leq \tilde{q}$. We choose the parameters so that the mean service time equals 1. Namely for the hyperexponential distribution parameters are $q = 0.2$, $\nu_1 = 0.4$ and $\nu_2 = 1.6$, and for the bounded Pareto distribution are $\alpha = 0.5$, $\tilde{q} = 6$ and $k = 1/\tilde{q}$. In Figure 4.5, we plot the mean number of jobs as a function of λ for the W-model ($\circ$) and the redundancy-2 topology with $K = 5$ ($\square$), and redundancy-4 topology with $K = 5$ ($\times$). For the W-model, $\vec{\mu} = (1, 2)$ and $p_c = 1/3$ for all $c \in \mathcal{C}$. For the redundancy-d topologies with $K = 5$, for FCFS and ROS $\vec{\mu} = (1, 1.5, 2, 2.5, 3)$, and for PS $\vec{\mu} = (1, 2, 4, 6, 8)$. For PS, the respective parameters $\vec{p}$ are chosen such that the system is stable for the simulated arrival rates.

We observe that when either FCFS or ROS is implemented, Figure 4.5 (a) and (c), the performance depends on the variability of the service time distribution, and it seems to degrade as the variability of the service time distribution increases.

We also observe that when $K = 5$ under FCFS and PS, the stability region under $d = 2$ is larger than that under $d = 4$. We also observe that when PS is implemented, Figure 4.5 (b), the performance seems to be nearly insensitive to the service time distribution, beyond its mean value. We note that similar conclusions were derived for



(a) Exponential service times (b) Bounded Pareto service times

Figure 4.6 For the W-model with $\vec{p} = (0.35, 0.40, 0.25)$ under (a) exponentially and (b) bounded Pareto, with parameters $\alpha = 0.5$, $\tilde{q} = 15$, distributed service times, and with respect to μ_2, for $\lambda = 1.5$ and $\mu_1 = 2$.

the redundancy-d model in Section 3.6.

In Figure 4.6 we consider (a) exponential and (b) bounded Pareto, with parameters $\alpha = 0.5$, $\tilde{q} = 15$, service time distributions and plot the mean number of jobs for different values of μ_2, when $\lambda = 1.5$, $\vec{p} = (0.35, 0.4, 0.25)$ and $\mu_1 = 2$. As before, with exponentially distributed service times, Figure 4.6 (a), FCFS and ROS slightly outperform PS. In the case where jobs have bounded Pareto distributed service times, Figure 4.6 (b), PS outperforms both FCFS and ROS. This seems to indicate that as the variability of the service time distribution increases, PS might become a preferable choice over FCFS and ROS in redundancy systems.

4.5 Concluding remarks

In this chapter, we characterize the stability condition of redundancy systems with a general topology when servers implement PS. In Chapter 5, we aim to understand under which circumstances redundancy improves the performance of a given system compared to a non-redundant system. We consider non-redundant Bernoulli routing, where jobs are dispatched uniformly at random to one of the compatible servers of the job, and JSQ.

We note that the stability condition when the scheduling policy is either FCFS or ROS remains an open problem. We provide further discussions, conjectures and open problems on the stability condition of several redundancy systems in Chapter 8.

4.6 Appendix

A: Proofs of Section 4.2

Proof of Corollary 4.2.5: The stability condition of such a system is given by Corollary 4.2.4. We note that each server $s \in S$ receives $\mathcal{C}(s) = \binom{K-1}{d-1}$ different job types, that is, by fixing a copy in server s, all possible combinations of $d-1$ servers out of $K-1$. Thus, $\mathcal{L}_1 = \arg\min_{s\in S_1}\{\lambda\frac{\binom{K-1}{d-1}}{\binom{K}{d}\mu_s}\} = K$, $S_2 = S - \{K\}$ and condition $\lambda\frac{\binom{K-1}{d-1}}{\binom{K}{d}} < \mu_K$.

We note each server $s \in S_i$ receives $\binom{|S_i|-1}{d-1}$ different job types, for $i = 1, \ldots, i^*$ and thus, the least-loaded server in the subsystem with servers S_i, only depends on the capacities of servers in S_i, that is $\mathcal{L}_i = \arg\min_{s\in S_i}\{1/\mu_s\}$. Additionally since, $\mu_1 < \ldots < \mu_K$, one obtains that $\mathcal{L}_i = K - i + 1$, for $i = 1, \ldots, K - d + 1$. The associated conditions are $\lambda\frac{\binom{K-i+1}{d-1}}{\binom{K}{d}} < \mu_{K-i+1}$ for $i = 1, \ldots, K - i + 1$. This set of conditions is equivalent to that in Corollary 4.2.5. □

Proof of Proposition 4.2.6: We assume that both systems are coupled as follows: at time $t = 0$, both systems start at the same initial state $N_c^{PS}(0) = N_c^{UB}(0)$ and $a_{cjs}^{PS}(0) = a_{cjs}^{UB}(0)$ for all c, j, s. Arrivals and service times are also coupled. For simplicity in notation, we assume that when in the original system a type-c copy reaches its service requirement b, the attained service of its $|c| - 1$ additional copies is fixed to b and the job remains in the system until the copy of that same job in the UB system is fully served at all servers in $\mathcal{R}(c)$.

We prove this result by induction on t. It holds at time $t = 0$. We assume that for $u \leq t$ it holds that $N_c^{PS}(t) \leq N_c^{UB}(t)$ and $a_{cjs}^{PS}(t) \geq a_{cjs}^{UB}(t)$ for all c, j, s. We show that this inequality holds for t^+.

We first assume that at time t, it holds that $N_c^{PS}(t) = N_c^{UB}(t)$ for some $c \in \mathcal{C}$. The inequality is violated only if there is a job for which the copy in the UB system is fully served at all servers $\mathcal{R}(c)$, but none of the copies in the original system is completed. That means, there exist a j such that $a_{cjs}^{PS}(t) < a_{cj\mathcal{R}(c)}^{UB}(t) = b_j$ for all $s \in c$. However, this can not happen, since by hypothesis $a_{cjs}^{PS}(t) \geq a_{cjs}^{UB}(t)$ for all $s \in c$.

We now assume that at time t, $a_{cjs}^{PS}(t) = a_{cjs}^{UB}(t)$ for some c, j, s. There are now two cases. If this copy (and job) has already left in the original system, then $a_{cjs}^{PS}(t) = a_{cjs}(t^+) = b_{cj}$ and hence $a_{cjs}^{PS}(t^+) \geq a_{cjs}^{UB}(t^+)$. If instead the copy has not left in the original system, then by hypothesis it holds that $N_c^{PS}(t) \leq N_c^{UB}(t)$ and thus, $M_s^{PS}(t) \leq M_s^{UB}(t)$ and $\frac{\mu_s}{M_s^{PS}(t)} \geq \frac{\mu_s}{M_s^{UB}(t)}$. That means that the copy in the original system has a higher service rate at time t than the same copy in the UB system. Hence, $a_{cjs}^{PS}(t^+) \leq a_{cjs}^{UB}(t^+)$. □

Proof of Proposition 4.2.8: The proof is identical to the the proof of Theorem 5.2.1 in [37] (which is itself based on Lemma 5 in [52]). We only need to ensure that $\bar{\eta}^{min}_{\mathcal{R}(c)}(v,t)$ and $\bar{\eta}_{\mathcal{R}(c),s}(v,t)$ are decreasing in v and continuous on $v \in [\psi_s(t)+\epsilon, t]$, where $\psi_s(t) = \sup(v \in [0,t] \ : \ m_s(u) = 0)$.

Let us verify that $\bar{\eta}^{min}_{\mathcal{R}(c)}(v,t)$ and $\bar{\eta}_{\mathcal{R}(c),s}(v,t)$ are decreasing and continuous on v. We note that the function $\eta_s(\cdot,t)$ that gives the cumulative service that a copy in server s received during time interval $(\cdot,t)$, is a Lipschitz continuous function, increasing for $t < \tau_s$ and non decreasing for $t > \tau_s$, where $\tau_s = \inf\{t > 0 \ : \ M_s(t) = 0\}$.

If $\bar{\eta}^{min}_{\mathcal{R}(c)}(v,t) = \bar{\eta}_{s_1}(v,t)$ and $\bar{\eta}_{\mathcal{R}(c),s}(v,t) = \bar{\eta}_{s_2}(v,t)$ for all $v \in [0,t)$ and some $s_1, s_2 \in S$, then both $\bar{\eta}^{min}_{\mathcal{R}(c)}(v,t)$ and $\bar{\eta}_{\mathcal{R}(c),s}(v,t)$ are decreasing and continuous on v, since by definition $\bar{\eta}_s(v,t)$ is decreasing and continuous on v for all $s \in S$.

Let us assume that for $v_0 \in [0,t)$ is such that $\bar{\eta}^{min}_{\mathcal{R}(c)}(v,t) = \bar{\eta}_{\tilde{s}^1}(v,t)$ for $v \leq v_0$ and $\bar{\eta}^{min}_{\mathcal{R}(c)}(v_0^+,t) = \bar{\eta}_{\tilde{s}^2}(v_0,t)$, for some $\tilde{s}^1, \tilde{s}^2 \in \mathcal{R}(c)$. We first verify that $\bar{\eta}^{min}_{\mathcal{R}(c)}(v,t)$ is continuous on $v = v_0$. Since, $\bar{\eta}_{\tilde{s}^1}(v,t)$ and $\bar{\eta}_{\tilde{s}^2}(v,t)$ are continuous on $v = v_0$, then

$$\lim_{x^- \to v_0} \bar{\eta}^{min}_{\mathcal{R}(c)}(x,t) = \bar{\eta}_{\tilde{s}^1}(v,t) = \bar{\eta}_{\tilde{s}^2}(v,t) = \lim_{x^+ \to v_0} \bar{\eta}^{min}_{\mathcal{R}(c)}(x,t).$$

Therefore, we conclude that $\bar{\eta}^{min}_{\mathcal{R}(c)}(x,t)$ is continuous on $v \in [0,t)$. Analogously, one can verify that $\bar{\eta}^{min}_{\mathcal{R}(c),s}(x,t)$ is continuous on $v \in [0,t)$.

We now verify that $\bar{\eta}^{min}_{\mathcal{R}(c)}(x,t)$ is decreasing on $v \in [0,t)$. Let us consider $0 < t_1 < v_0 < t_2 < t$. Then for $\bar{\eta}^{min}_{\mathcal{R}(c)}(v,t)$,

$$\bar{\eta}^{min}_{\mathcal{R}(c)}(t_1,t) = \bar{\eta}_{\tilde{s}^1}(t_1,t) \leq \bar{\eta}_{\tilde{s}^1}(t_2,t) \leq \bar{\eta}_{\tilde{s}^2}(t_2,t) = \bar{\eta}^{min}_{\mathcal{R}(c)}(t_2,t),$$

where the first inequality holds since $\bar{\eta}_{\tilde{s}^1}(v,t)$ is decreasing on v. We conclude that $\bar{\eta}^{min}_{\mathcal{R}(c)}(v,t)$ is decreasing v.

Let us verify that $\bar{\eta}_{\mathcal{R}(c),s}(v,t)$ is decreasing on v. W.l.o.g. we assume that there exists $v_0 \in [0,t)$, such that $\bar{\eta}_{\mathcal{R}(c),s}(v,t) = \bar{\eta}^{min}_{\mathcal{R}(c)}(v,t)$ for $v < v_0$ and $\bar{\eta}_{\mathcal{R}(c),s}(v,t) = \bar{\eta}_s(v,t)$ for $t > v > v_0$. Then,

$$\bar{\eta}_{\mathcal{R}(c),s}(t_1,t) = \bar{\eta}^{min}_{\mathcal{R}(c)}(t_1,t) \leq \bar{\eta}_s(t_1,t) \leq \bar{\eta}_s(t_2,t) = \bar{\eta}_{\mathcal{R}(c),s}(t_2,t)$$

where the first inequality holds since $\bar{\eta}_{\tilde{s}^1}(v,t)$ is decreasing on v. We conclude that $\bar{\eta}_{\mathcal{R}(c),s}(x,t)$ is decreasing v. □

Proof of Proposition 4.2.9: For simplicity in notation, we remove the superscript UB throughout the proof.

First assume $s \in \mathcal{L}_1$. Since $\mathcal{D}^0 = \emptyset$, from Eq. (4.4), we directly obtain

$$m_s(t) = \sum_{c \in \mathcal{C}_1(s)} [m_{s,c}(0)\,(1 - G(\bar{\eta}_s(0,t))) + \lambda p_c \int_{x=0}^{t} (1 - F(\bar{\eta}_s(x,t)))dx], \forall t > 0.$$

This expression coincides with the fluid limit of an $M/G/1$ PS queue with arrival rate $\lambda \sum_{c \in \mathcal{C}_1(s)} p_c$ and server speed μ_s. Since $\lambda \sum_{c \in \mathcal{C}_1(s)} p_c < \mu_s$, we know that there exists a $\bar{\tau}_s$ such that $m_s(t) = 0$, for all $t \geq \bar{\tau}_s$.

The remainder of the proof is by induction. Consider now a server $s \in \mathcal{L}_l$ and assume there exists a time $\tilde{T}$ such that $m_s(t) = 0$, for all $t \geq \tilde{T}$ and $s \in \cup_{j=1}^{l-1} \mathcal{L}_j$. Thus, for $t \geq \tilde{T}$, also $m_{s,c}(t) = 0$ for all $s \in \cup_{j=1}^{l-1} \mathcal{L}_j$, $c \in \mathcal{D}^j(s)$, $j = 1, \ldots, l-1$. We consider server $s \in \mathcal{L}_l$. From Eq. (4.4) its drift is then given by:

$$\begin{aligned} m_s(t) &= \sum_{j=1}^{l-1} \sum_{c \in \mathcal{D}^j(s)} m_{s,c}(t) + \sum_{c \in \mathcal{C}_l(s)} m_{s,c}(t) \\ &= \sum_{c \in \mathcal{C}_l(s)} \left[m_{s,c}(0)\,(1 - G(\bar{\eta}_s(0,t))) + \lambda p_c \int_{x=0}^{t} (1 - F(\bar{\eta}_s(x,t)))dx \right], \end{aligned}$$

for all $t \geq \tilde{T}$. Now note that $\phi_s(\vec{m}(t)) = \frac{\mu_s}{m_s(t)} = \frac{\mu_s}{\sum_{c \in \mathcal{C}_l(s)} m_{s,c}(t)} = \phi_{s,l}(\vec{m}(t))$, where the second equality follows from the fact that $m_{s,c}(t) = 0$ for all for all $s \in \cup_{j=1}^{l-1} \mathcal{L}_j$, $c \in \mathcal{D}^j(s)$, $j = 1, \ldots, l-1$.

To finish the proof, Eq. (4.5) coincides with the fluid limit of an $M/G/1$ system with PS, arrival rate $\lambda \sum_{c \in \mathcal{C}_l(s)} p_c$ and server speed μ_s. Hence, if $l < i$, the standard PS queue is stable, and we are sure that it equals and remains zero in finite time. □

Proof of Lemma 4.2.10: In [73], the authors consider bandwidth sharing networks (with processor sharing policies), and show that under mild conditions, the stability of the fluid model (describing the Markov process of the number of per-class customers with their residual job sizes) is sufficient for stability (positive Harris recurrence).

Our system, though slightly different from theirs satisfies the same assumptions, and as a consequence their results are directly applicable to our model.

More precisely, given the assumptions on the service time distribution, our model satisfies the assumptions given in [73, Section 2.2] for inter-arrival times and job-sizes. (In particular exponential inter-arrival times satisfy the conditions given in [73, Assumption 2.2.2].) □

Proof of Lemma 4.2.18: From Definition 4.2.17, we obtain that for each server

$s \in S_\iota$,

$$\begin{aligned}\frac{m_s^{LB}(t)}{\mu_s^{LB}} &= \frac{1}{\mu_s^{LB}} \sum_{c \in \mathcal{C}_\iota(s)} n_c^{LB}(t) \\ &= \sum_{c \in \mathcal{C}_\iota(s)} \left[\frac{n_c^{LB}(0)}{\mu_s^{LB}} \left(1 - G\left(\bar{\eta}_c^{LB}(0,t)\right)\right) \right. \\ &\quad \left. + \frac{\lambda p_c}{\mu_s^{LB}} \left(\int_{x=0}^{t} 1 - F\left(\bar{\eta}_c^{LB}(x,t)\right) dx \right) \right].\end{aligned}$$

We recall that $\alpha(t)$ is defined as

$$\alpha(t) = \alpha(0)\left(1 - G\left(\bar{\eta}_\alpha^{LB}(0,t)\right)\right) + \frac{\lambda}{\gamma}\left(\int_{x=0}^{t} 1 - F\left(\bar{\eta}_\alpha^{LB}(x,t)\right) dx \right).$$

We let the initial condition be such that $\frac{m_s^{LB}(0)}{\mu_s^{LB}} = \alpha(0)$ for all $s \in S_\iota$ and we will prove by contradiction that for all $t > 0$,

$$\frac{m_s^{LB}(t)}{\mu_s^{LB}} = \alpha(t), \quad \text{for all } s \in S_\iota.$$

Let us assume that t_0 is the first time such that there exists $\tilde{s} \in S_\iota$ such that $\alpha(t_0) \neq m_{\tilde{s}}^{LB}(t_0)/\mu_{\tilde{s}}^{LB}$. Since $\sum_{c \in \mathcal{C}_\iota(s)} \frac{n_c^{LB}(0)}{\mu_s^{LB}} = \frac{m_s^{LB}(0)}{\mu_s^{LB}} = \alpha(0)$ and $\sum_{c \in \mathcal{C}_\iota(s)} \frac{p_c}{\mu_s^{LB}} = 1/\gamma$, this implies that there exist $\tilde{c} \in \tilde{\mathcal{C}}$ and t_1, $0 \leq v \leq t_1 < t_0$ such that $\bar{\eta}_\alpha^{LB}(v,t_1) \neq \bar{\eta}_{\tilde{c}}^{LB}(v,t_1)$. However, since $\alpha(t) = m_s^{LB}(t)/\mu_s^{LB}$ for all $t < t_0$, this implies that $\phi_c(\vec{n}(t)) = 1/\alpha(t)$ for all $t < t_0$ and $c \in \tilde{\mathcal{C}}(s)$, and hence $\bar{\eta}_\alpha^{LB}(v,t) = \bar{\eta}_{\tilde{c}}^{LB}(v,t)$, for all $t < t_0$. We have hence reached a contradiction, which concludes the proof. □

Proof of Proposition 4.2.16: The number of type-c jobs in the system is given by $N_c^{LB}(t) = 0$, for $c \in \mathcal{C}\backslash\mathcal{C}_\iota$, and for $c \in \mathcal{C}_\iota$,

$$N_c^{LB}(t) = \left[\sum_{m=1}^{N_c^{LB}(0)} 1\left(b'_{cms} > \eta_c^{LB}(0,t)\right) + \sum_{j=1}^{E_c(t)} 1\left(b_{cj} > \eta_c^{LB}(U_{cj},t)\right) \right].$$

We note that for all $c \in \mathcal{C}\backslash\mathcal{C}_\iota$ the result is direct since $p_c^{LB} = 0$ for all $c \in \mathcal{C}\backslash\mathcal{C}_\iota$. Then, let us consider $c \in \mathcal{C}_\iota$. For any $\vec{N}^{PS}$ and $\vec{N}^{LB}$ such that $\vec{N}^{PS} \geq \vec{N}^{LB}$, the following inequalities hold:

$$\begin{aligned}\phi_s(\vec{N}) &= \frac{\mu_s}{M_s^{PS}} = \frac{\mu_s(\sum_{c \in \mathcal{C}_\iota(s)} p_c)}{(\sum_{c \in \mathcal{C}_\iota(s)} p_c) M_s^{PS}} = \frac{(\sum_{c \in \mathcal{C}_\iota(s)} p_c)\mu_s/(\sum_{c \in \mathcal{C}_\iota(s)} p_c)}{\sum_{c \in \mathcal{C}(s)\backslash\mathcal{C}_\iota(s)} N_c^{PS} + \sum_{c \in \mathcal{C}_\iota(s)} N_c^{PS}} \\ &\leq \frac{(\sum_{c \in \mathcal{C}_\iota(s)} p_c)\mu_s/(\sum_{c \in \mathcal{C}_\iota(s)} p_c)}{\sum_{c \in \mathcal{C}_\iota(s)} N_c^{PS}} \leq \frac{\gamma \times (\sum_{c \in \mathcal{C}_\iota(s)} p_c)}{\sum_{c \in \mathcal{C}_\iota(s)} N_c^{LB}} \leq \max_{s \in c}\left\{ \frac{\mu_s^{LB}}{M_s^{LB}} \right\} \\ &= \phi_c^{LB}(\vec{N}^{LB}).\end{aligned}$$

The second last inequality holds since $\gamma \geq \mu_s/(\sum_{c\in\mathcal{C}_\iota(s)} p_c)$ for all $s \in S_\iota$ and $N_c^{LB} \leq N_c^{PS}$ for all $c \in \mathcal{C}_\iota$. We note that $\sum_{c\in\mathcal{C}_\iota(s)} N_c^{LB} = M_s^{LB}(t)$. It follows from straight forward sample-path arguments that $N_c^{LB}(t) \leq N_c^{PS}(t)$ for all $t \geq 0$ and $c \in \mathcal{C}_\iota$. □

Proof of Proposition 4.2.19: We prove this proposition by using similar arguments as in Theorem 3.2 in [33], but to keep the manuscript self-contained, we adapt the proof to our case.

We show that the stochastic system is unstable, that is, $|\vec{M}(t)| \to \infty$ as $t \to \infty$, with probability 1. Because the fluid limit of the original system is weakly unstable, there exists a time $T > 0$ such that $|\vec{m}(T)| > 0$ for any fluid limit $\vec{m}(t)$ of $\vec{M}(t)$. In the following, we show that $\liminf_{r\to\infty} |\vec{M}(rT)/r| > 0$ with probability 1 by contradiction, which directly implies that $\lim_{t\to\infty} |\vec{M}(t)| = \infty$.

Assume that $\liminf_{r\to\infty} |\vec{M}(rT)/r| = 0$. The latter implies that there exists a subsequence r_n of r such that $|\vec{M}(r_nT)/r_n| \to 0$ as $n \to \infty$. We note that $\{M(r\cdot)/r, r \geq 1\}$ is precompact, thus there exists a subsequence r_{n_m} of r_n such that $\vec{M}(r_{n_m}T)/r_{n_m}$ converges u.o.c. to a fluid limit $\vec{m}(\cdot)$. The latter implies that

$$|\vec{M}(r_{n_m}T)/r_{n_m}| \to |\vec{m}(T)| > 0.$$

Hence, $\liminf_{r\to\infty} |\vec{M}(rT)/r| > 0$, which concludes the proof. □

Chapter 5

When does redundancy improve performance

In the present chapter we investigate the impact that redundancy scheduling has on the performance of the system, by comparing redundancy scheduling to Bernoulli routing and Join-the-Shortest-Queue (JSQ). Under Bernoulli routing, a type-c job is send with uniform probability to one of its compatible servers in c. Under JSQ, each job is dispatched to the compatible server that has the least number of jobs (ties are broken at random). Our analysis consists of two steps: in the first step, we compare analytically the stability condition of the system under redundancy scheduling to that under Bernoulli routing. Secondly, we numerically compare the mean number of jobs in the system under redundancy to that under Bernoulli routing and JSQ.

From a dispatchers viewpoint, the analytical comparison between redundancy and Bernoulli routing is reasonable under the assumption that the dispatcher only knows the type of the job and the set of its compatible servers. Numerically, we also compare to JSQ, which is the optimal scheduling policy under certain conditions ([74]). However, we note that opposite to redundancy, JSQ uses full information on the state of the system.

We aim to understand when having redundant copies is beneficial for the performance of the system in this context. Observe that the answer is not clear upfront as adding redundant copies has two opposite effects: on the one hand, redundancy helps exploiting the variability across server capacities, but on the other hand, it induces a waste of resources as servers work on copies that do not end up being completely served.

In order to make this analysis concrete, we assume that jobs have exponentially distributed service times and identical copies. We consider various server capacity settings where PS is implemented. We compute the stability condition for this system and compare it to that under Bernoulli routing.

We observe that when the capacities of the servers are sufficiently heterogeneous, the stability region under redundancy is larger than that under Bernoulli routing. In addition, numerical computations allow us to conclude that the degree of heterogeneity needed in the servers in order for redundancy to be beneficial, decreases in the number of servers, and increases in the number of redundant copies. By simulation, we observe that these conclusions are also appreciable in the mean number of jobs in the system.

Using a simulator, we compare numerically the mean number of jobs under redundancy to those under Bernoulli routing and JSQ routing, where either FCFS, PS or ROS is implemented in the servers. We observe that redundancy might not always improve the performance and when it does improve, the improvement is small. However, we note that JSQ requires full information of the state of the queues at every instant of time, which is unpractical.

The present chapter is organized as follows: In Section 5.1 we derive the stability condition under Bernoulli routing and in Section 5.2 compare it to the stability condition under redundancy. In Section 5.3, we numerically compare the mean number of jobs under redundancy to that under both Bernoulli routing and JSQ.

5.1 Stability under Bernoulli routing

We define by λ^B the value of λ such that the Bernoulli routing model is stable if $\lambda < \lambda^B$ and unstable if $\lambda > \lambda^B$. Under Bernoulli routing, the redundancy topology determines the set of compatible servers of each job type where the dispatcher can send the job. Upon arrival the job is dispatched to one of its compatible servers chosen uniformly at random. There is an arrival of a type-c job to server s at rate $\lambda p_c/|c|$. Thus, the Bernoulli system reduces to K independent servers, where server s receives arrivals at rate $\lambda(\sum_{c\in\mathcal{C}(s)} \frac{p_c}{|c|})$ and has a departure rate μ_s, for all $s \in S$. This gives the following stability result.

Proposition 5.1.1. *The Bernoulli routing system, where jobs are uniformly distributed across its compatible servers and any work-conserving scheduling policy is implemented in the servers, is stable if and only if,*

$$\lambda < \lambda^B := \min_{s\in S} \left\{ \frac{\mu_s}{\sum_{c\in\mathcal{C}(s)} \frac{p_c}{|c|}} \right\}. \tag{5.1}$$

5.2 Comparison of the stability condition

In the present section we compare the stability region under redundancy scheduling, obtained in Chapters 3 and 4, to that under Bernoulli routing, Proposition 5.1.1. Recall

that $\lambda^{R,P}$ denotes the value of λ such that the redundancy scheduling under scheduling policy P is stable if $\lambda < \lambda^{R,P}$ and unstable if $\lambda > \lambda^{R,P}$, for $P \in \{\text{FCFS, PS, ROS}\}$.

For the PS scheduling policy we have that $\lambda^{R,PS} = \min_{i=1,\dots,i^*, s\in\mathcal{L}_i} \left\{ \frac{\mu_s}{\sum_{c\in\mathcal{C}_i(s)} p_c} \right\}$, from Corollary 4.2.4. Together with Eq. (5.1), we obtain the following sufficient and necessary conditions for redundancy to improve the stability condition.

Corollary 5.2.1. *Consider the heterogeneous PS server system with a general redundancy topology and with exponentially distributed service times. The stability region under redundancy with identical copies is larger than under Bernoulli routing if and only if*

$$\min_{i=1,\dots,i^*, s\in\mathcal{L}_i} \{ \frac{\mu_s}{\sum_{c\in\mathcal{C}_i(s)} p_c} \} > \min_{s\in S} \{ \frac{\mu_s}{\sum_{c\in\mathcal{C}(s)} \frac{p_c}{|c|}} \}.$$

The stability condition when either FCFS or ROS is implemented is only known for the redundancy-d model. For this setting, the following corollary is straightforward after Proposition 3.3.3 and Proposition 3.5.3.

Corollary 5.2.2. *Let us consider the redundancy-d model, jobs have exponential service times and identical copies.*

- *The stability condition when FCFS is implemented is larger of equal than that under Bernoulli routing if and only if*

$$\bar{\ell}\mu > \min_{s\in S} \{ \frac{\mu_s}{\sum_{c\in\mathcal{C}(s)} \frac{p_c}{|c|}} \}.$$

- *The stability condition when ROS is implemented is equal to that under Bernoulli routing, that is given by, $\lambda < K\mu$.*

We the reminder of this section, we consider the redundancy-d and nested redundancy topologies and compute the stability region for various server capacities settings. We aim to understand how the stability region behaves under redundancy when the heterogeneity among the servers increases.

5.2.1 Redundancy-d topology

We consider the redundancy system with redundancy-d topology and heterogeneous server capacities. When jobs are dispatched according to Bernoulli routing, the stability condition reduces to the following.

Corollary 5.2.3. *Assume the redundancy system with redundancy-d topology and K heterogeneous servers where $\mu_1 < \ldots < \mu_K$. The stability condition under Bernoulli routing is given by $\lambda < K\mu_1$.*

Table 5.1 The maximum arrival rates $\lambda^{R,PS}$ and λ^{B} in the system under PS and capacities $\mu_k = \mu^{k-1}$ with the redundancy-d topology.

		$\mu = 1$	$\mu = 1.2$	$\mu = 1.4$	$\mu = 2$	$\mu = 3$	μ^*
Red-2	$K = 3$	1.5	2.16	2.94	6	9	1.41
	$K = 4$	2	3.45	5.48	12	18	1.26
	$K = 5$	2.5	5.18	9.14	20	30	1.19
	$K = 10$	5	22.39	41.16	90	135	1.08
Red-3	$K = 4$	1.33	2.30	3.65	10.66	**36**	1.44
	$K = 5$	1.66	3.45	6.40	**26.66**	**90**	1.31
	$K = 10$	3.33	17.19	**60.23**	**320**	**1080**	1.13
BR	$K = 3$	3	3	3	3	3	
	$K = 4$	4	4	4	4	4	
	$K = 5$	5	5	5	5	5	
	$K = 10$	10	10	10	10	10	

When servers implement PS, we obtain that $\lambda^{R,PS} = \min_{i=d,\ldots,K} \left\{ \frac{\binom{K}{d}}{\binom{i-1}{d-1}} \mu_i \right\}$, from Corollary 4.2.5. The following corollary is straightforward.

Corollary 5.2.4. *Assume the redundancy system with redundancy-d topology and K heterogeneous servers where $\mu_1 < \ldots < \mu_K$. When PS is implemented in the servers, the stability condition under redundancy is strictly larger than under Bernoulli routing if and only if*

$$K\mu_1 < \min_{i=d,\ldots,K} \left\{ \frac{\binom{K}{d}}{\binom{i-1}{d-1}} \mu_i \right\}.$$

Because $\binom{i-1}{d-1}$ is increasing in i, the following corollary is straightforward.

Corollary 5.2.5. *Assume the K heterogeneous server system with the redundancy-d topology and where $\mu_1 < \ldots < \mu_K$. When PS is implemented in the servers, the system under redundancy has a larger stability region than the Bernoulli routing if $\mu_1 d < \mu_d$.*

Hence, if there exists a redundancy parameter d such that $\mu_1 d < \mu_d$, then adding d redundant copies to the system improves its stability region. In that case, the stability condition of the system will improve by at least a factor $\frac{\mu_d}{d\mu_1}$.

In the following, we consider two server capacities settings and compare the stability region under redundancy to that under Bernoulli routing when servers implement PS.

Server setting 1: In Table 5.1, we chose $\mu_k = \mu^{k-1}$, $k = 1, \ldots, K$, so that the minimum capacity equals 1. Hence, for Bernoulli, $\lambda^B = K$. We assume that the servers implement PS and compare the stability condition under redundancy scheduling to that under Bernoulli routing.

Under redundancy we have the following: For $\mu = 1$ the system is a redundancy-d model (homogeneous capacities and redundancy-d topology), so that $\lambda^{R,PS} = K/d$,

see Section 3.4. We denote by μ^* the value of μ for which the stability region of the redundant system coincides with that of Bernoulli routing, i.e., the value of μ such that $\lambda^{R,PS} = \lambda^B$. For $\mu < \mu^*$ (the gray area on the left-hand-side of the thick line in Table 5.1), Bernoulli has a larger stability region, while for $\mu > \mu^*$ (the area on the right-hand-side of the thick line in in Table 5.1), redundancy outperforms Bernoulli.

We observe that for a fixed d, μ^* decreases as K increases as well. Therefore, as the number of servers increases, the level of heterogeneity that is needed in the servers in order to improve the stability under redundancy decreases.

We also observe that for fixed K, μ^* increases as d increases for PS. This means that as the number of redundant copies d increases, the server capacities need to be more heterogeneous in order to improve the stability region under redundancy.

Finally, let us focus on the numbers in bold, we observe that when the number of servers K is large enough and the servers are heterogeneous enough (large μ), the stability region increases in the number of redundant copies d.

Server setting 2: In Table 5.2, we consider linearly increasing capacities on the interval $[1, M]$, that is $\mu_k = 1 + \frac{M-1}{K-1}(k-1)$, for $k = 1, \ldots, K$. In the area on the right-hand-side of the thick line, redundancy outperforms Bernoulli.

Table 5.2 The maximum arrival rates $\lambda^{R,PS}$ and λ^B in the system under PS and capacities $\mu_k = 1 + \frac{M-1}{K-1}(k-1)$ with the redundancy-d topology.

		$M=1$	$M=2$	$M=3$	$M=4$	$M=6$
Red-2	$K=3$	1.5	3	4.5	6	9
	$K=4$	2	4	6	8	12
	$K=5$	2.5	5	7.5	10	15
	$K=10$	5	10	15	20	30
Red-3	$K=4$	1.33	2.66	4	5.33	8
	$K=5$	1.66	3.33	5	6.66	10
	$K=10$	3.33	6.66	10	13.33	20
BR	$K=3$	3	3	3	3	3
	$K=4$	4	4	4	4	4
	$K=5$	5	5	5	5	5
	$K=10$	10	10	10	10	10

When PS is implemented, the following corollary is straightforward due to simple qualitative rules can be deduced.

Corollary 5.2.6. *For the system with the redundancy-d topology, where server implement PS and capacities $\mu_k = 1 + \frac{M-1}{K-1}(k-1)$, for $k = 1, \ldots, K$, the stability condition is given by: $\lambda^{R,PS} = \frac{MK}{d}$, for $d > 1$, while $\lambda^B = K$. Hence, the redundancy system outperforms the stability condition of the Bernoulli routing if and only if $M \geq d$.*

If $M \geq d$, redundancy is a factor M/d better than Bernoulli. Hence, increasing

M, that is, the heterogeneity among the servers, is significantly beneficial for the redundancy system, as we can observe in Table 5.2. However, the stability condition of the redundancy system degrades as the number of copies d increases.

5.2.2 Nested redundancy systems

We consider the following nested redundancy topologies: N, W, WW (Figure 1.1 (b), (c), (d)) and $WWWW$. The latter is a $K = 8$ server system that is composed of 2 WW models and an additional job type $c = \{1, \ldots, 8\}$ for which all servers are compatible. For all three models, we assume that a job is with probability $p_c = 1/|\mathcal{C}|$ of type c.

Let us start by analyzing the stability conditions under the N-model. The stability condition of the N-model with Bernoulli routing is given by the following expression:

$$\lambda^B = \begin{cases} 2\min\{\mu_1, \mu_2\}, & \text{if } p = 0 \\ 2\mu_1/(1-p), & \text{if } 0 \leq p \leq \left(\frac{\mu_2-\mu_1}{\mu_1+\mu_2}\right)^+ \\ 2\mu_2/(1+p), & \text{if } \left(\frac{\mu_2-\mu_1}{\mu_1+\mu_2}\right)^+ < p \leq 1. \end{cases}$$

The above set of conditions is obtained from the fact that under Bernoulli routing, $\lambda^B = \min\{2\mu_1/(1-p), \mu_2/(p+\frac{1}{2}(1-p))\}$. Note that λ^B is a continuous function with a maximum $\mu_1 + \mu_2$ at the point $p = \frac{\mu_2-\mu_1}{\mu_1+\mu_2}$.

Comparing λ^B to $\lambda^{R,PS}$ (obtained in Section 4.2.2.2) leads to the following:

Corollary 5.2.7. *Under an N-model where servers implement PS, the stability condition under redundancy is larger than under Bernoulli routing under the following conditions: If $\mu_2 \leq \mu_1$, then $p \in ((\frac{2\mu_2-\mu_1}{2\mu_2+\mu_1})^+, 1)$. If $\mu_2 > \mu_1$, then $p \in (0, (\frac{\mu_2-2\mu_1}{\mu_2})^+) \cup (\frac{2\mu_2-\mu_1}{2\mu_2+\mu_1}, 1)$.*

From the above we conclude that under PS, if μ_1 is larger than $2\mu_2$, then redundancy is always better than Bernoulli, independent of the arrival rates of job types. For the case $\mu_2 > \mu_1$, we observe that for μ_2 large enough, redundancy will outperform Bernoulli.

In the reminder of this section, we consider the PS scheduling policy and W-based redundancy topologies. For the W-model, the stability condition of the Bernoulli system is given by the following expression, see Eq. (5.1):

$$\lambda^B = \begin{cases} \mu_1/(p_{\{1\}} + \frac{1}{2}p_{\{1,2\}}), & \frac{\mu_1}{\mu_1+\mu_2} < p_{\{1\}} + \frac{1}{2}p_{\{1,2\}} \\ \mu_1 + \mu_2, & \frac{\mu_1}{\mu_1+\mu_2} = p_{\{1\}} + \frac{1}{2}p_{\{1,2\}} \\ \mu_2/(1 - p_{\{1\}} - \frac{1}{2}p_{\{1,2\}}), & \frac{\mu_1}{\mu_1+\mu_2} > p_{\{1\}} + \frac{1}{2}p_{\{1,2\}} \end{cases}$$

The above conditions have been analyzed by analyzing the different cases with respect to $p_{\{1\}}$ and $p_{\{1,2\}}$, as in the N-model case.

In Table 5.3 we consider two settings of parameters, $\mu_k = \mu^{k-1}$ for $k = 1, \ldots, K$,

Table 5.3 The maximum arrival rates $\lambda^{R,PS}$ and λ^{B} in W-based redundancy topologies.

$\mu_k = \mu^{k-1}$		$\mu = 1$	$\mu = 1.2$	$\mu = 1.4$	$\mu = 2$	$\mu = 3$	μ^*
Red	W-model	1.5	1.8	2.10	3	3	1.33
	WW-model	2.33	4.03	4.90	7	7	1.19
	$WWWW$-model	3.75	8.64	10.5	15	15	1.17
$\mu_k = 1 + \frac{M-1}{K-1}(k-1)$		$M = 1$	$M = 2$	$M = 3$	$M = 4$	$M = 6$	
Red	W-model	1.5	1.8	2.10	3	3	
	WW-model	2.33	4.66	7	7	7	
	$WWWW$-model	3.75	7.14	10.71	12.85	15	
BR	W-model	2	2	2	2	2	
	WW-model	4	4	4	4	4	
	$WWWW$-model	8	8	8	8	8	

upper part of the table, and $\mu_k = 1 + \frac{M-1}{K-1}(k-1)$ for $k = 1, \ldots, K$, lower part of the table. We note that that $\lambda^B = K$ under both settings and that for the W model, server capacities are $\vec{\mu} = (1, \mu_2)$.

We observe that the similar results are obtained as when server capacities are $\mu_k = \mu^{k-1}$ for $k = 1, \ldots, K$. We denote by μ^* the value of μ for which $\lambda^{R,PS} = \lambda^B$. We observe that for PS as the number of servers duplicate, the μ^* decreases, and is always smaller than 1.5. So that, as the number of servers increases, the level of heterogeneity that is needed in order for redundancy to outperform Bernoulli decreases too.

When instead servers capacities are $\mu_k = 1 + \frac{M-1}{K-1}(k-1)$ for $k = 1, \ldots, K$, we observe that when PS is implemented and $M \geq K$ the stability condition under redundancy equals $\lambda^{R,PS} = |\mathcal{C}|$, which is always larger than $\lambda^B = K$. However, as the number of servers increases, the maximum capacity of the servers, M, needs to increase M in order for redundancy to outperform Bernoulli.

5.3 Numerical analysis

In the present section we investigate impact of redundancy in the performance of multi-server systems. To do so, we consider a system where servers implement either FCFS, or PS, or ROS and compare the performance under redundancy scheduling to that of the Bernoulli routing and JSQ. We have implemented a simulator and particularly evaluate the redundancy system and compare to the following two systems:

- when the scheduling policy is Bernoulli routing. In Section 5.2 we compare the stability condition analytically.

- when the scheduling policy Join the Shortest Queue (JSQ) policy according to which each job is dispatched to the compatible server that has the least number of jobs (ties are broken at random). In a recent paper, [30], it was shown that

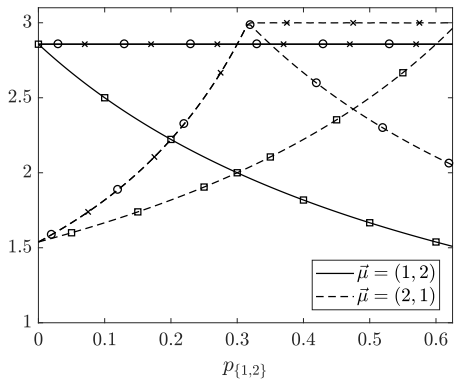


Figure 5.1 W-model with $p_{\{1\}} = 0.35$ and $p_{\{2\}} = 1 - p_{\{1\}} - p_{\{1,2\}}$. Servers have capacities $\vec{\mu} = (1,2)$ (solid line) and $\vec{\mu} = (2,1)$ (dashed line). Depict the stability condition under redundancy when PS is implemented, $\lambda^{R,PS}$ ($\circ$), under Bernoulli routing λ^{B} ($\square$) and under JSQ λ^{J} ($\times$).

> JSQ – with exponential service time distributions – combined with size-unaware scheduling policies such as FCFS, PS or ROS, is maximally stable, i.e., if there exists a static dispatching policy that achieves stability, so will JSQ.

Our simulations consider a large number of busy periods (10^6), so that the variance and confidence intervals of the mean number of jobs in the system are sufficiently small.

In Figure 5.1 and Figure 5.2, we consider the W-model with exponential service time distributions for which either FCFS, PS or ROS is implemented in the servers. We set $p_{\{1\}} = 0.35$ and $p_{\{2\}} + p_{\{1,2\}} = 0.65$, and vary the value of $p_{\{1,2\}}$. We consider either $\vec{\mu} = (1,2)$ or $\vec{\mu} = (2,1)$. We note that when $p_{\{1,2\}}$ increases, the load in server 1 increases as well, whereas the load in server 2 stays constant.

In Figure 5.1 we plot $\lambda^{R,PS}$ ($\circ$), λ^{B} ($\square$) and λ^{J} ($\times$), for capacities $\vec{\mu} = (1,2)$ and $\vec{\mu}(2,1)$. We use the analysis in Section 4.2 to obtain the stability when servers implement PS and the dispatching policy is redundancy, the analysis in Section 5.1 to obtain the stability when that is Bernoulli routing and [30] when JSQ. We note that either for Bernoulli routing or JSQ the stability condition is invariant to the scheduling policy implemented in the servers in this case.

In Figure 5.1, we observe that when servers implement PS, redundancy consistently has a larger stability region than Bernoulli in the $\vec{\mu} = (1,2)$ case and for $p_{\{1,2\}} \in [0, 0.5)$ in the $\vec{\mu} = (2,1)$ case. We let λ^J be the value of λ such that JSQ is stable if $\lambda < \lambda^J$ and unstable if $\lambda > \lambda^J$. Using [30],

$$\lambda^J = \max_{p_{c,s} \geq 0, \sum_{s \in c} p_{c,s} = 1} \left\{ \min_{s \in S} \frac{\mu_s}{\sum_{c \in \mathcal{C}(s)} p_{c,s} p_c} \right\}.$$

We observe that the stability condition under redundancy coincides on a large region with that of JSQ, which, in view of the results of [30], implies that redundancy is in that region maximally stable.

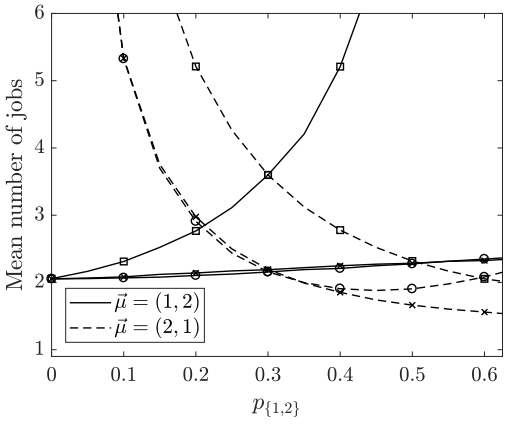
In Figure 5.2, we depict the mean number of jobs in the system when either redundancy ($\circ$), or Bernoulli routing ($\square$), or JSQ ($\times$) is the scheduling policy respectively when FCFS ((a) and (b)), PS ((c) and (d)) and ROS ((e) and (f)) is implemented in the servers. When analyzing the present figures, we will come back to Figure 5.1 in order to observe the stability condition at each study case.

When $\vec{\mu} = (1, 2)$, we observe that redundancy performs better than Bernoulli routing, under the three FCFS, PS and ROS. This difference becomes larger as $p_{\{1,2\}}$ increases. This is due to the fact that the redundancy policy does better in exploiting the larger capacity of server 2 than Bernoulli, which becomes more important as $p_{\{1,2\}}$ increases. In addition, we note that for redundancy, Bernoulli routing and JSQ, the mean number of jobs increases as $p_{\{1,2\}}$ increases. The reason for this is that as $p_{\{1,2\}}$ increases, the load on server 1 increases. Since server 1 is the slow server, this increases the mean number of jobs.

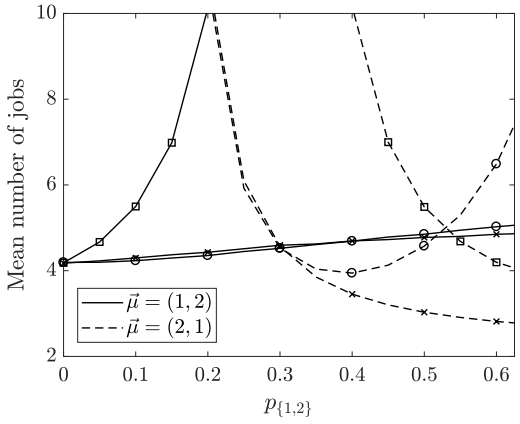
In the opposite case, i.e., $\vec{\mu} = (2, 1)$, the mean number of jobs is non-increasing in $p_{\{1,2\}}$. This is because as $p_{\{1,2\}}$ increases, the load on server 1 increases. Since server 1 is now the fast server, this has a positive effect on the performance (decreasing mean number of jobs). However, as $p_{\{1,2\}}$ gets larger, the additional load (created by the copies) makes that the performance can be negatively impacted. This happens for $\lambda = 2$, where the mean number of jobs under redundancy is a U-shape function, more pronounced under either FCFS or PS than under ROS. We furthermore observe that in the $\vec{\mu} = (2, 1)$ case, redundancy outperforms Bernoulli for any value of $p_{\{1,2\}}$ when $\lambda = 1.5$, for the three FCFS, PS and ROS. However, when $\lambda = 2$, Bernoulli outperforms redundancy under both FCFS and PS, for FCFS when $p_{\{1,2\}} > 0.51$ and for PS when $p_{\{1,2\}} > 0.49$. This is due to the additional load, generated under redundancy, that becomes more pronounced as $p_{\{1,2\}}$ becomes larger. We note that under ROS, redundancy performs better than Bernoulli routing for any $p_{\{1,2\}}$.

Let us focus on JSQ, we observe that when servers implement either FCFS ((a) and (b)) or ROS ((e) and (f)), JSQ outperforms redundancy. When $\vec{\mu} = (1, 2)$, the different is rather small and almost inappreciable under FCFS. However, when $\vec{\mu} = (2, 1)$, we observe that this difference increases as $p_{\{1,2\}}$ increases. When servers instead implement PS, we observe that under both $\vec{\mu} = (1, 2)$ and $\vec{\mu} = (2, 1)$, JSQ outperforms redundancy. For small values of $p_{\{1,2\}}$ the difference is rather small, however it becomes larger as $p_{\{1,2\}}$ increases due to the additional load that redundancy creates. However this improvement does not come for free, as JSQ requires precise information of the queue lengths at all times.

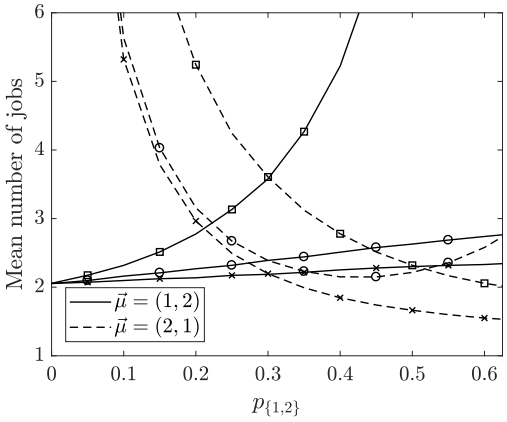
In Figure 5.3 we depict the performance of the W model for different values of μ_2,



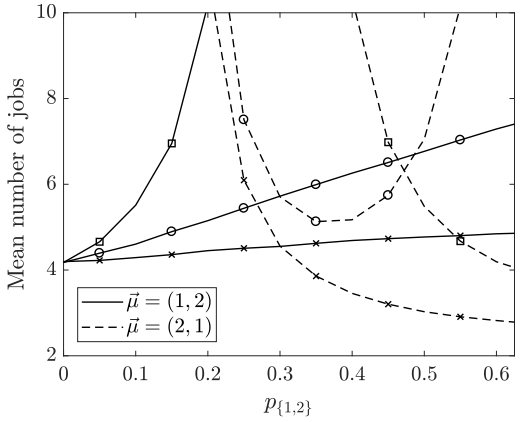
(a) FCFS, $\lambda = 1.5$



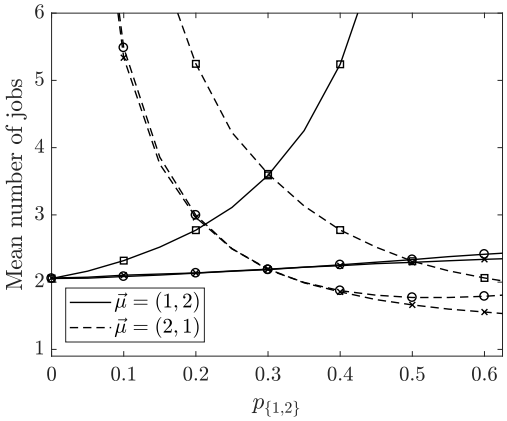
(b) FCFS, $\lambda = 2$



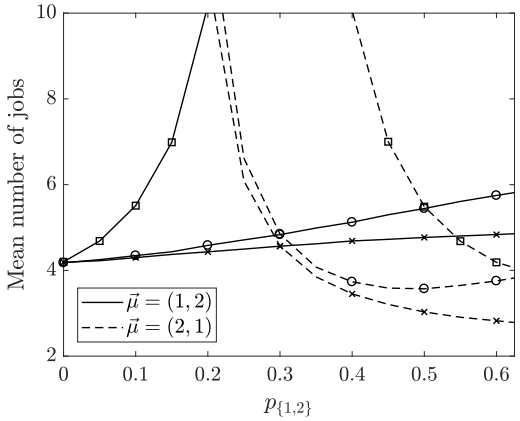
(c) PS, $\lambda = 1.5$



(d) PS, $\lambda = 2$

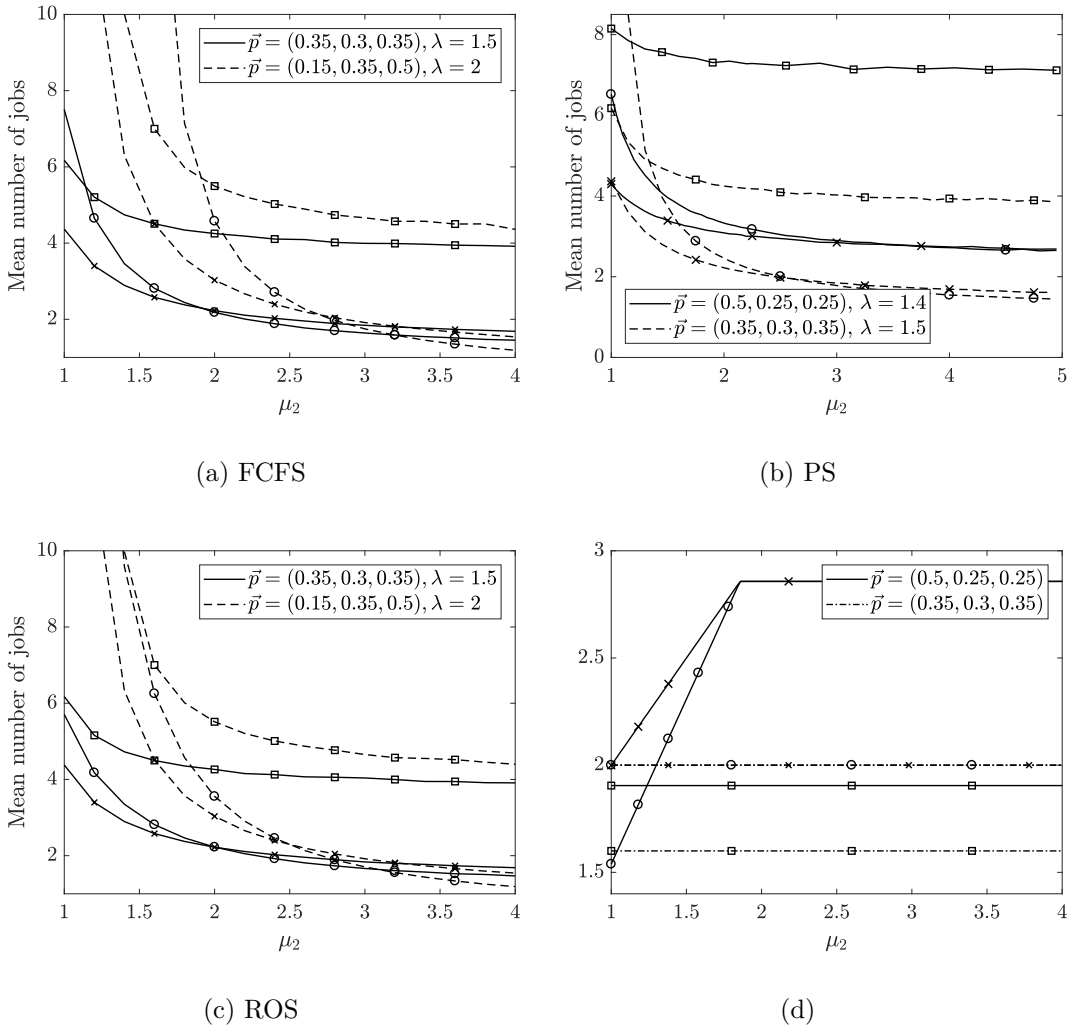


(e) ROS, $\lambda = 1.5$



(f) ROS, $\lambda = 2$

Figure 5.2 W-model with (a) and (b) FCFS servers, (c) and (d) PS servers and (e) and (f) ROS servers. Fixed parameter $p_{\{1\}} = 0.35$ and $p_{\{2\}} = 1 - p_{\{1\}} - p_{\{1,2\}}$. Depict the mean number of jobs under redundancy ($\circ$), Bernoulli routing ($\square$) and JSQ ($\times$) for $\lambda = 1.5$ and $\lambda = 2$.



(a) FCFS

(b) PS

(c) ROS

(d)

Figure 5.3 W-model with servers (a) FCFS, (b) PS and (c) ROS, for fixed parameters $\vec{p}$ and $\mu_1 = 1$. (a), (b), (c) depict the mean number of jobs under redundancy ($\circ$), Bernoulli routing ($\square$) and JSQ ($\times$), and (d) depicts the stability regions $\lambda^{R,PS}$, λ^{B} and λ^{J}.

while keeping fixed $\vec{p} = (p_{\{1\}}, p_{\{2\}}, p_{\{1,2\}})$ and $\mu_1 = 1$. In Figure 5.3 (a), (b) and (c) we plot the mean number of jobs and we see that when servers implement FCFS (a), PS (b) and ROS (c), for both configurations of $\vec{p}$, the performance of the redundancy, Bernoulli routing and JSQ improve as μ_2 increases. In Figure 5.3 (d), we depict the stability condition under redundancy, Bernoulli routing and JSQ, when PS is implemented.

In Figure 5.3 (d) we observe that the stability condition under redundancy with PS servers is larger than that under Bernoulli routing and, coincides with that under JSQ on a large region, which implies that the redundancy system is maximally stable in that region.

In Figure 5.3 (a), (b) and (c) we observe that under either FCFS, or PS, or ROS, the gap between redundancy and Bernoulli routing is significant as μ_2 grows large, in both cases of $\vec{p}$. We note that under PS, the stability condition under Bernoulli routing has larger than that redundancy when μ_2 is small (Figure 5.3 (d)), and hence, the performance under redundancy is worst. We note that for ROS, the performance under redundancy is always better than that under Bernoulli routing.

We also observe that redundancy can outperform JSQ as μ_2 grows large. Intuitively, we can explain this by observing that for very large values of μ_2, with both redundancy and JSQ, all jobs of type $p_{\{1,2\}}$ get served in server 2. We observe in Figure 5.3 (d) that the stability conditions with redundancy and JSQ are very similar, for PS.

Chapter 6

Impact of scheduling in redundancy models

In the present chapter we investigate the impact of the scheduling policy on the performance of redundancy systems. We do so by stochastically comparing the number of jobs under different scheduling polices. We focus on redundancy-aware policies that are composed of two levels. The first level (Π_1) describes the priority among the job types and the second level (Π_2) describes how the jobs with the same priority are served in a server. Examples of first-level policies are LRF and MRF, while second-level policies could be FCFS or ROS. This two-level policy is denoted by $\Pi_1 - \Pi_2$.

Most of the literature in this topic considers jobs with i.i.d. copies. For the flexible redundancy model, Koole and Righter [69] show that serving the same job in all servers minimizes the number of jobs in the system when service times are NWU. For a constrained redundancy model, more particularly, the nested model, Gardner et al. [41] prove that LRF-FCFS minimizes the number of jobs for exponentially distributed service times.

In this chapter, we consider redundancy systems with a general redundancy topology where jobs follow a general service time distribution and have either i.i.d. copies or identical copies. Under the i.i.d. copies assumption, we show that for the nested redundancy model with exponential service times, LRF-Π_2 minimizes the number of jobs in the system independently of the non-idling second-level policy Π_2. That is, we generalize the result in [41] to any non-idling second-level policy Π_2. Furthermore, when service times are NWU, we show that for a given non-idling first-level policy Π_1, Π_1-FCFS minimizes the number of jobs in the system. The latter is motivated by the fact that under NWU service times and i.i.d. copies, the service time of a copy that enters service is stochastically smaller than the remaining service time of a copy of that job that is already in service, which increases the chance that the job departs sooner.

Under the identical copies assumption, we prove that for the nested redundancy model MRF-Π_2, and particularly MRF-ROS, outperforms MRF-FCFS for any service time distributions, with Π_2 non-idling. The latter is motivated by the fact that when copies are identical, all the copies of each job have the same size, which induces a waste of resources when serving copies of the same job.

We also show that under Π_1-FCFS, the total number of jobs when having i.i.d. copies is stochastically smaller than when having identical copies.

Lastly, we discuss the stability condition for the redundancy-aware scheduling policies analyzed in the present chapter. In particular, for the redundancy system with a general topology and heterogeneous server capacities with exponentially distributed service times, we show that (i) for LRF-Π_2 with i.i.d. copies and non-idling Π_2, and (ii) for LRF-ROS with any correlation structure among the copies, the stability region is not reduced due to adding redundant copies.

The present chapter is organized as follows: In Section 6.1 we give a detailed description of the model. In Section 6.2 we stochastically compare the performance of the redundancy systems with respect to the scheduling policy (Section 6.2.1), and with respect to the correlation structure among the copies (Section 6.2.2). In Section 6.3 we discuss and characterize the stability condition of the redundancy-aware scheduling policies implemented in this chapter. In Section 6.4 we implement a simulator in order to asses the previous results and obtain further insights.

6.1 Model description

We consider the same setting as in Chapter 4. We consider a K parallel server system with heterogeneous capacities μ_s, for $s \in S$ and where $S = \{1, \dots, K\}$ is the set of all servers. Jobs arrive to the system according to a Poisson process of rate λ. Each job is labeled with a type c that represents the subset of servers to which it sends a copy: i.e., $c = \{s_1, \dots, s_n\}$, where $n \leq K$, $s_1, \dots, s_n \in S$ and $s_i \neq s_l$, for all $i \neq l$. We denote by $\mathcal{C}$ the set of all types in the system. An arriving job is with probability p_c of type c, with $\sum_{c \in \mathcal{C}} p_c = 1$. Let us denote by $\mathcal{C}(s) = \{c \in \mathcal{C} \ : \ s \in c\}$ the subset of types that dispatch a copy into server s. We generally assume a nested redundancy topology, that is, for all job types $c, c' \in \mathcal{C}$, either *i)* $c \subseteq c'$ or *ii)* $c' \subseteq c$ or *iii)* $c \cap c' = \emptyset$.

In this chapter, we assume that the service times of the jobs follow a general distribution, which is independent across jobs. Special attention will be given to exponential and New-Worse-than-Used (NWU) service time distributions. See Section 2.1 for more details on NWU distributions.

We denote by π a generic scheduling policy implemented in the system. In this chapter, we introduce two-level scheduling policies:

- The first level determines the priority among job types.
- The second level determines the scheduling policy of jobs with the same priority. This policy is assumed to be non-idling.

We refer to this two-level policy as $\pi = \Pi_1 - \Pi_2$, where Π_1 refers to the first level priority and Π_2 refers to the second level scheduling policy. Examples of first level policies Π_1 are Least-Redundant-First (LRF) and Most-Redundant-First (MRF), which can either be preemptive or non-preemptive. We say that a first level priority policy Π_1 is a strict priority policy, if in each server s there is a strict priority ranking of all job types in $\mathcal{C}(s)$ and the server serves the job type present with the highest priority. For example, LRF and MRF are strict priority policies for the nested redundancy models. Examples of second level policies Π_2 are FCFS, PS and ROS.

We assume that the correlation structure among the copies is either i.i.d. or identical copies. For a given scheduling policy π, we denote by $N_c^\pi(t)$ the number of type-c jobs present in the system at time t when jobs have i.i.d. copies and by $N^\pi(t) = \sum_{c\in\mathcal{C}} N_c^\pi(t)$ the total number of jobs at time t. We denote by $L_c^\pi(t)$ the number of type-c jobs in the system at time t under identical copies and by $L^\pi(t) = \sum_{c\in\mathcal{C}} L_c^\pi(t)$ the total number of jobs at time t.

In this chapter, we compare the total number of jobs through stochastic ordering. See Section 2.1 for more details on stochastic ordering.

6.2 Stochastic comparison results

In this section we analyze how the scheduling policy (Section 6.2.1) and the copy correlation structure (Section 6.2.2) affect the performance of the system.

6.2.1 Comparison of the scheduling policy

For a given system, we compare the total number of jobs with respect to the scheduling policy implemented in the system.

6.2.1.1 I.i.d. copies and exponential service times

We first assume that service times are exponentially distributed with i.i.d. copies. Due to the memoryless property, we can show that the number of jobs is insensitive to the implemented second-level policy Π_2. This result holds when the first-level policy Π_1 is a strict priority policy.

Lemma 6.2.1. *Consider a redundancy system with a general topology and heterogeneous server capacities, where jobs have exponentially distributed service times and i.i.d. copies.*

Then, for any non-idling Π_2 *and* Π_2', $\{N^{\Pi_1-\Pi_2}(t)\}_{t\geq 0} =_{st} \{N^{\Pi_1-\Pi_2'}(t)\}_{t\geq 0}$, *where* Π_1 *is a strict preemptive priority policy.*

Proof: Since Π_1 is a strict priority policy, exactly one type of job has priority at any given time in a given server. Because of the i.i.d. copies assumption and exponentially distributed service times, within a job type, all non-idling policies are equivalent. □

In Lemma 6.2.1 we show an insensitivity result with respect to the second-level scheduler, under the condition that Π_1 is a strict priority policy. In Section 6.4 (Figure 6.2), we will show that if this condition is not satisfied, the second-level policy does have an impact on the number of jobs in the system.

In the following proposition we characterize the scheduling policy that stochastically minimizes the number of jobs under exponentially distributed service times and i.i.d. copies. This proposition generalizes that in [41], where the authors prove that for any policy π, $\{N^{LRF-FCFS}(t)\}_{t\geq 0} \leq_{st} \{N^{\pi}(t)\}_{t\geq 0}$.

Proposition 6.2.2. *Consider a redundancy system with a nested topology and heterogeneous server capacities where jobs have exponentially distributed i.i.d. copies. Then,* $\{N^{LRF-\Pi_2}(t)\}_{t\geq 0} \leq_{st} \{N^{\pi}(t)\}_{t\geq 0}$ *with preemptive LRF and any policy* π.

Proof: In [41] it was proven that $\{N^{LRF-FCFS}(t)\}_{t\geq 0} \leq_{st} \{N^{\pi}(t)\}_{t\geq 0}$ for any policy π. Together with Lemma 6.2.1 this gives the result. □

6.2.1.2 I.i.d. copies and NWU service times

When jobs have i.i.d. copies and NWU service time distributions, the service time of a copy that is already in service is stochastically larger than that of a copy that has not received service yet. Hence, this suggests that whenever a server becomes available, it will be better to serve a copy of a job that has already a copy elsewhere in service, to have it leave faster. That is exactly what policy Π_2=FCFS does. In the result below we show that, given a first-level policy Π_1, FCFS is indeed optimal. The proof is deferred to the Appendix 6.5.A.

Proposition 6.2.3. *Consider a redundancy system with a general topology and heterogeneous server capacities, where* Π_1 *is a strict preemptive priority policy, jobs service times follow a NWU distribution and copies are i.i.d.. Then,* $\{N^{\Pi_1-FCFS}(t)\}_{t\geq 0} \leq_{st} \{N^{\Pi_1-\Pi_2}(t)\}_{t\geq 0}$, *for all* $t \geq 0$ *and any second-level non-idling policy* Π_2 *such that it serves one copy at a time and once started on a copy, it gives that copy priority over all other copies of the same type.*

This result is in agreement with the results obtained in [69]. In that paper, the authors show that in a flexible redundancy system with NWU service times and i.i.d. copies, it is optimal to maximize the number of copies of a job in service.

For exponential service times and nested topologies, the optimal first-level policy Π_1 is LRF. However, this does not extend to NWU service time distributions. We refer to Section 6.4 (Figure 6.3) where we observe that for the W-model LRF performs well when the variability of the service time distribution is low; however, MRF outperforms LRF when the variability of this distribution is high.

6.2.1.3 Identical copies

For the homogeneous server system when jobs have identical copies, each server will need the same amount of time to serve a copy. Hence, it is never optimal to assign a copy of a job that is already in service (an 'old job') to an idle server, regardless of the service time distribution. In the following proposition, we show that for the heterogeneous server system when the first-priority level is preemptive MRF, the number of jobs under Π_2 is stochastically smaller than that under FCFS. The proof is deferred to the Appendix 6.5.A.

Proposition 6.2.4. *Consider a redundancy system with a nested topology and heterogeneous server capacities, where the service times of the jobs are sampled from a general distribution and copies are identical. Then,* $\{L^{MRF-FCFS}(t)\}_{t\geq 0} \geq_{st} \{L^{MRF-\Pi_2}(t)\}_{t\geq 0}$ *with preemptive MRF and where* Π_2 *is non-idling and such that it serves one copy at a time and once started on a copy, it gives that copy priority over all other copies of the same type.*

The key property to prove the above result is that in a nested redundancy system with preemptive MRF-FCFS, all copies of a job enter service simultaneously and complete service in the server with highest capacity, wasting resources on the other servers serving this job. Hence, the system behaves as if the each job is served in its feasible server with the highest capacity, while the other feasible servers idle until that job is departed.

It is tempting to believe that Proposition 6.2.4 remains valid when MRF is replaced by any other preemptive strict priority policy. We refer to Section 6.4 (Figure 6.4) where we observe that for the W-model, the mean number of jobs for LRF-ROS is smaller than that under LRF-FCFS. However, we did not succeed in finding a proof for this claim.

6.2.2 Comparison between i.i.d. copies and identical copies

In the present section we investigate how the correlation structure among the copies affects the performance of the system. That is, we consider the total number of jobs per

type in the system with i.i.d. copies, $N(t)$, and compare this stochastically to the system with identical copies, $L(t)$. The proofs in this section are deferred to the Appendix 6.5.A.

In Chapter 3 we prove that the stability condition for the redundancy-d (with $d > 1$) model is larger under i.i.d. copies than under identical copies for FCFS. This is due to the fact that, when service times are exponential and copies are i.i.d., the departure rate of a subset of busy server is given by the sum of all the service rates, whereas under identical copies it is given by the sum of the rates of servers giving service to different jobs. Hence, the departure rate under i.i.d. copies is at least as large as that under identical copies. In the following lemma we show that the jobs with i.i.d. copies leave no later than with identical copies.

Proposition 6.2.5. *Consider a redundancy system with a general topology and heterogeneous server capacities, where the service times of the jobs are sampled from a general distribution. Then, for any preemptive policy Π_1, we have that $\{N^{\Pi_1-FCFS}(t)\}_{t\geq 0} \leq_{st} \{L^{\Pi_1-FCFS}(t)\}_{t\geq 0}$.*

We note the previous proposition also holds when the scheduling policy is (single-level) FCFS when compared with another single-level policy, such as ROS. That holds by taking Π_1 the policy where all the job types have the same priority.

In the following, we aim to generalize the previous result to general (not necessarily Π_1-FCFS) scheduling policies. We first provide a lemma that shows that idling is non-optimal. We then prove in the proposition that any system with identical copies, can be improved upon by a policy with i.i.d. copies.

Lemma 6.2.6. *Consider a redundancy system with a general topology and heterogeneous server capacities where service times follow a general distribution. For any preemptive, possibly idling policy π that is allowed to sample for each job either i.i.d. copies or identical copies, there exists a non-idling preemptive policy, π', that takes the same decision regarding sampling i.i.d. or identical copies as π, such that $\{N'(t)\}_{t\geq 0} \leq_{st} \{N(t)\}_{t\geq 0}$.*

The next proposition follows from Lemma 6.2.6 using arguments similar to those in the proof of Lemma 6.2.5.

Proposition 6.2.7. *Consider a redundancy system with a general topology and heterogeneous server capacities where service times follow a general distribution. For any preemptive, possibly idling policy with identical copies, π, there exists a non-idling preemptive policy with i.i.d. copies, π', such that $\{N^{\pi'}(t)\}_{t\geq 0} \leq_{st} \{L^{\pi}(t)\}_{t\geq 0}$.*

6.3 Stability condition under the LRF and MRF policies

In this section, we discuss the stability conditions under first-level LRF and MRF scheduling policies. We note that the stability conditions under FCFS and ROS scheduling policies are analyzed in Chapters 3 and 4. The proofs in this section are deferred to the Appendix 6.5.B.

I.i.d. copies

We first assume the nested redundancy topology and that copies are i.i.d.. The stability condition under preemptive LRF-Π_2 with exponentially distributed service times is straightforward from Proposition 6.2.2 and [46]. In particular, the result shows that preemptive LRF-Π_2 is maximally stable. Below, we prove that the same holds true for non-preemptive LRF-Π_2.

Proposition 6.3.1. *Consider a redundancy system with nested topology, where jobs are exponentially distributed with unit mean and have i.i.d. copies, and LRF-Π_2 is implemented. When LRF policy is either preemptive or non-preemptive, the system is stable if for all $c \in \mathcal{C}$,*

$$\lambda \sum_{\tilde{c} \subseteq c} p_{\tilde{c}} < \sum_{s \in c} \mu_s.$$

The system is unstable if there exists $\hat{c} \in \mathcal{C}$ such that $\lambda \sum_{\tilde{c} \subseteq \hat{c}} p_{\tilde{c}} > \sum_{s \in \hat{c}} \mu_s$.

Proof: The stability result when the LRF policy is preemptive follows directly from Proposition 6.2.2 and [46]. From Proposition 6.2.2, the stability region under FCFS must be at least as large as that under FCFS. For exponential service times, the latter is maximally stable [46], and hence, so is LRF-Π_2 with preemptive LRF. The proof when LRF policy is non-preemptive can be found in Appendix 6.5.B. □

The assumption that Π_1=LRF is crucial in order for the maximum stability result to hold. To see this, we refer to Example 6.3.5 where we will show that the N-model is not maximally stable under either MRF-FCFS, or MRF-ROS. The assumptions of a nested topology is crucial in the proof of Proposition 6.3.1. However, numerics suggest that the maximum stability result could also be valid when the topology is non-nested. Unfortunately, we did not succeed in proving this.

Non-i.i.d. copies

In the following proposition we discuss the stability condition when copies are non-i.i.d.. We first show that LRF-ROS is in this setting maximally stable. The proof is deferred to the Appendix 6.5.B.

Proposition 6.3.2. *Consider a redundancy system with nested topology, where jobs are exponentially distributed with unit mean and copies follow some general correlation structure and LRF-ROS is implemented, with LRF either preemptive or non-preemptive. The system is stable if for all $c \in \mathcal{C}$,*

$$\lambda \sum_{\tilde{c} \subseteq c} p_{\tilde{c}} < \sum_{s \in c} \mu_s.$$

The system is unstable if there exists $\hat{c} \in \mathcal{C}$ such that $\lambda \sum_{\tilde{c} \subseteq \hat{c}} p_{\tilde{c}} > \sum_{s \in \hat{c}} \mu_s$.

For a nested redundancy topology, assuming that first-level policy is LRF implies that given the state of the system, the job type in service at each server is completely characterized. In contrast to the ROS scheduling policy (analyzed in Section 4.3.2), the fact that we are given the job type in service simplifies the analysis of the system. Furthermore, since the second-level policy is ROS, when the number of jobs in service is large, the probability that more than one copy of the same job is simultaneously in service is close to zero, regardless of the correlation structure among the copies. Hence, when the scheduling policy is LRF-ROS, we can completely characterize the instantaneous departure rate of the system when in the fluid limit.

For non-nested systems, or two-level policies other than LRF-ROS, we did not succeed in deriving the stability conditions. We do however show, in the examples below, that this stability condition will not be maximally stable. We consider first a numerical example (Example 6.3.3) of a non-nested system under LRF-ROS with identical copies and observe that it is not maximally stable.

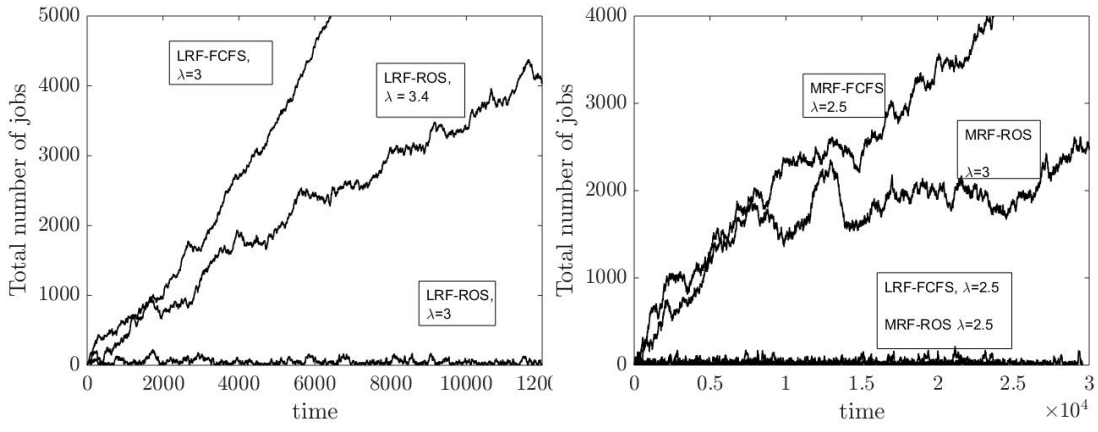


Figure 6.1 The redundancy topology where each job dispatches copies to either 2 or 3 servers out of 4 chosen uniformly at random with homogeneous server capacities. The trajectory of the total number of jobs for exponentially distributed service times and identical copies for scheduling policies Π_1-Π_2 with Π_1=LRF, MRF and Π_2=FCFS, ROS for various arrival rates.

Example 6.3.3. *We consider 4 homogeneous servers with unit capacities, and jobs dispatch either 2 or 3 copies chosen uniformly at random, and copies are identical. The maximum stability condition is $\lambda < 4$ [46]. However, in Figure 6.1 we observe that already for $\lambda = 3.4$ the system is not stable for any of the policies Π_1-Π_2, with $\Pi_1 \in \{LRF, MRF\}$ and $\Pi_2 \in \{FCFS, ROS\}$, including the LRF-ROS policy.*

Regarding the stability condition, we can also draw the following observations for two-level priority policies from the figure. We observe that the stability region under LRF-Π_2 is larger than that under MRF-Π_2. For LRF-ROS we observe that the stability region is at least $\lambda < 3$, whereas for MRF-ROS, we observe that already for $\lambda = 3$ the system is unstable. Similarly, we observe that the stability region for LRF-FCFS is at least $\lambda < 2.5$, whereas for MRF-FCFS already for $\lambda = 2.5$ the system is unstable. Lastly in Figure 6.1 we also observe that the stability region under Π_1-ROS is larger than that under Π_1-FCFS.

The following corollary is straightforward from Proposition 6.2.4.

Corollary 6.3.4. *Consider a redundancy system with nested topology, where jobs are exponentially distributed with unit mean and identical copies. The stability condition under preemptive MRF-ROS, is at least as large as than that under preemptive MRF-FCFS.*

We made the same observation for non-preemptive MRF in Example 6.3.3. We note that the above corollary does not give us any stability condition, since for both MRF-ROS and MRF-FCFS, the stability condition is unknown.

In the following example we discuss the stability condition for a nested model, but under two-level policies other than LRF-ROS. We observe that these systems are not maximally stable.

Example 6.3.5. ***N*-model:** *We assume the N-model where servers have heterogeneous capacities $\vec{\mu} = (\mu_1, \mu_2)$, and p $(1-p)$ is the probability that a job is of type $\{2\}$ $(\{1,2\})$. The maximum stability condition when jobs have exponentially distributed service times is given by $\lambda p < \mu_2$ and $\lambda < \mu_1 + \mu_2$.*

*In the case of **MRF**-Π_2, with Π_2 non-idling, we have that type-$\{2\}$ jobs can only be served if there is no type-$\{1,2\}$ job present in the system. Let us denote by μ_{Π_2} the mean departure rate of type-$\{1,2\}$ jobs in the system and by $\rho_1 = \lambda(1-p)/\mu_{\Pi_2}$. Then μ_{Π_2} is given by $\max\{\mu_1, \mu_2\}$ for identical copies, and by $\mu_1 + \mu_2$ for i.i.d. copies. In both cases, type-$\{2\}$ can only be served $(1-\rho_1)$ fraction of the time. Thus, the stability condition is given by $\lambda p < \mu_{\Pi_2}$ and $\lambda(1-p) < \mu_2(1-\rho_1)$.*

*In the case of **LRF-FCFS**, with identical copies when type-$\{2\}$ is present in the system, the total departure rate is given by $\mu_1 + \mu_2$, assuming that type-$\{1,2\}$ is also present. Then, when type-$\{2\}$ is no longer there, $\rho_2 = \lambda(1-p)/\mu_2$ fraction of the time,*

the total departure rate is time-varying and given by $\mu_{12}(t) := \alpha(t)\mu_1 + (1-\alpha(t))\mu_2$*, where* $\alpha(t)$ *is either 0 or 1. Note that* $\mu_{12}(t) \leq \max\{\mu_1, \mu_2\}$*. Therefore, the stability condition is given by* $\lambda(1-p)/\mu_2$*,* $\lambda < \mu_1 + \mu_2$ *and* $\lambda p < \tilde{\mu}_{12}(1-\rho_2)$*, where* $\tilde{\mu}_{12} \leq \max\{\mu_1, \mu_2\}$*.*

We observe that for both examples, the stability region is reduced when adding redundant copies.

6.4 Numerical analysis

We have implemented a simulator in order to investigate the impact of the scheduling policies on the performance of redundancy models. In particular, we simulate the performance under the policies FCFS, ROS, LRF-FCFS, LRF-ROS, MRF-FCFS and MRF-ROS, where MRF and LRF are assumed to be preemptive. Our simulations consider a large number of busy periods (10^6), so that the variance and confidence intervals of the mean number of jobs in the system are sufficiently small.

In this section, we consider deterministic, exponential, and degenerate hyperexponential service time distributions, with unit mean. We recall that under the degenerate hyperexponential distribution, with probability q the service requirement is exponentially distributed with parameter μq, and is 0 otherwise. We set $\mu = 1$. Hence, the mean service time equals $\mu = 1$ (independent of q) and the squared coefficient of variation equals $C^2 = \frac{2}{q} - 1$, which increases as q decreases. We note that when $q = 1$, this is an exponential distribution, with $C^2 = 1$. The deterministic distribution is NBU and the degenerate hyperexponential distribution is NWU, while the exponential distribution is both.

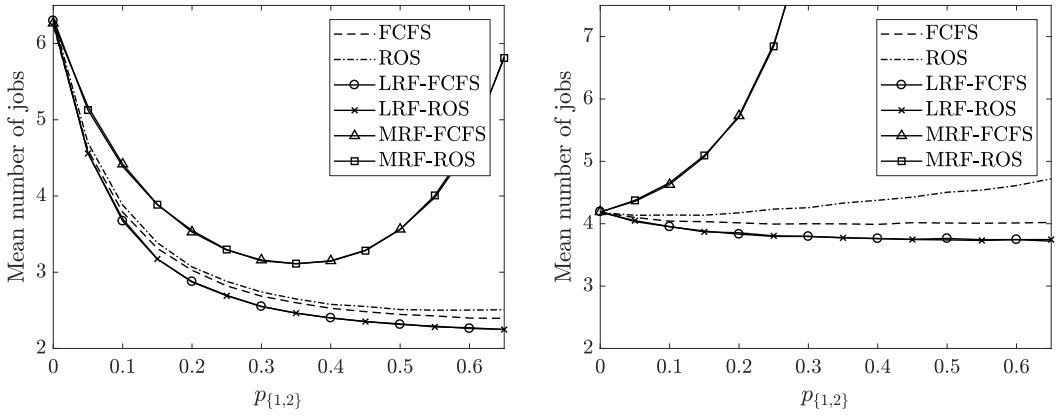
We note that the stability condition is unknown for most of the policies we simulate. Due to this reason, we choose the value of the arrival rate λ in such a way that the performance differences are perceivable, while preserving (seemingly) stability.

6.4.1 I.i.d. copies

In Figure 6.2 we consider a W-model and compare the performance under different scheduling policies for exponential service times. We assume homogeneous capacities $\vec{\mu} = (1, 1)$ and fix $\lambda = 1.3$. We plot the mean number of jobs with respect to $p_{\{1,2\}}$, with $p_{\{1\}} = 0.35$ and $p_{\{2\}} = 1 - p_{\{1\}} - p_{\{1,2\}}$.

We draw three main observations form Figure 6.2. Firstly, let us focus on the mean response time for policies $\pi = \Pi_1$-Π_2 for fixed Π_1 and Π_2=FCFS, ROS. We observe that the mean number of jobs under Π_1-FCFS and Π_1-ROS coincide for both Π_1=LRF and, Π_1=MRF. We note that LRF and MRF are first-level strict priority policies for the W-model. This observation is in agreement with the result obtained in Lemma 6.2.1.

Secondly, we focus on policies FCFS and ROS. For these policies the first-level



(a) $\vec{\mu} = (1, 1)$ and $\lambda = 1.3$

(b) $\vec{\mu} = (1, 2)$ and $\lambda = 2$

Figure 6.2 The mean number of jobs for the W-model with respect to $p_{\{1,2\}}$, with fixed $p_{\{1\}} = 0.35$ and $p_{\{2\}} = 1 - p_{\{1\}} - p_{\{1,2\}}$ with exponential service times and i.i.d. copies.

policy Π_1 treats all types equally, and hence is not a strict priority policy. We observe that the performance depends on the scheduling policy implemented in the servers and that FCFS outperforms ROS. Note that this is not in disagreement with Lemma 6.2.1, since there it was assumed that the first-level policy was strict priority. The differences between FCFS and ROS are pronounced when servers are heterogeneous ($\vec{\mu} = (1, 2)$) and there are many redundant jobs ($p_{\{1,2\}}$ approaches $1 - p_{\{1\}}$). The latter can be explained as follows. In the extreme case $p_{\{1,2\}} = 1 - p_{\{1\}}$, there is no traffic of type-$\{2\}$ jobs. By giving priority to type-$\{1\}$ jobs in server 1, type-$\{1, 2\}$ jobs will have the fast server for themselves, and thus the amount of time that servers idle while there are jobs to serve in the system is reduced. This is most likely to happen under FCFS.

Thirdly, we observe in Figure 6.2 that LRF-Π_2, with Π_2=FCFS, ROS, outperform the other policies. That is in agreement with Proposition 6.2.2.

In Figure 6.3 we consider the W-model and compare the performance under different scheduling policies assuming degenerate hyperexponential service times. We assume $\lambda = 1.3$, homogeneous capacities $\vec{\mu} = (1, 1)$ and $p_c = 1/3$ for all $c \in \mathcal{C}$. We plot the mean number of jobs with respect to q. We note that the coefficient of variation $C^2 = 2/q - 1$ is a decreasing function such that $C^2 \to \infty$ as $q \to 0$, and $C^2 = 1$ when $q = 1$. Recall that for $q = 1$, the distribution is exponential with coefficient of variation 1.

In Figure 6.3, let us focus on the redundancy-aware policies $\pi = \Pi_1$-Π_2. We observe that Π_1-FCFS (solid line) outperforms Π_1-ROS (dashed line) for Π_1=LRF ($\times$), and Π_2=MRF ($\circ$). These are in agreement with Lemma 6.2.3. This observation also holds when Π_1 treats all types equal, that is, we observe that FCFS outperforms ROS.

In Figure 6.3 we also observe that as q approaches 1 (that is, the distribution

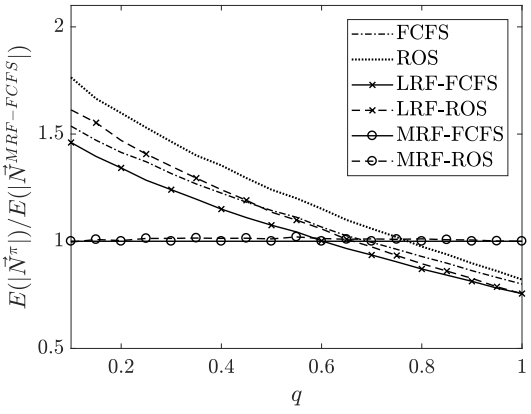


Figure 6.3 The mean number of jobs for the W-model with respect to q for capacities $\vec{\mu} = (1, 1)$, $\lambda = 1.3$ and $p_c = 1/3$ for all $c \in \mathcal{C}$. Assume degenerate hyperexponential service times with parameter q and i.i.d. copies.

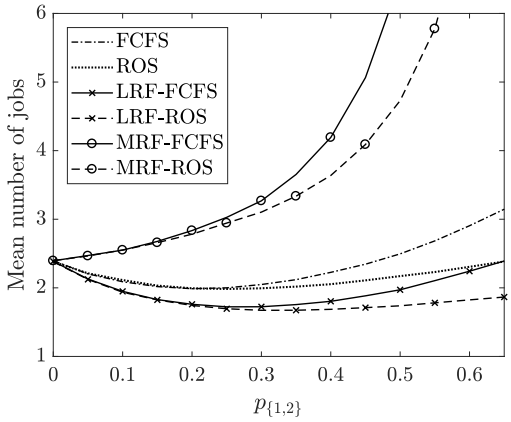
approaches the exponential distribution), LRF-FCFS outperform the other policies. When q equals 1, i.e., the exponential distribution, we observe that the performance under LRF-FCFS and LRF-ROS coincide, which is in agreement with Lemma 6.2.1 and Proposition 6.2.2. When instead q approaches 0, that is $C^2 \to \infty$, we observe that MRF-FCFS outperforms all other scheduling policies. This observation can be explained as follows: When the service time of the copies is not that variable, the service policy that is more work-conserving improves the mean number of jobs. However, when the service times are very variable, the policy that maximizes the number of copies of the same job in service, that is, MRF, improves the performance. In Chapter 8 we draw a conjecture based on these observations.

6.4.2 Identical copies

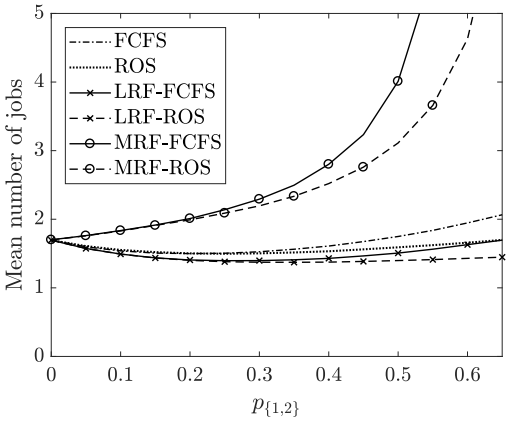
In Figure 6.4 we consider the W-model and plot the mean number of jobs with respect to $p_{\{1,2\}}$, with $p_{\{1\}} = 0.35$ and $p_{\{2\}} = 1 - p_{\{1\}} - p_{\{1,2\}}$. We assume identical copies and homogeneous capacities $\vec{\mu} = (1, 1)$ and $\lambda = 1$. We consider three service time distributions: (a) exponential, (b) deterministic, and (c) degenerate hyperexponential with parameter $q = 0.1$. We note that for deterministic service times, there is no distinction between i.i.d. and identical copies.

We observe that MRF-ROS outperforms MRF-FCFS for any service time distribution and any load. This is in agreement with the results in Proposition 6.2.4.

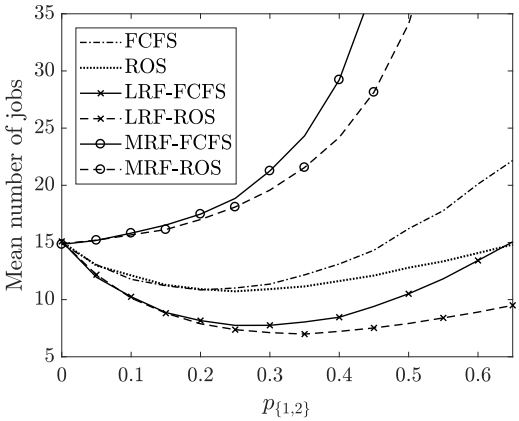
We also draw the following observations from Figure 6.4. Firstly, we observe that when the first-level policy is either LRF or no-priority, then ROS outperforms FCFS in the second-level, for any service time distribution. Secondly, for a given second-level



(a) Exponential distribution



(b) Deterministic



(c) Degenerate hyperexponential with $q = 0.1$

Figure 6.4 The mean number of jobs for the W-model with respect to $p_{\{1,2\}}$, with capacities $\vec{\mu} = (1, 1)$, $\lambda = 1$, $p_{\{1\}} = 0.35$ and $p_{\{2\}} = 1 - p_{\{1\}} - p_{\{1,2\}}$ with identical copies.

policy, we observe that LRF outperforms MRF in the first-level, for any service time distribution. As a consequence, we also observe that LRF-ROS outperforms all other policies, for any service time distribution. These observations can be explained as follows. When copies are identical, having several copies of the same job in service will degrade the performance when servers are homogeneous. We note that for a given second-level policy, LRF minimizes the number of copies of the same job in service. Furthermore, for a given first-level policy, ROS minimizes the number of copies of the same job in service.

Let us focus on Figure 6.4 (b) deterministic service times. We note that under deterministic service times the correlation of the copies does not affect the performance

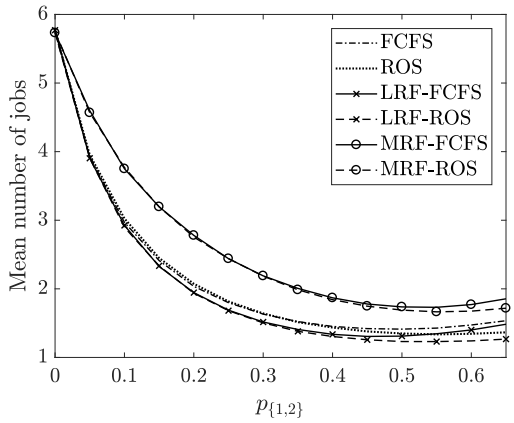


Figure 6.5 The mean number of jobs for the W-model with respect to $p_{\{1,2\}}$, with capacities $\vec{\mu} = (2, 1)$, $\lambda = 1.3$, $p_{\{1\}} = 0.35$ and $p_{\{2\}} = 1 - p_{\{1\}} - p_{\{1,2\}}$ with identical copies and exponential service time distributions.

of the system. Hence, the figure is also valid for i.i.d. copies. We can then make the following observation for i.i.d. copies: for degenerate hyperexponential service times (which is of NWU type) it is optimal to set Π_2=FCFS, while for deterministic service times (which is of NBU type), the numerical example shows that Π_2=ROS performs better. This observation for NBU service time distributions and i.i.d. copies is in agreement with the result in [69], where the authors prove that for a two-server system with a flexible redundancy structure, no-replication maximizes the number of completed jobs at any time instant.

Finally, in Figure 6.4 we observe that for a given policy π, the performance deteriorates as the variability of the service time increases. We note that the coefficient of variation of the distributions exponential, deterministic and degenerate hyperexponential are 1, 0, and 19, respectively.

In Figure 6.5 we consider the W-model with heterogeneous server capacities $\vec{\mu} = (2, 1)$ and plot the mean number of jobs with respect to $p_{\{1,2\}}$, with $p_{\{1\}} = 0.35$ and $p_{\{2\}} = 1 - p_{\{1\}} - p_{\{1,2\}}$. We assume identical copies, $\lambda = 1.3$ and exponential service times. We observe that with heterogeneous capacities, the same observations as for Figure 6.4 can be made.

6.5 Appendix

A: Proofs of Section 6.2

Proof of Proposition 6.2.3: We prove that when there are two jobs of the same type to be served, it is always better to serve a copy of the job that already has a copy in service in another server, than to serve a copy of the job that has no other copies in service. That is a stronger result than the claim of the proposition. The argument is similar to that of [44], but we outline the proof here briefly for completeness.

We assume that at some time t, under policy π a server starts to serve a copy of a job that has no other copies in service, say this copy is from job 2 and in server 1. Additionally, there is a job in server 1 of the same type as job 2 that has copies being served in another server(s), say that this is job 1. Let $\mathcal{A}$ be the set of servers that job 1 has already received some service on and let $\mathcal{B}$ be the set of servers that can serve job 1 but have not yet started to serve job 1.

We let π' serve job 1 on server 1 at time t and thereafter always serve job 1 whenever π serves job 2 or job 1 on any server in $\mathcal{B}$, until either job 1 or job 2 completes service under π, at some time τ. Otherwise we let π' agree with π until time τ (including when π serves job 1 on servers in $\mathcal{A}$). We couple the service times of the copies of jobs 1 and 2 on server 1 under the two policies, and let all other service times and arrival times be the same under both policies. At time τ, under policy π', job 1 has completed service and job 2 has not yet received service at any of its compatible servers. Under policy π, either job 1 or job 2 has completed service. Let us denote by a the job that did not complete service under π, a is either 1 or 2. We note that job a received service partially at some servers under policy π, say servers in the set $\mathcal{J}$. We couple the (new) service times of job 2 under π' on servers $\mathcal{J}$, denoted by S'_{2j} for $j = 1, \ldots, |\mathcal{J}|$, with the (used, or remaining) service times of job a, denoted by S'_{aj} for $j = 1, \ldots, |\mathcal{J}|$. Therefore, $S'_{2j} \leq S_{aj}$ with probability 1 for all $j = 1, \ldots, |\mathcal{J}|$. We can do this from the NWU assumption. We let π' serve a copy of job 2 whenever π serves a copy of job a from time τ on, until job 2 completes under π'. Thereafter, we let the servers that are serving a copy of job a in π idle in π', and let it otherwise π' agree with π. Then $\{\vec{N}'(t)\}_{t\geq 0} \leq \{\vec{N}(t))\}_{t\geq 0}$ with probability 1.

By repeating this argument, one obtains that the FCFS scheduling policy which is possibly idling is optimal. Then, one can easily verify that non-idling FCFS policy is better than idling FCFS, by following the steps of the proof of Lemma 6.2.5. □

Proof of Proposition 6.2.4: Assume we start at time $t = 0$ with an empty system. We note that for the nested redundancy topology, under first-level preemptive MRF, if any server is serving a type-c job, all servers $s \in c$ are also serving a type-c job. Moreover, when the second-level policy is FCFS, all these servers serve the same type-c

job and, because MRF is preemptive, all the copies of that job entered service at the same time. The job will leave due to completion of the copy in the server with the highest capacity μ_s, with $s \in C$. We denote by $s^*(c) := \arg\max_{s \in c} \mu_s$ the server with the highest capacity for type c. Hence, under MRF-FCFS, every time a type-c job enters service, it is as if the server $s^*(c)$ serves the job and the other servers $s \in c \backslash s^*(c)$ idle until that job is either served or preempted by a higher priority job (which due to MRF preempts all the servers serving that job).

Consider a general MRF-Π_2 policy and let S be the system under this policy and $L(t)$ the number of jobs. Consider an alternate system, S', with $L'(t)$ jobs, that follows the same MRF-Π_2 policy, but whenever a job of type c completes on a server that is distinct from server $s^*(c)$, a new type-c job immediately replaces it. If the job had already a copy in service on server $s^*(c)$, this copy is kept in service, and all other copies are replaced. Then $L'(t) \geq_{st} L(t)$. By coupling the service completions on server $s^*(c)$ for all c, we have $L'(t) = L^{MRF-FCFS}(t)$ for all t with probability 1, and the result follows. □

Proof of Proposition 6.2.5: We actually show a stronger result: we prove that the system with i.i.d. copies is optimal when for each job, we can choose either independent or identical copies. We consider a policy π where servers are required to follow Π_1-FCFS and idling is allowed. Under policy π, whenever a server starts serving a first copy of a job, that server determines whether the service times of the copies of that job are identical or i.i.d.. This decision is independent of the history of the process. We show that for this policy π, if at time t it idles or schedules identical copies for a job in service for the first time, we can construct a policy π' with a coupled sample path, such that at time t it does not idle or samples i.i.d. copies, and such that $\{N'(s)\}_{s \geq 0} \leq \{N(s)\}_{s \geq 0}$ with probability 1, where N (N') denotes the number of jobs in the system under policy π (π'). The result follows by starting at time 0 and repeating the argument each time a policy deviates from the policy π, until we have the non-idling Π_1-FCFS policy with i.i.d. copies.

We first show that idling is non-optimal. Assume that at time t, π idles a server during some time τ, when it has copies to serve according to Π_1-FCFS. Say this is server 1 that idles τ units of time and the next copy to serve in server 1 with highest priority is of job 1. We let π' agree with π for all of its copies correlation decisions, and let their arrival times and service times be coupled. Moreover, we let π' agree with π until time t. At time t, under π' server 1 serves the copy of job 1 from time t to time $t + \min\{\tau, s, r - t, a - t\}$, where s is the remaining service time of the copy of job 1 on server 1, r is the earliest completion time of all other copies of job 1 running on other servers and a is the arrival time of a higher priority job than job 1. Three different events can occur:

- $\tau = \min\{\tau, s, r-t, a-t\}$. We let π' idle server 1 after time $t+\tau$ whenever π serves job 1 on server 1, until it has received service for τ units of time in that server. Let π' otherwise agree with π for all time after $t+\tau$ and all servers. Then the systems under the two policies will be in the same states at time $t+\tau$, and $\{N'(s)\}_{s\geq 0} = \{N(s)\}_{s\geq 0}$ wp 1.

- $s = \min\{\tau, s, r-t, a-t\}$. We let π' idle the servers serving job 1 after time $t+s$ whenever π serves job 1 on those servers, until job 1 departs under π and let it otherwise agree with π. We let π' agree with π thereafter. At the time that job 1 departs under π, the two systems will be in the same state, but job 1 will have departed earlier under π', so $\{N'(s)\}_{s\geq 0} \leq \{N(s)\}_{s\geq 0}$ with probability 1.

- $r-t = \min\{\tau, s, r-t, a-t\}$. Then the job departs at the same moment in both systems. Hence, π and π' will be in the same state at time $t+\tau$, and letting π' agree with π from then on, $\{N'(s)\}_{s\geq 0} = \{N(s)\}_{s\geq 0}$ with probability 1.

- $a-t = \min\{\tau, s, r-t, a-t\}$. At time a, job 1 is preempted in π'. Then, we let π' idle server 1 after time a whenever π servers job 1 on server 1, until it has received service for $a-t$ units of time in that server (like job 1 received under policy π'). Let π' otherwise agree with π for all time after a. Then the systems under the two policies will be in the same states at time a, and $\{N'(s)\}_{s\geq 0} = \{N(s)\}_{s\geq 0}$ with probability 1.

Now let us assume from our argument above, that π never idles but that at some time t, π starts serving the first copy of job 1 and chooses identical copies. We let π' agree with π before time t and let it choose i.i.d. copies for job 1. We denote by τ the time that job 1 completes service under π, say at server 1. Because the copies are identical, server 1 has done the most work on job 1 between times t and τ. We couple the service time of job 1 on server 1 under π' to that under π. Then, π' samples i.i.d. copies for the rest of the copies of job 1. We let all other service times and all arrival times be coupled under both policies. We let π' agree with π until job 1 departs under π', at time τ'. From our coupling, $\tau' \leq \tau$. Let π' idle any server that serves job 1 under π between times τ' and τ and let it otherwise agree with π from time τ' on. From our argument above, a policy that agrees with π' but does not idle will have even earlier departures than π'. The argument can be repeated, each time reducing the number-in-system process, until we have all i.i.d copies and no idling. □

Proof of Lemma 6.2.6: The structure of this proof is similar to that of Lemma 6.2.5. We consider a policy π where idling is allowed. We show that if at time t policy π idles, we can construct a policy π' with a coupled sample path, such that at time t it does not idle and $\{N'(s)\}_{s\geq 0} \leq \{N(s)\}_{s\geq 0}$ with probability 1.

We assume that at time t, π idles a server during a fixed amount of time τ, when there is a copy of another job to be served. We let π' agree with π for all of its copies correlation decisions, and let their arrival times and service times be coupled. Moreover, we let π' agree with π until time t. At time t, under π' server 1 serves the copy of job 1 from time t to time $t + \min\{\tau, s, r - t\}$, where s is the remaining service time of the copy of job 1 on server 1, and r is the earliest completion time of all other copies of job 1 running on other servers. We let π' agree with π for all other servers from time t to $t + \tau$. Thereafter, whenever π serves job 1 on server 1, let π' also serve job 1 on server 1 if it has not completed under π' ($s = \min\{\tau, s, r - t\}$) and otherwise let it idle server 1 ($\tau = \min\{\tau, s, r - t\}$), and let π' agree with π for all other jobs and servers. We note that if $r - t = \min\{\tau, s, r - t\}$, both systems are in the same state at time r. Then $\{N'(s)\}_{s\geq 0} \leq \{N(s)\}_{s\geq 0}$ with probability 1. □

Proof of Proposition 6.2.7: By similar arguments as those in Lemma 6.2.5, we can construct a policy $\tilde{\pi}$ that is possibly idling and dispatches i.i.d. copies. From Lemma 6.2.6, for that policy $\tilde{\pi}$, that only chooses i.i.d. copies, there exists a policy π' that is non-idling and schedules i.i.d. copies. □

B: Proofs of Section 6.3

Proof of Proposition 6.3.1 and Proposition 6.3.2

In order to prove both propositions, we analyze the fluid-scaled system. We recall that the redundancy structure is nested and that $N_c(t)$ denotes the number of type-c jobs at time t. For $r > 0$, we denote by $N_c^r(t)$ the system where the initial state satisfies $N_c^r(0) = rn_c(0)$, for all $c \in \mathcal{C}$. We write the fluid-scaled number of jobs per type by using standard arguments, see [26],

$$\frac{N_c^r(rt)}{r} = n_c(0) + \frac{1}{r}\tilde{A}_c(rt) - \frac{1}{r}\tilde{S}_c(T_c^r(rt)), \tag{6.1}$$

where $\tilde{A}_c(t)$ and $\tilde{S}_s(t)$ are independent Poisson processes having rates λp_c and 1, respectively. $T_c^r(t)$ is the cumulative amount of capacity spend in serving type-c jobs, which strongly depends upon the correlation structure among the copies, that is,

$$T_c^r(t) = g((T_{s,c}^r(t))_{s\in c}),$$

where $T_{s,c}^r(t)$ is the cumulative amount of capacity spend on serving type-c jobs in server $s \in c$ during the time interval $(0, t]$ and g is characterized by the correlation structure of the copies. We note that when copies are i.i.d. $T_c^r(t) = \sum_{s\in c} T_{s,c}^r(t)$ and when copies are identical, $T_c^r(t) = \max_{s\in c}\{T_{s,c}^r(t)\}$.

In the following result, we obtain the general characterization of a fluid limit. The existence of fluid limits can be proved following the same steps as in Lemma 3.2.1, so its proof is omitted.

Lemma 6.5.1. *For almost all sample paths ω and sequence $r_k \to \infty$, there exists a subsequence $r_{k_j} \to \infty$ such that for all $c \in \mathcal{C}$ and $t \geq 0$,*

$$\lim_{j\to\infty} \frac{N_c^{r_{k_j}}(r_{k_j}t)}{r_{k_j}} = n_c(t) \text{ } u.o.c^1 \quad \text{and} \quad \lim_{j\to\infty} \frac{T_c^{r_{k_j}}(r_{k_j}t)}{r_{k_j}} = \tau_c(t) \quad u.o.c., \tag{6.2}$$

with $(n_c(\cdot), \tau_c(\cdot))$ continuous functions. In addition,

$$n_c(t) = n_c(0) + \lambda p_c t - \tau_c(t), \tag{6.3}$$

where $n_c(t) \geq 0$, $\tau_c(0) = 0$, $\tau_c(t) \leq t \max_{s\in c}\{\mu_s\}$, and $\tau_c(\cdot)$ are non-decreasing and Lipschitz continuous functions for all $c \in \mathcal{C}$.

In order to provide the characterization of the fluid limit, we first introduce some notation. Let us group the types with respect to their number of copies: In $\mathcal{L}_1$, the jobs of types with a single copy, that is, $\mathcal{L}_1 = \{c \in \mathcal{C} : |c| = 1\}$. We denote by $\mathcal{L}_i$ the jobs of types with i copies, that is, for $i = 2, \ldots, |C|$,

$$\mathcal{L}_i = \{c \in \mathcal{C} : |c| = i\}.$$

From the nested structure of the system, we note that for each $c \in \mathcal{L}_i$ and $\tilde{c} \in \mathcal{L}_j$ with $j < i$, either $\tilde{c} \subset c$ or $\tilde{c} \cap c = \emptyset$. For all $c \in \mathcal{L}_i$, let us denote by $\mathcal{L}_i(c) = \{\tilde{c} \subset c\}$ the job types that are subsumed in type c, for $i = 1, \ldots, |C|$.

In Section 3.5, we consider the redundancy-d model when server implement ROS, and show that in the fluid limit each server serves a copy of a job that is not being served at any other server in the system. In the following lemma we show that the latter is also true for a nested redundancy topology where Π_1-ROS is implemented in the servers. As in Section 3.5, we denote by $P_s(\vec{N})$, the probability that when $\vec{N}(0) = \vec{N}$, at time $t = 0$ a given server s is serving a copy that is not in service in any other server. Then, the following lemma is true.

Lemma 6.5.2. *Consider a redundancy system with a nested topology and where servers implement Π_1-ROS. For any server $s \in S$ and $\vec{N}^r(0) = r\vec{n}^r$, such that $\lim_{r\to\infty} \sum_{c\in\mathcal{C}(s)} rn_c^r > 0$, then*

$$\lim_{r\to\infty} P_s(r\vec{n}^r) = 1. \tag{6.4}$$

Proof: Assume at time 0, server s idles and that we are in state $\vec{N}^r(0) = \vec{N}$. At this moment, under Π_1-ROS server s can only serve a single type in the system, say type

c. Let us consider servers $l \in c$ that are serving a type-c job at time $t = 0$, say servers $S(c)$. We denote by $-\tilde{T}_l^r < 0$ the time at which server l started serving a new type c job, whether another job departed from the server, or it preempted a copy with less priority in that server. We note that $\frac{N_c(-\tilde{T}_l^r)-1}{N_c(-\tilde{T}_l^r)}$ is the probability that server l is *not* serving the copy of the same job that is now in service in server s. Hence,

$$P_s(\vec{N}) = \Pi_{l \in S(c), l \neq s} \frac{N_c^r(-\tilde{T}_l^r) - 1}{N_c^r(-\tilde{T}_l^r)}. \tag{6.5}$$

We set $\vec{N}(0) = r\vec{n}^r$. Since the transition rates μ_s and λ are of order $O(1)$, it follows directly that $\tilde{T}_s^r$ and $\vec{N}(-\tilde{T}_s^r) - \vec{N}(0)$ are of order $O(1)$ as well, so that

$$\lim_{r\to\infty} \frac{N_c^r(-\tilde{T}_l^r) - 1}{N_c^r(-\tilde{T}_l^r)} = \lim_{r\to\infty} \frac{N_c^r(0) - 1}{N_c^r(0)} = 1. \tag{6.6}$$

It hence follows from Eq. (6.5) that $\lim_{r\to\infty} P_s(r\vec{n}^r) = 1$. □

Let us characterize the instantaneous departure of a type-$\tilde{c}$ job. Let us denote by $\mathcal{C}_{\tilde{c}}$ the types that are a subsumed in type $\tilde{c}$. That is,

$$\mathcal{C}_{\tilde{c}} = \{c \in \mathcal{C} \ : \ c \subseteq \tilde{c}\}.$$

Note that if $\tilde{c} \in \mathcal{L}_i$, then $\mathcal{C}_{\tilde{c}} = \mathcal{L}_i(\tilde{c})$.

Lemma 6.5.3. *Assume that jobs have exponential service times and*

- *if copies are i.i.d. copies, assume LRF-Π_2, with LRF non-preemptive and Π_2 non-idling,*
- *if copies follow some general correlation structure, assume LRF-ROS.*

For each type $\tilde{c} \in \mathcal{C}$, the fluid limit $\sum_{c\in\mathcal{C}_{\tilde{c}}} n_c(t)$ satisfies the following:

$$\frac{\mathrm{d}\sum_{c\in\mathcal{C}_{\tilde{c}}} n_c(t)}{\mathrm{d}t} = \sum_{c\in\mathcal{C}_{\tilde{c}}} \lambda p_c - \sum_{s\in\tilde{c}} \mu_s, \quad \textit{if} \quad n_{\tilde{c}}(t) > 0.$$

Proof: We first consider a general correlation structure among the copies and LRF-ROS. When starting in state $\vec{N}(0) = r\vec{n}^r$, the drift function is

$$\begin{aligned} \tilde{f}(r\sum_{c\in\mathcal{C}_{\tilde{c}}} n_c^r) \ &= \ \sum_{c\in\mathcal{C}_{\tilde{c}}} \lambda p_c \\ &\quad - \sum_{s\in\tilde{c}} \mu_s \left(\prod_{l\in\tilde{c}} P_l(r\vec{n}^r)\right) - g_{\tilde{c}}(r\vec{n}^r)(1 - \prod_{l\in\tilde{c}} P_l(r\vec{n}^r)) \end{aligned} \tag{6.7}$$

for $n_{\tilde{c}} > 0$, where $g_{\tilde{c}}$ is a function that captures the instantaneous departure rate of the system when more than one copies (with any correlation structure) of the same job are in service, and note that $g_{\tilde{c}} = O(1)$ as $r \to \infty$. Assuming that $n_{\tilde{c}} > 0$, due to LRF and nested structure, each server $s \in \tilde{c}$ serve jobs of a single type-c, with $c \subseteq \tilde{c}$ which is the type with least redundant copies in that server s. We note that the first departure term in Eq. (6.7) represents departures from servers $s \in \tilde{c}$ of type-c jobs $c \in \tilde{c}$, who were served in one unique server. Since these jobs have no other copies in service, the total departure rate from the servers in $\tilde{c}$ equals $\sum_{s\in\tilde{c}} \mu_s$. The second departure term in Eq. (6.7) represents departures due to a type-c job that is being served in more than one server simultaneously.

Therefore, from equation 6.7 together with Lemma 6.5.2, we obtain

$$\lim_{r\to\infty} \tilde{f}(r \sum_{c\in\mathcal{C}_{\tilde{c}}} n_c^r) = \lambda \sum_{c\in\mathcal{C}_{\tilde{c}}} p_c - \sum_{s\in\tilde{c}} \mu_s. \tag{6.8}$$

Now assume copies are i.i.d. copies and servers implement LRF-Π_2 with LRF non-preemptive and Π_2 non-idling. Under these assumptions, the drift function is given by

$$\tilde{f}(r \sum_{c\in\mathcal{C}_{\tilde{c}}} n_c^r) = \lambda \sum_{c\in\mathcal{C}_{\tilde{c}}} p_c - \sum_{s\in\tilde{c}} \mu_s, \text{ when } n_{\tilde{c}} > 0. \tag{6.9}$$

This can be seen as follows. When there are type-c jobs present, $c \in \tilde{c}$, each server is working on such a job, possibly the same job. Because of the i.i.d. copies assumption, the departure rate of these jobs is simply the sum of all the capacities. □

In order to prove Proposition 6.3.1 and Proposition 6.3.2, we introduce the redundancy degree per type. Let us assume w.l.o.g. that $\mathcal{L}_1$ is non-empty. For each type $c \in \mathcal{L}_1$, the fluid limit $n_c(t)$ is given by the following due to Lemma 6.5.3:

$$\frac{\mathrm{d}n_c(t)}{\mathrm{d}t} = \lambda p_c - \sum_{s\in c} \mu_s, \quad \text{if} \quad n_c(t) > 0.$$

We note that $\lambda p_c - \sum_{s\in c} \mu_s < 0$, by hypothesis. The latter coincides with the fluid limit of an $M/M/1$ system with arrival rate λp_c and server capacity $\sum_{s\in c} \mu_s$. Hence, for all $c \in \mathcal{L}_1$, $n_c(t)$ reaches zero in finite time, say at time T_1, and stays zero.

The proof follows now by induction. Assume that there is a time T_j, such that $n_c(t) = 0$ for $t > T_j$, for all $c \in \mathcal{L}_i$ and $i \leq j$. In the following we show that for $\mathcal{L}_{j+1}$, there is a $T_{j+1} > T_j$, such that $n_c(t) = 0$ for $t > T_{j+1}$ and for all $c \in \mathcal{L}_{j+1}$.

For a type $c \in \mathcal{L}_{j+1}$, the fluid drift of $\sum_{\tilde{c}\in\mathcal{L}_{j+1}(c)} n_{\tilde{c}}(t)$ is given by the following due

to Lemma 6.5.3:

$$\frac{\mathrm{d}\sum_{\tilde{c}\in\mathcal{L}_{j+1}(c)} n_{\tilde{c}}(t)}{\mathrm{d}t} = \sum_{\tilde{c}\in\mathcal{L}_{j+1}(c)} \lambda p_{\tilde{c}} - \sum_{s\in c} \mu_s, \quad \text{if} \quad n_c(t) > 0. \tag{6.10}$$

From hypothesis, we note that there exits time T_j such that $n_{\tilde{c}}(t) = 0$ for all $\tilde{c} \in \mathcal{L}_i$ with $i \leq j$. Hence, $\frac{\mathrm{d}\sum_{\tilde{c}\in\mathcal{L}_{j+1}(c)} n_{\tilde{c}}(t)}{\mathrm{d}t} = \frac{\mathrm{d}n_c(t)}{\mathrm{d}t}$, when $t \geq T_j$. Together with Eq. 6.10,

$$\frac{\mathrm{d}n_c(t)}{\mathrm{d}t} = \sum_{\tilde{c}\in\mathcal{L}_{j+1}(c)} \lambda p_{\tilde{c}} - \sum_{s\in c} \mu_s, \quad \text{if} \quad n_c(t) > 0,$$

for all $t \geq T_j$. We note that $\lambda \sum_{\tilde{c}\in\mathcal{L}_{j+1}(c)} p_{\tilde{c}} - \sum_{s\in c} \mu_s < 0$, by hypothesis. The latter coincides with the fluid limit of an $M/M/1$ system with arrival rate $\lambda \sum_{\tilde{c}\in\mathcal{L}_{j+1}(c)} p_{\tilde{c}}$ and server capacity $\sum_{s\in c} \mu_s$. Hence, for all $c \in \mathcal{L}_{j+1}$, $n_c(t)$ reaches zero in finite time, say at time T_{j+1}, and stays zero. Hence, there exists time $\tilde{T} > 0$ when the fluid process is empty. Together with Theorem 2.4.10, we conclude that the system is stable. □

Chapter 7

Impact of mobility in cellular networks

In the present chapter we investigate the impact of mobility in cellular networks. In order to do so we consider the same open queuing network that was studied in Ganesh at al. [40]. This is a K parallel server system with heterogeneous capacities μ_i for $i = 1, \ldots, K$. Users arrive to server i according to a Poisson process of rate λ_i, with $\lambda = \sum_{i=1}^{K} \lambda_i$, and have exponentially distributed service times with unit mean. The time that a job spends in server i before it moves from that server to server j is exponentially distributed with rate αr_{ij}, where α is the parameter that controls mobility speed and $r_{ij} > 0$.

For a model where there is no mobility, i.e., $\alpha = 0$, the stability condition is given by $\max_{\{i=1,\ldots,K\}}\{\lambda_i/\mu_i\} < 1$. Meanwhile, for the system with mobility speed $\alpha > 0$, [40] shows that the stability condition upgrades up to $\sum_{i=1}^{K} \lambda_i < \sum_{i=1}^{K} \mu_i$, which is independent of the mobility speed. Motivated by the fact that mobility improves the stability condition of the system, we aim to characterize how mobility impacts the delay performance of its users.

We investigate the delay performance of the users with respect to the mobility parameter. Our main results state that mobility might not always minimizes the mean delay of the users. Moreover, we observe that the mean delay might not be monotone on the mobility parameter, implying that there exits some positive and finite α that achieves the minimum delay. The latter is in contrast to prior work, [17] among other examples, that showed that mobility is always beneficial for the user's delay perspective.

In order to do so, we analyze the mean number of jobs in the system, which by Little's law is proportional to the mean delay. Given the complexity of the model, exact analysis of steady state performance is not possible. We thus study the light-traffic regime of the mean number of jobs, i.e., when $\lambda \approx 0$, and consider two metrics to assess

the performance: (i) for any given $\alpha < \infty$, the mean number of jobs in the system, and (ii) the difference in the number of jobs between the extreme cases in which α is either 0 or ∞. For both measures, we obtain a sufficient set of conditions depending on the model parameters, such that for sufficiently low λ (i) the mean number of jobs decrease in, and (ii) the difference in the mean number of jobs for the extreme values of is negative.

The rest of the chapter is organized as follows. In Section 7.1, we give a detailed description of the model. In Section 7.2 we analyze the light-traffic approximation of the mean number of jobs with respect to the mobility parameter. In Section 7.3 we consider the system outside the light-traffic regime; we define the system as $\alpha \to \infty$ and analyze the difference between the mean number of jobs for the system with $\alpha = 0$ and $\alpha = \infty$. Section 7.4 presents numerical results. Proofs are deferred to Appendix 7.5.

7.1 Model description

We consider a K parallel server system with heterogeneous capacities μ_i for $i = 1, \dots, K$, where incoming jobs might move among the servers while receiving service. New jobs arrive to the system according to a Poisson process of rate λ and are routed towards server i with probability p_i, where $\sum_{i=1}^{K} p_i = 1$. Thus, there is an arrival at server i at rate $\lambda_i := \lambda p_i$. We assume that jobs have exponentially distributed service times with unit mean. We let $\bar{\mu} = \sum_{i=1}^{K} \mu_i$ denote the sum of the capacities of each server.

A job in server i, $i = 1, \dots, K$, moves to server j with an exponential rate αr_{ij}, where α is the parameter that controls the moving speed. Thus, while in the network users move according to the Q-matrix $\alpha R = (\alpha r_{ij})_{i,j}$. We assume that the corresponding irreducible Markov chain is independent across jobs and we denote by $\vec{\pi} = (\pi_1, \dots, \pi_K)$ its unique stationary distribution. We note that $\vec{\pi}$ does not depend on α.

The state of the system at time $t \geq 0$ is given by the number of jobs present at each server, $\vec{N}^{\alpha}(t) = (N_1^{\alpha}(t), \dots, N_K^{\alpha}(t))$, and lives in $\mathbb{Z}_+^K$. The non-zero components in the transition matrix of the process $\vec{N}^{\alpha}(t)$ are given by:

$$Q^{\alpha}(\vec{n}, \vec{m}) = \begin{cases} \lambda p_i, & \text{if } \vec{m} = \vec{n} + \vec{e}_i, \text{ for } i = 1, \dots, K \\ \mu_i, & \text{if } \vec{m} = \vec{n} - \vec{e}_i \text{ and } n_i^{\alpha} > 0, \text{ for } i = 1, \dots, K \\ \alpha n_i r_{ij}, & \text{if } \vec{m} = \vec{n} + \vec{e}_j - \vec{e}_i, \text{ for } i, j = 1, \dots, K, \ i \neq j \end{cases}$$

where $\vec{n}, \vec{m} \in \mathbb{N}^K$ and $\vec{e}_i = (0, \dots, 0, 1, 0, \dots, 0) \in \mathbb{R}^K$, with a 1 in the $i-$th position. Note that λp_i (μ_i) corresponds to an arrival to (a departure from) server i. Each job in server i moves into server j at rate αr_{ij}, hence, the total moving rate from server i to sever j is $\alpha n_i r_{ij}$.

From the stability viewpoint, with $\alpha = 0$, there is no mobility, and all nodes need to be stable separately, and the stability condition of the system is $\max_i\{\lambda p_i/\mu_i\} < 1$. On the other hand for any $\alpha > 0$, the network becomes interconnected, and in [40] it was shown that the stability condition becomes $\lambda < \bar{\mu}$. Thus, the stability region with mobility is always larger than in the case without mobility. We will denote by $\rho = \lambda/\bar{\mu}$ the load in the system.

Whenever the stability condition for given $\alpha \geq 0$ holds, we let $\vec{\Pi}^\alpha = (\Pi^\alpha(\vec{n}))_{\vec{n}\in\mathbb{Z}_+^K}$ denote the steady-state distribution of $\vec{N}^\alpha(t)$. From theory, we know that $\vec{\Pi}^\alpha$ is the unique solution of the balance equations given by:

$$\lambda\Pi^\alpha(\vec{0}) = \sum_{i=1}^{K} \mu_i \Pi^\alpha(\vec{e}_i), \tag{7.1}$$

and for states $\vec{n} \neq \vec{0}$

$$\begin{aligned}\left(\sum_{i=1}^{K}\lambda p_i + \sum_{\substack{i,j=1\\ i\neq j}}^{K}\alpha n_i \mathrm{r}_{ij} + \sum_{i=1}^{K}\mu_i 1(n_i>0)\right)\Pi^\alpha(\vec{n}) \\ = \sum_{i=1}^{K}(\lambda p_i \Pi^\alpha(\vec{n}-\vec{e}_i)1(n_i>0) + \mu_i\Pi^\alpha(\vec{n}+\vec{e}_i)) \\ + \sum_{\substack{i,j=1\\ i\neq j}}^{K}(n_j+1)\alpha \mathrm{r}_{ji}\Pi^\alpha(\vec{n}+\vec{e}_j-\vec{e}_i)1(n_i>0)\end{aligned} \tag{7.2}$$

where $1(\cdot)$ denotes the indicator function.

Our main performance measure is the mean number of jobs in steady state, which by Little's law is proportional to the mean delay. We denote by $\vec{N}^\alpha(\infty)$ a random variable distributed as $\vec{\Pi}^\alpha$. We recall that $\mathbb{E}_{\vec{n}}(\cdot) = \mathbb{E}(\cdot|\vec{N}^\alpha(0) = \vec{n})$ and $\mathbb{P}_{\vec{n}}(\cdot) = \mathbb{P}(\cdot|\vec{N}^\alpha(0) = \vec{n})$.

7.2 Light-traffic analysis for a 2-cell system

In the present section we obtain the light-traffic approximation for the system with $K = 2$. As we have introduced in Section 2.5, the light-traffic approximation corresponds to the first-order asymptotic expansion of the system as $\lambda \to 0$, see [105] for more details. That is, we consider that $\mathbb{E}(|\vec{N}^\alpha(\infty)|) = \lambda m^{LT}(\alpha) + o(\lambda)$ as $\lambda \to \infty$, for some $m^{LT}(\alpha) > 0$. In order to find an expression, we neglect states with more than one user, as these states will become negligible in the limit $\lambda \to 0$. The proof of this section are deferred to Appendix 7.5.A.

We note that the light-traffic analysis of general K is cumbersome, we thus focus on

the $K = 2$ case. However, as we will see later, this provides interesting insights on the performance of the system. When neglecting states with two or more users, the balance Eq. (7.1) and Eq. (7.2), simplify into the following system of equations:

$$\begin{aligned}
\lambda\Pi^{\alpha,LT}(0,0) &= \mu_1\Pi^{\alpha,LT}(1,0) + \mu_2\Pi^{\alpha,LT}(0,1) \\
(\mu_1 + \alpha r_{12})\Pi^{\alpha,LT}(1,0) &= \lambda p_1\Pi^{\alpha,LT}(0,0) + \alpha r_{21}\Pi^{\alpha,LT}(0,1) \\
(\mu_2 + \alpha r_{12})\Pi^{\alpha,LT}(0,1) &= \lambda p_2\Pi^{\alpha,LT}(0,0) + \alpha r_{12}\Pi^{\alpha,LT}(1,0)
\end{aligned} \tag{7.3}$$

By solving these balance equations we obtain the following results:

$$\begin{aligned}
\Pi^{\alpha,LT}(0,0) &= \frac{\alpha(\mu_1 r_{21}+\mu_2 r_{12})+\mu_1\mu_2}{\lambda(\alpha(r_{12}+r_{21})+p_1\mu_2+p_2\mu_1)+\alpha(\mu_1 r_{21}+\mu_2 r_{12})+\mu_1\mu_2} \\
\Pi^{\alpha,LT}(1,0) &= \frac{\lambda(\alpha r_{21}+p_1\mu_2)}{\lambda(\alpha(r_{12}+r_{21})+p_1\mu_2+p_2\mu_1)+\alpha(\mu_1 r_{21}+\mu_2 r_{12})+\mu_1\mu_2} \\
\Pi^{\alpha,LT}(0,1) &= \frac{\lambda(\alpha r_{12}+p_2\mu_1)}{\lambda(\alpha(r_{12}+r_{21})+p_1\mu_2+p_2\mu_1)+\alpha(\mu_1 r_{21}+\mu_2 r_{12})+\mu_1\mu_2}
\end{aligned}$$

which gives, as $\lambda \to 0$, (here $\approx$ has an informal sense, while $\sim$ is the usual leading asymptotic term)

$$\mathbb{E}(|\vec{N}^{\alpha}(\infty)|) \approx \Pi^{\alpha,LT}(1,0) + \Pi^{\alpha,LT}(1,0) \sim \lambda m^{LT}(\alpha)$$

with

$$m^{LT}(\alpha) = \frac{\alpha + \frac{p_1\mu_2+p_2\mu_1}{(r_{12}+r_{21})}}{\alpha(\mu_1\pi_1 + \mu_2\pi_2) + \frac{\mu_1\mu_2}{(r_{12}+r_{21})}} \tag{7.4}$$

where in this simple 2-node case we have $\pi_1 = \frac{r_{21}}{r_{12}+r_{21}}$ and $\pi_2 = \frac{r_{12}}{r_{12}+r_{21}}$.

Therefore, the following result on the behavior of the mean delay the in the light-traffic regime is straightforward.

Proposition 7.2.1. *If $\mu_1 = \mu_2$ or $p_1\pi_2\mu_2 = p_2\pi_1\mu_1$, then m^{LT} is constant. If $\mu_1 > \mu_2$, then m^{LT} is strictly increasing if $p_1\pi_2\mu_2 > p_2\pi_1\mu_1$ and strictly decreasing if $p_1\pi_2\mu_2 < p_2\pi_1\mu_1$.*

These conditions can be put in a more concise form as follows. Let in the sequel

$$C := \left(\sum_{i=1}^{K} \frac{p_i}{\mu_i}\right) - \frac{1}{\sum_{i=1}^{K} \mu_i\pi_i}. \tag{7.5}$$

Then, in the case $K = 2$ and $\mu_1 > \mu_2$, the above result can be restated by saying that m^{LT} is strictly increasing if $C < 0$ and strictly decreasing if $C > 0$. Eq. (7.5) can be interpreted as follows. The first term in C is the mean sojourn time of a user arriving to the network if it were not to move, while the second term is the mean sojourn time of a user moving at infinite speed. Thus $C < 0$ means that a user would depart sooner by not moving because it is more likely to have arrived to a favorable node, and so in

this case mobility should worsen the system's performance, which is indeed the content of Proposition 7.2.1 since in this case m^{LT} is increasing with α. It is interesting that, in the light-traffic regime, only the two extreme cases with zero and infinite speeds matter. The comparison between these two extreme cases is the purpose of the next section.

We observer that in in the particular case $K = 2$, the previous result shows that delay is monotone in the light-traffic regime. Moreover, it can be increasing or decreasing depending on the precise parameters as shown below. That is in contrast with what we announced in the introduction of this chapter, that the delay performance is not necessarily monotone, let it increasing, with the mobility speed. In Section 7.4 we show some examples to illustrate the behavior of $\mathbb{E}(|\vec{N}^{\alpha}(\infty)|)$ outside the light-traffic regime, where we observe that delay performance might not actually be monotonic in the mobility speed.

7.3 Arbitrary number of servers

The constant C introduced above suggests to compare the case without mobility to the case with infinity mobility, which is what we do here. The main difference with the preceding section is that we do not restrict ourselves to the case $K = 2$. To do so, we assume that $\lambda_i < \mu_i$ for every i, so that the systems with and without mobility are stable, and we compare the two extreme cases $\alpha = 0$ and $\alpha = \infty$ through the metric

$$\Delta := \mathbb{E}|\vec{N}^{0}(\infty)| - \mathbb{E}|\vec{N}^{\infty}(\infty)|.$$

In order to give sense to this metric, we first explain what we mean by the case $\alpha = \infty$.

7.3.1 Infinite speed system

Here we define the limiting process $\vec{N}^{\infty}$ corresponding to infinite speed. As the speed of mobility increases, the dynamics within the system can be decomposed into two types. On a relatively slow time scale, the total number of jobs changes due to an arrival or a departure, whereas on a relatively fast time scale, jobs move across servers. As $\alpha \to \infty$, one can expect a complete decomposition between these two dynamics which is indeed what happens.

This separation of time scales induces the following behavior in the limit. Conditioned on the total number of users in the system, since users move at infinite speed and thus forget instantaneously their initial location, at each point in time they are spread in the network according to $\vec{\pi}$ and their locations at different time instants are independent. Moreover, the total number of users evolves according to an $M/M/1$ queue with arrival rate λ but whose departure rate depends on the current queue length because some of the queues may be empty. More precisely, if there are x customers in the system, then queue

i will be nonempty with probability $1-(1-\pi_i)^x$ which gives an instantaneous service rate $\sum_{i=1}^K \mu_i(1-(1-\pi_i)^x)$. Thus, the limiting process is the process $(\vec{N}^\infty(t), t \geq 0)$ defined as follows:

- $|\vec{N}^\infty|$ is the $\mathbb{Z}_+$-valued birth-and-death process with non-zero transition rates $q(x, x+1) = \lambda$ and $q(x, x-1) = \sum_{i=1}^K \mu_i \left(1-(1-\pi_i)^x\right)$;

- let $T \subset R_+$ a finite set: conditioned on $|\vec{N}^\infty|$, $(\vec{N}^\infty(t), t \in T)$ are independent random variables such that $\vec{N}^\infty(t)$ follows a multinomial distribution with parameter $(|\vec{N}^\infty(t)|, \vec{\pi})$, i.e., for $\vec{n} = (n_1, \ldots, n_K)$ with $n_1 + \cdots + n_K = |\vec{N}^\infty(t)|$, we have

$$\mathbb{P}(\vec{N}^\infty(t) = \vec{n} \mid |\vec{N}^\infty|) = \frac{|\vec{N}^\infty(t)|!}{n_1! \cdots n_K!} \pi_1^{n_1} \cdots \pi_K^{n_K}. \tag{7.6}$$

We emphasize that because users move at infinite speed, in-between two times $s < t$ users move infinitely many times. In particular, the multi-dimensional process $\vec{N}^\infty$ is not càdlàg and its trajectory resembles a white noise process. This rough behavior prevents $\vec{N}^\infty$ from being a Markov process, although the sequence embedded at arrival and departure epochs is a Markov chain. Another related Markov process is the one-dimensional process $|\vec{N}^\infty|$ counting the total number of users: this process does not "see" the wild oscillations caused by users moving infinitely fast: it behaves smoothly and is a Markov process. In the following, by stationary distribution we mean a distribution such that if $\vec{N}^\infty(0)$ starts according to this distribution, the law of $\vec{N}^\infty(t)$ does not change over time. The following result describes the stationary behavior of $\vec{N}^\infty$.

Proposition 7.3.1. *$\vec{N}^\infty$ has a unique stationary distribution $\vec{\Pi}^\infty$ given for $\vec{n} \in \mathbb{Z}_+^K$ by*

$$\Pi^\infty(\vec{n}) = \frac{(n_1 + \cdots + n_K)!}{n_1! \cdots n_K!} \pi_1^{n_1} \cdots \pi_K^{n_K} \times \frac{\lambda^{|\vec{n}|}}{\prod_{x=1}^{|\vec{n}|} \mu(x)} \Pi^\infty(\vec{0}) \tag{7.7}$$

where

$$\mu(x) = \sum_{i=1}^K \mu_i (1-(1-\pi_i)^x)$$

and $\Pi^\infty(\vec{0})$ is the normalization constant.

We note that the first term in Eq. (7.7) corresponds to the multinomial distribution where $|\vec{n}|$ jobs are distributed within K servers in batches of n_i jobs, for $i = 1, \ldots, K$. The second multiplying term in $\Pi^\infty(\vec{n})$ corresponds to the distribution of a birth-and-death process with arrival rate λ and departure rate $\mu(|\vec{n}|)$, when the number of users in the system is $|\vec{n}|$.

7.3.2 Convergence of $\vec{N}^\alpha$ towards $\vec{N}^\infty$

We now establish the convergence of $\vec{N}^\alpha$ toward $\vec{N}^\infty$ as $\alpha \to \infty$. Since $\vec{N}^\infty$ is not càdlàg, this convergence cannot hold at the functional level. Rather, we show that the convergence holds in the sense of finite-dimensional distributions, and also for the stationary distributions.

In order to do so, we first prove the following technical result on the tightness of the process with respect to parameter α, defined in Section 2.3.

Proposition 7.3.2. *Let $\theta > 0$ be such that $\lambda(e^\theta - 1) < \frac{\lambda+\bar{\mu}}{2}(1 - e^{-\theta})$ and $\Phi(\vec{n}) = e^{\theta|\vec{n}|}$ for $\vec{n} \in \mathbb{Z}_+^K$. Then Φ is a geometric Lyapounov function of $\vec{N}^\alpha$ uniformly on α. That is, there exist $\eta \in (0,1)$, $\alpha_0 > 0$, $t_0 > 0$ and $n \in \mathbb{Z}_+$ such that for every $\alpha \geq \alpha_0$ and every $\vec{n} \in \mathbb{Z}_+^K$ with $|\vec{n}| \geq n$,*

$$\mathbb{E}_{\vec{n}}\left(e^{\theta|\vec{N}^\alpha(t_0)|}\right) \leq (1-\eta)e^{\theta|\vec{n}|}.$$

In the following proposition we give relevant convergence properties on $\vec{N}^\alpha$ and its stationary distribution, with respect to $\alpha \to \infty$. We provide the relative definitions of convergence in Section 2.3.

Proposition 7.3.3. *As $\alpha \to \infty$, we have:*

- $(\vec{N}^\alpha(t), t \in T) \Rightarrow (\vec{N}^\infty(t), t \in T)$ *for any finite $T \subset \mathbb{R}_+$, i.e., $\vec{N}^\alpha$ converges to $\vec{N}^\infty$ in the sense of finite-dimensional distributions;*
- $|\vec{N}^\alpha| \Rightarrow |\vec{N}^\infty|$, *i.e., $|\vec{N}^\alpha|$ converges to $|\vec{N}^\infty|$ uniformly on compact sets;*
- $\vec{\Pi}^\alpha \Rightarrow \vec{\Pi}^\infty$, *i.e., $\vec{\Pi}^\alpha$ converges in distribution to $\vec{\Pi}^\infty$.*

7.3.3 Comparison of the cases $\alpha = 0$ and $\alpha = \infty$

Since for $\alpha = 0$ the system is a collection of K independent $M/M/1$ queues, we have according to Eq. (7.7)

$$\Delta(\lambda) = \sum_{i=1}^{K} \frac{\mu_i}{\mu_i - \lambda p_i} - \frac{1}{Z} \sum_{n \geq 1} \frac{n\lambda^n}{\prod_{x=1}^n \mu(x)}$$

with $Z = 1 + \sum_{n\geq 1} \frac{\lambda^n}{\prod_{x=1}^n \mu(x)}$

$$\Delta'(\lambda) = \sum_{i=1}^{K} \frac{\mu_i p_i}{(\mu_i - \lambda p_i)^2} - \frac{1}{Z} \sum_{n=1}^{\infty} \frac{n^2\lambda^{n-1}}{\prod_{x=1}^n \mu(x)} + \frac{Z'}{Z^2} \sum_{n=1}^{\infty} \frac{n\lambda^n}{\prod_{x=1}^n \mu(x)}$$

and so $\Delta'(0) = C$ since $Z(0) = 1$ and $\mu(1) = \sum_{i=1}^K \mu_i \pi_i$. We thus obtain the following result.

Proposition 7.3.4. *If $C > 0$ then $\Delta(\lambda) > 0$ for λ small enough, i.e., the system with mobility performs better than the system without. In contrast, if $C < 0$ then $\Delta(\lambda) < 0$ for λ small enough, i.e., the system without mobility performs better than the system with mobility.*

7.4 Numerical analysis

In this section we investigate the performance of the system by numerical means. In order to do so, we solve numerically the balance equations of the system described in Eq. (7.1) and Eq. (7.2).

7.4.1 Mean response time

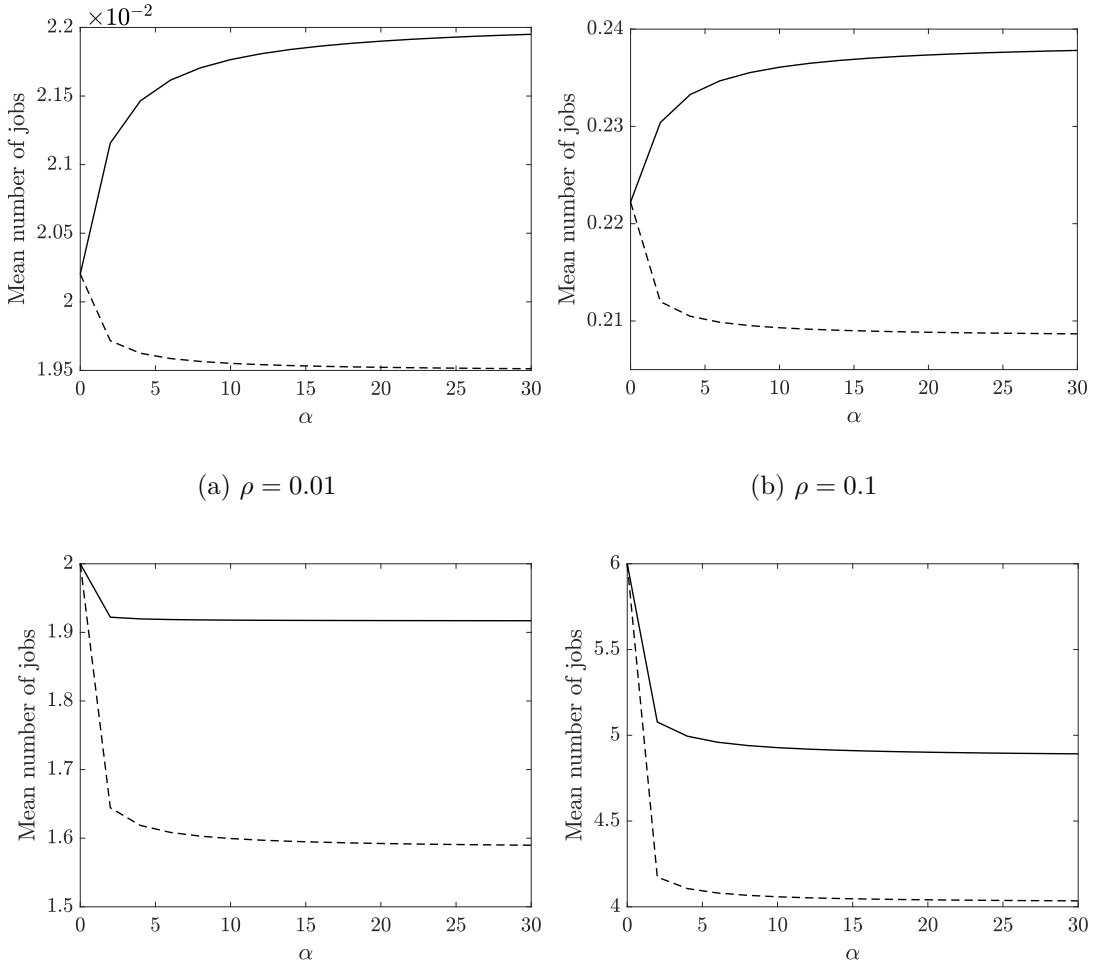
In Figure 7.1 we plot the mean response time with respect to α for different values of loads. We consider $K = 2$ servers, fix parameters $p_1 = 0.6$, $\mu_1 = 1.5$, $\mu_2 = 1$, $r_{12} = 0.5$ and analyze two systems; when $r_{21} = 0.2$ and when $r_{21} = 0.7$.

For the system with $r_{21} = 0.2$, (for which $(\pi_1, \pi_2) = (0.28, 0.71)$), the following inequality holds: $\frac{p_1}{\mu_1\pi_1} > \frac{p_2}{\mu_2\pi_2}$. From Proposition 7.2.1, we know that under sufficiently low loads, the system with no mobility has the best performance. From Figure 7.1 we observe that this remains true until the load is $\rho = 0.25$. We also observe that for $\rho = 0.5, 0.75$ the system with $\alpha = \infty$ has the best performance.

For the system with $r_{21} = 0.7$ (for which $(\pi_1, \pi_2) = (0.58, 0.41)$) the inequality holds in the opposite direction: $\frac{p_1}{\mu_1\pi_1} < \frac{p_2}{\mu_2\pi_2}$. We observe that the system with $\alpha = \infty$ has the best performance for any load.

To assess the performance of the system with $K > 2$, we consider the case $K = 3$ and compute numerically the performance for a large number of parameter settings. The main objective is to determine to what extend the sing of the parameter C permits to predict the monotonicity of the performance. Our analysis consisted in fixing $K = 3$ servers and different parameters p_i and μ_i, for $i = 1, 2, 3$. We then select randomly the values of r_{ij}, and we calculate numerically $\mathbb{E}(|\vec{N}^\alpha|)$ as a function of α. By numerical inspection, we deduce whether $\mathbb{E}(|\vec{N}^\alpha|)$ is monotone or not. Then, among the monotone ones we classify them as increasing or decreasing versus the sing of the value C in that system. The main results we obtain are (i) when the mean number of jobs is monotone, the slope of the function coincides with that fixed by the sign of value C and (ii) the fraction of set of parameters that yield non monotone performance is relatively smaller, see Table 7.1.

To be specific, we explain in detail one of the experiments we considered for the system with $p_1 = p_2 = p_3 = 1/3$ and $\mu_1 = 1$, $\mu_2 = 1.2$ and $\mu_3 = 1.5$. In Table 7.1 we show first the proportion of monotone and non monotone functions. Then, for the



(a) $\rho = 0.01$ (b) $\rho = 0.1$

(c) $\rho = 0.5$ (d) $\rho = 0.75$

Figure 7.1 Mean number of jobs depending on α for $K = 2$ and from left to right an increasing set of load values. For fixed parameters $\mu_1 = 1.5$, $\mu_2 = 1$, $p_1 = 0.6$, $p_2 = 0.4$, $r_{12} = 0.5$. The black filled line corresponds to the system with $r_{21} = 0.2$ and the dashed line to the one with $r_{21} = 0.7$.

system that are monotone on α, the slope of each system versus the sign of the C value. We note that the 90% of the cases the mean number of jobs is monotone and is well classified by its value C. For the remaining 10%, the function is non monotone on α. In Figure 7.2 (a) we plot three examples of the mean number of jobs with respect to α for these particular systems. In the system R_1^α and R_3^α, we note that there is a finite positive α with the minimum number of jobs. In summary, we conclude that even though C fully characterizes the monotonicity for the $K = 2$ case, this is no longer the case for $K > 2$. However, as we saw in Proposition 7.3.4, C does suffice to characterize the sing of Δ, for any value of K.

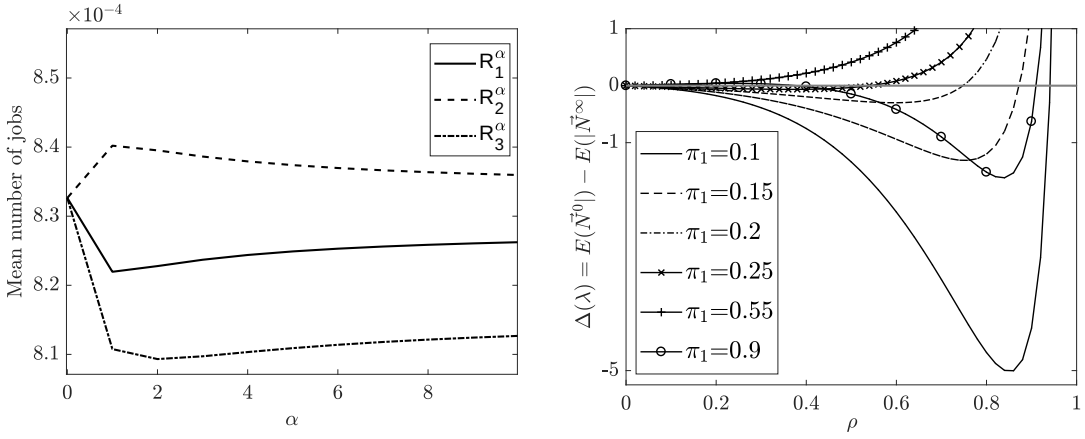
			$C > 0$	$C < 0$
monotone	0.8985	**decreasing**	0.6369	0
		increasing	0	0.2616
non monotone	0.1015			
Total	1			

Table 7.1 Classification of events

7.4.2 Comparison of the cases $\alpha = 0$ and $\alpha = \infty$

In Figure 7.2 (b) we plot function Δ with respect to ρ for $K = 2$ servers and several values of π_1, and π_2. From Eq. (7.5), we obtain that $\Delta < 0$ iff $\pi_1 < 0.5$. We note that mobility has a bad impact for the case $\pi_1 = 0.1$ until loads $\rho < 0.9$. Here π_1 is such that the delay of a job in the system with no mobility reminds smaller than that of the system with mobility $\alpha = \infty$. We also observe that as $\pi_1 \to 0.5$, Δ becomes positive at smaller load values. Additionally, for any π_1, as $\rho \to 1$, Δ is positive. This event can be argued in the following way: as $\rho \to 1$, we expect that $|\vec{N}^{\infty}|$ approaches a single server queue with capacity $\bar{\mu}$, which is more efficient than a K parallel $M/M/1$ system with capacities μ.

Another particular event holds when $\pi_1 = 0.9$. We observe that for intermediate



(a) $K = 3$ servers

(b) $K = 2$ servers

Figure 7.2 (a) Fixed parameters $p_i = 1/3$ for $i = 1, 2, 3$ and $\mu_1 = 1$, $\mu_2 = 1.2$ and $\mu_3 = 1.5$. Three different system with $R_1^{\alpha} = \{r_{12} = 0.71, r_{13} = 0.97, r_{21} = 0.43, r_{23} = 0.07, r_{13} = 0.07, r_{23} = 0.65\}$, $R_2^{\alpha} = \{0.15, 0.97, 0.04, 0.10, 0.16, 0.98\}$ and $R_3^{\alpha} = \{0.04, 0.46, 0.37, 0.07, 0.58, 0.07\}$. (b) For fixed parameters $\mu_1 = 1.5 > \mu_2 = 1$, $p_1 = 0.6$, $p_2 = 0.4$ and $\pi_1, \pi_2 \in (0, 1)$ such that $\pi_1 + \pi_2 = 1$.

values of ρ, the system with no mobility has better performance. This shows that under arbitrary loads, mobility might also have a negative impact on the performance of the system.

7.5 Appendix

A: Proofs of Section 7.2

Proof of Proposition 7.2.1: This result comes from the expression of the derivative of m^{LT} in α, namely

$$\frac{d}{d\alpha} m^{LT}(\alpha) = \frac{(\mu_1 - \mu_2)(p_1\pi_2\mu_2 - p_2\pi_1\mu_1)}{(r_{12} + r_{21}) \left(\alpha(\mu_1\pi_1 + \mu_2\pi_2) + \frac{\mu_1\mu_2}{(r_{12}+r_{21})} \right)^2}$$

□

B: Proofs of Section 7.3

Proof of Proposition 7.3.1: Let X be a random variable distributed according to Eq. (7.7). Then we immediately get

$$\mathbb{P}(|X| = k) = \sum_{\vec{n}:|\vec{n}|=k} \Pi^{\infty}(\vec{n}) \propto \frac{\lambda^k}{\prod_{x=1}^{k} \mu(x)}.$$

According to standard results for birth-and-death process, we recognize the stationary distribution of $|\vec{N}^{\infty}|$. Thus, $|X|$ is the stationary distribution of $|\vec{N}^{\infty}|$. According to Eq. (7.7), conditionally on $|X|$ the coordinates $X_1, \ldots, X_K$ follow the multinomial distribution with parameter $|X|$. Since $\vec{N}^{\infty}(t)$ is obtained similarly from $|\vec{N}^{\infty}|$, this implies that X is the stationary distribution for $\vec{N}^{\infty}$. □

Proof of Proposition 7.3.2: By definition of geometric Lyapunov functions, see Definition 2.2.2, we need to fix mobility speed α^*, initial state $\vec{n}^*$ and parameters t^*, η such that

$$\frac{\mathbb{E}^{\alpha}_{\vec{n}} \left[e^{\left(\theta|\vec{N}^{\alpha}(t^*)|\right)} \right]}{e^{\theta|\vec{N}^{\alpha}(0)|}} < 1 - \eta \tag{7.8}$$

$\forall |\vec{N}^{\alpha}(0)| = |\vec{n}^{\alpha}| \geq |\vec{n}^*|$ and $\alpha \geq \alpha*$. We develop Eq. (7.8) in order to obtain the bound:

$$\begin{aligned}\frac{\mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\left(\theta|\vec{N}^{\alpha}(t)|\right)}\right]}{e^{\theta|\vec{N}^{\alpha}(0)|}} &= \frac{\mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\left(\theta\left(|\vec{N}^{\alpha}(0)|+A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N_i^{\alpha}(u^-)>0)\right)\right)}\right]}{e^{\theta|\vec{N}^{\alpha}(0)|}} \\ &= \mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N_i^{\alpha}(u^-)>0)\right)\right)}\right]\end{aligned} \tag{7.9}$$

where $A(t)$ denotes the total number of arrivals to the system at time t and $D_i(t)$ is the number of potential departures at server i at time t for $i = 1, \ldots, K$.

We denote by $\vec{N}_c^{\alpha}(t)$ the closed system where only the $\vec{N}^{\alpha}(0)$ initial jobs are present and has mobility speed α. Thus, for all $t > 0$, $|\vec{N}^{\alpha}(t) - \vec{N}_c^{\alpha}(t)| \leq A(t) + \sum_{i=1}^{K} D_i(t) = \xi$ and Eq. (7.9) is bounded by the following,

$$\mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N_i^{\alpha}(u^-)>0)\right)\right)}\right] \leq \mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N_{c,i}^{\alpha}(u)>\xi)\right)\right)}\right] \tag{7.10}$$

We divide the value space of ξ in two disjoint subsets: $(\xi \geq \sqrt{n^*})$ and $(\xi < \sqrt{n^*})$. Therefore, Eq. (7.10) equals

$$\begin{aligned}&\mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N_{c,i}^{\alpha}(u)>\xi)\right)\right)}\right] \\ &\quad= \mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N_{c,i}^{\alpha}(u)>\xi)\right)\right)}; \xi \geq \sqrt{n^*}\right] \\ &\quad+ \mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N_{c,i}^{\alpha}(u)>\xi)\right)\right)}; \xi < \sqrt{n^*}\right]\end{aligned} \tag{7.11}$$

In the following we analyze each expressions on its own. For Eq. (7.11), notice that $\xi = A(t) + \sum_{i=1}^{K} D_i(t)$ is distributed by a Poisson process of rate $(\lambda + \bar{\mu})t$.

$$\begin{aligned}&\mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N_{c,i}^{\alpha}(u)>\xi)\right)\right)}; \xi \geq \sqrt{n^*}\right] \\ &\quad\leq \mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\theta A(t)}; \xi = A(t) + \sum_{i=1}^{K} D_i(t) \geq \sqrt{n^*}\right] \\ &\quad\leq \mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\theta\xi}; \xi \geq \sqrt{n^*}\right] = f_{\theta,t}(\sqrt{n^*}) < \frac{\eta}{2}\end{aligned} \tag{7.12}$$

Then, as $\sqrt{n^*} \to \infty$, $f_{\theta,t}(\sqrt{n^*}) \to 0$. Therefore, there exists a constant value, say $\frac{\eta}{2}$ that bounds Eq. (7.11).

On the other hand, turn back attention to Eq. (7.11). Now we develop the second term of the sum,

$$\begin{aligned}
&\mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N^{\alpha}_{c,i}(u)>\xi)\right)\right)}; \xi < \sqrt{n^*}\right] \\
&\quad = \quad \mathbb{E}^{1}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N_{c,i}(\alpha u)>\xi)\right)\right)}; \xi < \sqrt{n^*}\right] \qquad (7.13) \\
&\quad \leq \quad \mathbb{E}^{1}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N_{c,i}(\alpha u)>\sqrt{n^*})\right)\right)}\right] \qquad (7.14)
\end{aligned}$$

For the equality in Eq. (7.13), we rescale the system in the following way: $(N^{\alpha}_c(t), t \geq 0)$ with mobility transition rate matrix αR is equivalent to $(N^1_c(\alpha t), t \geq 0)$ with transition rate matrix R.

Now, we focus on the initial state $\vec{n}$ and fix it as the one with maximum expected value, $\vec{n}_m$. This further implies that the maximum is obtained for some initial state such that $|\vec{n}_m| \geq |\vec{n}^*|$. However, the maximum is obtained at $|\vec{n}_m| = |\vec{n}^*|$ by applying the following argument consecutively. Denoted by $\vec{N}^{\vec{n}_m}_c$ a closed system with $|\vec{n}_m|$ particles that starts at position $\vec{n}^m$. Thus, $1(N^{\vec{n}_m}_{c,i}(t) > \sqrt{n^*}) \leq 1(N^{\vec{n}_m+e_i}_{c,i}(t) > \sqrt{n^*})$ for any t. This happens until the least number of particles given by $|\vec{n}_m| = |\vec{n}^*|$. Therefore, take

$$\vec{n}_m \in \arg \max_{\vec{n}:|\vec{n}_m|=|\vec{n}^*|} \left\{\mathbb{E}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N_{c,i}(\alpha u)>\sqrt{n^*})\right)\right)}\right]\right\}$$

Remark that $\vec{n}_m$ depends on variables α and t, which will be lately fixed. Thus, $\vec{n}_m = \vec{n}_m(\alpha, t)$. Once $\vec{n}_m$ is fixed, we look at closed system $\vec{N}^m(t)$, which starts at position $\vec{n}_m$ and has $|\vec{n}_m| = |\vec{n}^*|$ particles. Therefore, Eq. (7.14) is upper bounded by the followings:

$$\begin{aligned}
&\mathbb{E}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N_{c,i}(\alpha u)>\sqrt{n^*})\right)\right)}\right] \\
&\quad\leq \quad \mathbb{E}_{\vec{n}_m}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N^m_{c,i}(\alpha u)>\sqrt{n^*})\right)\right)}\right] \\
&\quad= \quad e^{\lambda t(e^\theta-1)}\mathbb{E}_{\vec{n}_m}\left[\prod_{i=1}^{K} e^{\left(-\theta\int_0^t D_i(du)1(N^m_{c,i}(\alpha u)>\sqrt{n^*})\right)}\right] \\
&\quad= \quad \mathbb{E}_{\vec{n}_m}\left[e^{\left(\lambda t(e^\theta-1)-\sum_{i=1}^{K}\mu_i(1-e^{-\theta})\int_0^t 1(N^m_{c,i}(\alpha u)>\sqrt{n^*})du\right)}\right]
\end{aligned}$$

By applying the Laplace transform of a Poisson process and then verifying that $1-e^{-\theta 1(N^m_{c,i}(\alpha u)>\sqrt{n^*})} = 1(N^m_{c,i}(\alpha u)>\sqrt{n^*})(1-e^{-\theta})$, equality in Eq. (7.15) holds.

Denote by $Z(\alpha)=\sum_{i=1}^{K}\mu_i\frac{1}{t}\int_0^t 1(N^m_{c,i}(\alpha u)>\sqrt{n^*})\mathrm{d}u$. First, by a change of variables, $Z(\alpha)=\sum_{i=1}^{K}\mu_i\frac{1}{\alpha t}\int_0^{\alpha t} 1(N^m_{c,i}(u)>\sqrt{n^*})\mathrm{d}u$. Take $Z^*=\inf_{\alpha\geq\alpha^*}Z(\alpha)$. Then, Eq. (7.15) is bounded by the following:

$$\mathbb{E}_{\vec{n}_m}\left[e^{\left(\lambda t(e^\theta-1)-\sum_{i=1}^{K}\mu_i(1-e^{-\theta})\int_0^t 1(N^m_{c,i}(\alpha u)>\sqrt{n^*})du\right)}\right] \leq \mathbb{E}_{\vec{n}_m}\left[e^{\left(\lambda t(e^\theta-1)-t(1-e^{-\theta})Z^*\right)}\right] \tag{7.15}$$

From the ergodic theorem, see [87], when $\alpha^*\to\infty$, Z^* converges almost surely into $\sum_{i=1}^{K}\mu_i\mathbb{E}_{\vec{n}_m}(1(N^m_{c,i}(t)>\sqrt{n^*})) = \sum_{i=1}^{K}\mu_i\mathbb{P}_{\vec{n}_m}(N^m_{c,i}(t)>\sqrt{n^*})$ in steady-state. Last distribution expression has the shape of a multinomial distribution. Denote $\sum_{i=1}^{K}\mu_i\mathbb{P}_{\vec{n}_m}(N^m_{c,i}(t)>\sqrt{n^*}) = z^*(n^*,\sqrt{n^*})$. Additionally, $\mathbb{P}_{x^*}(N^m_{c,i}(0)>\sqrt{n^*})\to 1$ as $n^*\to\infty$, for all $i=1,\ldots,K$. Hence, $z^*(n^*,\sqrt{n^*})\to\bar{\mu}$. From the hypothesis: θ is such that $\lambda(e^\theta-1)-\frac{\lambda+\bar{\mu}}{2}(1-e^{-\theta})<0$. Therefore, $e^{\lambda(e^\theta-1)-(1-e^{-\theta})\bar{\mu}} < e^{(1-e^\theta)\frac{\lambda-\bar{\mu}}{2}}$.

$$\begin{aligned}
&\mathbb{E}_{\vec{n}_m}\left[e^{\left(\lambda t(e^\theta-1)-t(1-e^{-\theta})Z^*\right)}\right] \\
&\quad= \mathbb{E}_{\vec{n}_m}\left[e^{\left(\lambda t(e^\theta-1)-t(1-e^{-\theta})(Z^*\pm z^*(n^*,\sqrt{n^*})\right)}\right] \\
&\quad= \quad e^{\lambda t(e^\theta-1)-t(1-e^{-\theta})z^*(n^*,\sqrt{n^*})}\times\mathbb{E}_{\vec{n}_m}\left[e^{\left(-t(1-e^{-\theta})(Z^*-z^*(n^*,\sqrt{n^*}))\right)}\right] \\
&\quad< \quad e^{t(1-e^{-\theta})\frac{\lambda-\bar{\mu}}{2}}\mathbb{E}_{\vec{n}_m}\left[e^{\left(-t(1-e^{-\theta})(Z^*-z^*(n^*,\sqrt{n^*}))\right)}\right]
\end{aligned}$$

Therefore, initial Eq. (7.10) is bounded by Eq. (7.12) and Eq. (7.16):

$$\mathbb{E}^{\alpha}_{\vec{n}}\left[e^{\left(\theta\left(A(t)-\sum_{i=1}^{K}\int_0^t D_i(du)1(N^{\alpha}_{c,i}(u)>\xi)\right)\right)}\right]$$
$$\leq \quad \frac{\eta}{2}+e^{t(1-e^{-\theta})\frac{\lambda-\bar{\mu}}{2}}\mathbb{E}_{\vec{n}_m}\left[e^{\left(-t(1-e^{-\theta})(Z^*-z^*(n^*,\sqrt{n^*})\right)}\right]$$

Fix $t=1$. From stability, $e^{(1-e^{-\theta})\frac{\lambda-\bar{\mu}}{2}}$ is positive and strictly smaller than 1. Say it is bounded by $1-\eta$, for some small $\eta>0$. Lastly, since stated before, $Z^*\to z^*(n^*,\sqrt{n^*})$ when $\alpha^*\to\infty$. Then, we fix α^* for which the previous expectation is bounded by $(1+\frac{\eta}{10})$, Hence, Eq. (7.16) is bounded as follows:

$$\frac{\eta}{2}+e^{(1-e^{-\theta})\frac{\lambda-\bar{\mu}}{2}}\mathbb{E}_{\vec{n}_m}\left[e^{\left(-(1-e^{-\theta})(Z^*-z^*(n^*,\sqrt{n^*})\right)}\right]<\frac{\eta}{2}+(1-\eta)(1+\frac{\eta}{10})<1-\frac{4\eta}{10}$$

Turn attention back to $\vec{n}_m=\vec{n}_m(\alpha,t)$. For this just fixed α^* and for $t=1$, $\vec{n}_m=\vec{n}_m(\alpha^*,1)$ must be fixed and the later steps done again for that particular $\vec{n}_m$.Therefore, we have shown that Φ is a geometric Liapunov function of $\vec{N}^{\alpha}$ uniformly for α. □

Proof of Proposition 7.3.3: We first explain how to derive the last convergence $\vec{\Pi}^{\alpha}\Rightarrow\vec{\Pi}^{\infty}$ from the first item (convergence of finite-dimensional distributions). We then explain the two first convergences, which are based on a coupling argument.

From Proposition 7.3.2 we have that $\Phi(\vec{n})=e^{\theta|\vec{n}|}$ is a geometric Liapunov function. Together with Theorem 2.2.3, we have that

$$\mathbb{E}_{\vec{n}}(e^{\theta|N^{\alpha}(\infty)|})\leq c$$

for any $\alpha\geq\alpha_0$ and for some constant c independent of α. In the following we show that the family of probability distributions $(\vec{\Pi}^{\alpha},\alpha\geq\alpha_0)$ is tight (Definition 2.3.1). By the Markov inequality, we obtain the following inequality:

$$\sup_n P_n(|\vec{X}_n|>C)\leq\sup_n P_n(\phi(|\vec{X}_n|)|>\phi(C))\leq e^{-\phi(C)}\mathbb{E}(e^{|\vec{X}(0)|})\leq e^{-\phi(C)}c,$$

which is 0 as $C\to\infty$.

Let κ be any accumulation point, and assume without loss of generality by working along an appropriate subsequence that $\vec{\Pi}^{\alpha}\Rightarrow\kappa$. Then the finite-dimensional convergence implies that

$$\mathbb{P}_{\vec{\Pi}^{\alpha}}(\vec{N}^{\alpha}(t)=n(t),t\in T)\to\mathbb{P}_{\kappa}(\vec{N}^{\infty}(t)=n(t),t\in T)$$

for any finite subset $T\subset\mathbb{R}^K_+$ and any vector $(n(t),t\in T)$ with $n(t)\in\mathbb{N}^K$ for each $t\in T$. As $\vec{\Pi}^{\alpha}$ is the stationary distribution of $\vec{N}^{\alpha}$, the above convergence implies that κ is

also invariant for $\vec{N}^\infty$ and so $\kappa = \vec{\Pi}^\infty$ according to Proposition 7.3.1. Therefore, from Theorem 2.3.2, it follows that $\vec{\Pi}^\alpha$ converges in distribution to $\vec{\Pi}^\infty$.

Let us now prove the first two convergences. They rely on a coupling argument which we explain. However, we omit the technical details which are cumbersome but do not add significant understanding of the proof. Let $(t_k, k \geq 1)$ be the sequence of arrivals and potential departures from the original system with finite α. Consider another system $\vec{M}$ obtained in the following way: for each $k \geq 1$, $|\vec{M}(t_k+)| = |\vec{N}^\alpha(t_k+)|$ but for $\vec{M}$, these particles are spread according to π in the system. Note that this is not the case for $\vec{N}^\alpha$ because in this system, users are not spread according to π which is the stationary distribution while users only move at finite speed. So, in-between times t_k and t_{k+1}, there are the same number of particles but potentially starting at different positions. Moreover, we couple these trajectories so that if they meet, then they stay merged until time t_{k+1}. As α increases, particles move and therefore also merge faster and faster so that in the limit $\alpha \to \infty$, all particles merge almost instantaneously with probability going to one. In particular, for every $\varepsilon > 0$ we have

$$\mathbb{P}(\vec{N}^\alpha(t) = \vec{M}(t), t \in [0, T] \setminus [t_k, t_k + \varepsilon]) \to 1.$$

In $\vec{M}$, particles are by construction distributed according to π in the network at all times and so the previous relation implies that as $\alpha \to \infty$, this is also the case for $\vec{N}^\alpha$. From this we readily derive the first two convergence results. □

Chapter 8

Conclusions and future work

In this thesis, we studied how both redundancy and mobility impact the performance of computer systems and cellular networks, respectively. This study is motivated by empirical and analytical evidence showing that both redundancy and mobility may improve the perceived service experience by the users. A unifying feature in both problems is that mathematically they can be modeled via multi-server queuing systems.

Under redundancy models, we have first investigated the stability condition when copies are either i.i.d. copies or identical copies. Under the former assumption, we show that under redundancy-d and exponentially distributed service times, the stability condition with both PS and ROS is not reduced when adding redundant copies, as it is the case for FCFS.

However, several studies reveal that the i.i.d. copies assumption, which has been widely assumed in literature, is unrealistic for real-world computer systems. Therefore, we study the performance of system when copies are identical. The main observation is that the stability condition strongly depends on the implemented scheduling policy. We observe that under redundancy-d, the stability condition under ROS is not reduced compared to the non-redundant system. However, the latter is not the case for both FCFS and PS, for which we observe that the stability condition can significantly degrade due to adding redundant copies. Under FCFS the stability condition is derived as the average departure rate of the so-called associated saturated system. Furthermore under PS, we show that the stability condition under any general redundancy topology coincides with that of a system where jobs are only dispatched to its least-loaded compatible servers. For the redundancy-d model, this reduces to the stability condition of a system where all the copies need to be fully served.

In order to have a better understanding of the impact of redundancy, we compare both the stability region and the mean response times, which are the zeroth and first moment of the response time distribution, to non-redundant load balancers such as

Bernoulli routing and JSQ. Our main finding is that when the server capacities are sufficiently heterogeneous, redundancy can considerably improve both performance measures when compared to Bernoulli routing.

We have also investigated how the scheduling policy implemented in the servers impacts the performance of the redundancy system. When jobs have i.i.d. copies, we generalize the result in [41] and show that for a nested model with exponential service times, LRF-Π_2 minimizes the number of jobs in the system for any non-idling policy Π_2. We observe that when the service times of the copies are variable, as it is the case for NWU service times and i.i.d. copies, the performance is improved by the policy that maximizes the number of copies of a job in service, whereas under non-variable service times, or for identical copies, it is improved by the policy that minimizes the number of copies of one job in service. We prove that for a general redundancy topology with i.i.d. copies and NWU service times, Π_1-FCFS minimizes the number of jobs present in the system among all two-level policies of type $\Pi_1 - \Pi_2$, for any first-level strict priority policy Π_1. If instead, copies are identical, we prove that for the nested redundancy model, MRF-Π_2, and particularly MRF-ROS, outperforms MRF-FCFS for any service time distributions, with Π_2 non-idling.

In the case of cellular networks, we investigate how mobility impacts the performance of the system through α, the parameter that controls mobility. The main takeaway message is that mobility might not always have a positive impact on the performance of the system even if it improves its throughput. This result was not evident at first, since there are several recent papers (for instance [17]) where the opposite conclusion has been reached. Our analysis shows that mobility need not always improve the performance, and we have characterized a condition such that, if satisfied, the performance might improve or deteriorate as mobility increases under low loads. Numerical analysis shows that under moderate loads, mobility might have a negative effect on the performance of the system. However, under high loads we observe that mobility does improve the performance and that the system with infinity mobility speed has the minimum delay.

8.1 Open problems on the stability of redundancy models

In the course of this thesis, we have identified several open problems related to stability that might be of interest to the community. We present them in this section, together with several conjectures that arise out of our understanding of the system.

8.1.1 I.i.d. copies

The stability region under FCFS is maximally stable with exponential service times and i.i.d copies, see Gardner et al. [46] and Bonald and Comte [18].

Proposition 8.1.1 ([18, 46])**.** *For the heterogeneous redundancy system with $\mathcal{C}$ job types, exponentially distributed service times and i.i.d. copies, the system under FCFS is stable if for all $C \subseteq \mathcal{C}$,*

$$\lambda \sum_{c \in C} p_c < \sum_{s \in S(C)} \mu_s,$$

where $S(C) = \cup_{c \in C}\{s \in c\}$. The system is unstable if there exists $\tilde{C} \subseteq \mathcal{C}$ such that

$$\lambda \sum_{c \in \tilde{C}} p_c > \sum_{s \in S(\tilde{C})} \mu_s.$$

We believe that the above result should remain valid for any work-conserving scheduling policy with non-preferential treatment across types, such as PS, ROS, Last-Come-First-Served (LCFS) and Least-Attained-Service (LAS). The intuition behind this is the following: the i.i.d assumption combined with the non-preferential treatment across types allows us to take advantage of diversity when the system is close to saturation.

Conjecture 8.1.2. *Consider a redundancy system with a general topology, with exponentially distributed service times and i.i.d. copies. For any work-conserving non-preferential scheduling policy, the system is stable if for all $C \subseteq \mathcal{C}$,*

$$\lambda \sum_{c \in C} p_c < \sum_{s \in S(C)} \mu_s,$$

where $S(C) = \bigcup_{c \in C}\{s \in c\}$. The system is unstable if there exists $\tilde{C} \subseteq \mathcal{C}$ such that

$$\lambda \sum_{c \in \tilde{C}} p_c > \sum_{s \in S(\tilde{C})} \mu_s.$$

Open problem 1: If we relax the exponential service times to general service time distribution, the stability condition is unknown.

8.1.2 FCFS and identical copies

In Section 3.3, we saw that $\lambda/\mu K < \bar{\ell}/K$ is the stability condition of the redundancy-d system with FCFS where jobs have identical copies and exponential service times. In Section 4.3.1 we discuss the stability condition of redundancy systems with a general topology. As we mention there, the state descriptor of the system needs to capture both the type of the jobs in their order of arrival and the attained service of each of the copies in service. The latter implies the techniques used in Section 3.3 are no longer valid and that new techniques are required. Relaxing the assumption that servers have homogeneous capacities makes the analysis of this model hard. Similar multi-server

systems with synchronized departures have been analyzed in literature, however, always for homogeneous server capacities. For instance in [53, 90], the authors derive the stability condition of a multi-server queuing system with homogeneous capacities in which each arriving job requires a random number of servers simultaneously and retains these servers for an exponentially distributed amount of time.

Open problem 2. If we relax the redundancy-d topology to general topologies, or the exponential service times to general service times, the stability condition under FCFS is unknown.

For the redundancy-d model with exponential service times, we observed in Figure 3.2 that for a given number of copies d, $\lim_{K\to\infty} \bar{\ell}/K < 1$. Note that $\lambda/\mu K < 1$ is the stability condition for a system with no redundancy. Hence, if it can be proved that $\lim_{K\to\infty} \bar{\ell}/K < 1$, this would imply that as the number of servers grows large, the traffic load that a redundancy system can support is smaller than if no redundancy was implemented.

Conjecture 8.1.3. *Consider the redundancy-d model where FCFS is implemented and jobs have exponentially distributed service times and identical copies. Then, for fixed d,* $\lim_{K\to\infty} \bar{\ell}/K < 1$.

The limit should coincide with the stability condition given in [59], where the authors develop a numerical method to derive the stability condition in the mean-field limit.

We also observed the following monotonicity property in the number of redundant copies. More precisely, we conjecture that as the degree of redundancy increases, the stability region becomes smaller.

Conjecture 8.1.4. *Consider the redundancy-d model where FCFS is implemented and jobs have exponentially distributed service times and identical copies. Then, for fixed K,* $\bar{\ell}$ *is decreasing in d, and hence, the stability region is decreasing in d.*

8.1.3 ROS and general correlated copies

We conjecture that Proposition 8.1.1 also provides the stability condition of the system where ROS is implemented and copies follow a general correlation structure, including identical copies. The latter is motivated by the following: let us assume that there are multiple copies of the same job being served at various of its compatible servers. Due to the heterogeneous capacities and the correlation among the copies, the departure rate of that job depends on the residual service time of each copy, which makes the analysis hard. We note that if server capacities are homogeneous and copies are identical, the potential departure of a job is characterized by the minimum among the attained service times of the copies in service.

However, when the number of jobs in the system is large, the probability that more than one copy of the same job are simultaneously in service is zero in the fluid limit, as we have shown in Lemma 3.5.1. Then, the fluid limit of the system is that of the system where jobs have i.i.d. copies. Hence, if Conjecture 8.1.2 is valid, this would imply that Conjecture 8.1.5 is true as well.

Conjecture 8.1.5. *Consider a redundancy system with a general topology, with exponentially distributed service times and any correlation structure among the copies. The system where ROS is implemented is stable if for all $C \subseteq \mathcal{C}$,*

$$\lambda \sum_{c \in C} p_c < \sum_{s \in S(C)} \mu_s,$$

where $S(C) = \cup_{c \in C}\{s \in c\}$. The system is unstable if there exists $\tilde{C} \subseteq \mathcal{C}$ such that

$$\lambda \sum_{c \in \tilde{C}} p_c > \sum_{s \in S(\tilde{C})} \mu_s.$$

8.2 Open problems on the impact of the scheduling policy

In Chapter 6 we have investigated how the scheduling policy affects the performance of redundancy models and search for policies that minimize the number of jobs in the system. In this section, we present several conjectures that we believe hold based on our understanding of the system.

8.2.1 I.i.d. copies

In Figure 6.3 we observe that for the W-model and degenerate hyperexponential service times with i.i.d. copies, MRF-FCFS minimizes the mean response time as the variability of the service times increases, that is, when $q \to 0$.

We note that under degenerate hyperexponential distributions and i.i.d. copies, when an idle server schedules a job that is already in service, with probability $1 - q$ this copy has zero service time and with probability q has positive service time. Hence, if q is close to 0, a copy has with a high probability a zero service time, which implies that the job departs immediately. The scheduling policy that maximizes the number of copies of each job in service will maximize this effect and we believe that this can be the optimal policy. Moreover, second-level FCFS maximizes the service time of a job with NWU i.i.d. copies, as it is show in [69]. This gives us the following conjecture.

Conjecture 8.2.1. *Consider the nested redundancy model with heterogeneous servers, where jobs service times are degenerate hyperexponential (with parameter q) and i.i.d. copies. Then, $N^{MRF-FCFS} \leq N^{\pi}$ as $q \to 0$, for any policy π.*

8.2.2 Identical copies

In Figures 6.4 and 6.5, we observe that for the W-model and identical copies, LRF-ROS minimizes the mean response time compared to any other policy. When service times are exponential, we do believe that, maximizing the number of different jobs in service is required in order to optimize the performance. We therefore have the following conjecture.

Conjecture 8.2.2. *Consider the nested redundancy model with heterogeneous server capacities, where jobs have exponential service times and copies are identical. Then,* $\{L^{LRF-ROS}(t)\}_{t\geq 0} \leq \{L^{\pi}(t)\}_{t\geq 0}$*, for any policy* π*.*

8.3 Other open problems

In order to overcome the loss of stability induced by the present redundancy techniques, researchers have proposed other scheduling techniques such as threshold-based redundancy. Under this model, incoming jobs dispatch copies to the servers that have a workload/number of copies of at most Z among its set of compatible servers, or if all compatible servers exceeds workload/number of copies Z the job is send to a compatible server chosen uniformly at random. That is, below the threshold the system behaves as the redundancy model and above the threshold it behaves as the Bernoulli routing model. We note that the stability region is not impacted under this redundancy technique. However, as observed in [57], where the authors obtain the response time distribution in the mean-field limit, the choice of the threshold Z can significantly affect the mean response time of the system.

We believe that it can be interesting to implement the self-learning threshold technique to the present threshold-based redundancy models. Learning has recently been implemented in Goldsztanjn et al. [50] for a parallel-server system where a central dispatcher balances the load according to a threshold-based policy. For a redundancy system, the self-learning threshold-based policy could be implemented in the following way: the dispatcher sends copies to the compatible servers below a threshold value and a single copy to a server uniformly chosen at random above the threshold value. The value of the threshold is adapted at each arrival time by the dispatcher so that the the average number of jobs in the system is minimized. We note that under this new policy, there is no loss of stability and we aim to minimize the mean response time of jobs.

Self References

SR1 Elene Anton, Urtzi Ayesta, Matthieu Jonckheere, and Ina Maria Verloop. On the stability of redundancy models. *Operations Research,* doi: 10.1287/opre.2020.2030, 2021.

SR2 Elene Anton, Urtzi Ayesta, Matthieu Jonckheere, and Ina Maria Verloop. Redundancy with processor sharing servers. *ACM SIGMETRICS Performance Evaluation Review*, 47(2): 15-17. September 2019.

SR3 Elene Anton, Urtzi Ayesta, Matthieu Jonckheere, and Ina Maria Verloop. Improving the performance of heterogeneous data centers through redundancy. *Proceedings of the ACM on Measurement and Analysis of Computing Systems (POMACS)*, SIGMETRICS 2021, 4(3): Article 48, pp. 29, 2020.

SR4 Elene Anton, Urtzi Ayesta, Matthieu Jonckheere, and Ina Maria Verloop. A survey of stability results for redundancy systems. In Alexey B. Piunovskiy and Yi Zhang (eds.), *Modern Trends in Controlled Stochastic Processes: Theory and Applications, V.III.* Springer US, 2021.

SR5 Elene Anton, Urtzi Ayesta, Florian Simatos. On the impact of mobility in cellular networks. *WiOpt 2019. 17th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*, Avignon, France.

SR6 Elene Anton, Rhonda Righter, and Ina Maria Verloop. The impact of the scheduling policy in redundancy systems. *In preparation*, 2021.

SR7 Kunhe Yang, Elene Anton, Mor Harchol-Balter, and Weina Wang. Response time tail bounds for M/M/1 Priority Queues. *In preparation*, 2021.

Bibliography

[1] Ivo Adan and Gideon Weiss. A skill based parallel service system under FCFS-ALIS - steady state, overloads, and abandonments. *Stochastic Systems*, 4(1): 250–299, 2014.

[2] Ivo Adan, Igor Kleiner, Rhonda Righter, and Gideon Weiss. FCFS parallel service systems and matchings models. In *Proceedings of the 11th EAI International Conference on Performance Evaluation Methodologies and Tools*, ValueTools 2017, pp. 106–112, 2017.

[3] Osman Akgun, Rhonda Righter, and Ronald Wolff. Partial flexibility in routing and scheduling. *Advances in Applied Probability*, 45(3): 673–691, 2013.

[4] Ganesh Ananthanarayanan, Ali Ghodsi, Scott Shenker, and Ion Stoica. Why let resources idle? aggressive cloning of jobs with dolly. In *Proceedings of the 4th USENIX Conference on Hot Topics in Cloud Computing*, HotCloud 2012, Article 17, 6 pp. , USA, 2012.

[5] Ganesh Ananthanarayanan, Ali Ghodsi, Scott Shenker, and Ion Stoica. Effective straggler mitigation: Attack of the clones. In *Proceedings of the 10th USENIX Conf. on Networked Systems Design and Implementation* , NSDI 2013, pp. 185–198, 2013.

[6] Ganesh Ananthanarayanan, Srikanth Kandula, Albert G Greenberg, Ion Stoica, Yi Lu, Bikas Saha, and Edward Harris. Reining in the outliers in map-reduce clusters using mantri. In *Proceedings of the 9th USENIX conference on Operating systems design and implementation*, OSDI 2010, pp. 265–278, 2010.

[7] Søren Asmussen. *Applied Probability and Queues*. Springer, 2002.

[8] Rami Atar, Isaac Keslassy, and Gal Mendelson. Replicate to the shortest queues. *Queueing Systems*, 92(1-2): 1–23, 2019.

[9] Urtzi Ayesta, Tejas Bodas, Jan-Pieter Dorsman, and Ina Maria Verloop. A token-based central queue with order-independent service rates. *Operations Research, In press*, 2021.

[10] Urtzi Ayesta, Tejas Bodas, and Ina Maria Verloop. On a unifying product form framework for redundancy models. *Performance Evaluation*, 127-128: 93–119, 2018.

[11] François Baccelli and Serguei Foss. On the saturation rule for the stability of queues. *Journal of Applied Probability*, 32(2):494–507, 1995.

[12] Luiz Andre Barroso, Jeffrey Dean, and Urs Hölzle. Web search for a planet: The google cluster architecture. *IEEE Micro*, 23: 22–28, 2003.

[13] Christian Bettstetter. Mobility modeling in wireless networks: Categorization, smooth movement, and border effects. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(3): 55–66, 2001.

[14] Patrick. Billingsley. *Convergence of Probability Measures*. Wiley Series in Probability and Statistics. Wiley, 2013.

[15] Thomas Bonald, Sem C. Borst, Nidhi Hegde, and Alexandre Proutière. Wireless data performance in multi-cell scenarios. *ACM SIGMETRICS Performance Evaluation Review*, 32: 378–380, 2004.

[16] Thomas Bonald, Sem C. Borst, and Alexandre Proutière. How mobility impacts the flow-level performance of wireless data systems. *IEEE INFOCOM 2004. The 23rd IEEE International Conference on Computer Communications*, Hong Kong, China, 3: 1872 – 1881, 2004.

[17] Thomas Bonald, Sem C. Borst, Nidhi Hegde, Matthieu Jonckheere, and Alexandre Proutière. Flow-level performance and capacity of wireless networks with user mobility. *Queueing Systems*, 63: 131–164, 2009.

[18] Thomas Bonald and Céline Comte. Balanced fair resource sharing in computer clusters. *Performance Evaluation*, 116: 70–83, 2017.

[19] Sem C. Borst, Onno Boxma, Jan Friso Groote, and Sjouke Mauw. Task allocation in a multi-server system. *Journal of Scheduling*, 6(5): 423–436, 2003.

[20] Sem C. Borst. User-level performance of channel-aware scheduling algorithms in wireless data networks. *IEEE INFOCOM 2003. The 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, San Francisco, CA, USA, pp. 321–331, 2003.

[21] Sem C. Borst, Alexandre Proutiére, and Nidhi Hegde. Capacity of wireless data networks with intra- and inter-cell mobility. In *Proceedings of the 25TH IEEE International Conference on Computer Communications*, IEEE INFOCOM 2006, pp. 1–12, 2006.

[22] Sem C. Borst and Matthieu Jonckheere. Flow-level stability of channel-aware scheduling algorithms. *WiOpt 2006. The 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, Boston, MA, USA, pp. 1–6, 2006.

[23] Sem C. Borst. Flow-level performance and user mobility in wireless data networks. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, 366: 2047–2058, 2008.

[24] Sem C. Borst, Nidhi Hegde, and Alexandre Proutière. Mobility-driven scheduling in wireless networks. In *Proceedings of the 28th IEEE International Conference on Computer Communications*, INFOCOM 2009, pp. 1260–1268, 2009.

[25] Sem C. Borst and Florian Simatos. A stochastic network with mobile users in heavy traffic. *Queueing Systems*, 74(1): 1–40, 2013.

[26] Maury Bramson. *Stability of Queueing Networks*. Springer, 2008.

[27] Jake Brutlag. Speed matters for google web search. *Google, Inc*, 2009.

[28] Ellen Cardinaels, Sem C. Borst, and Johan S. H. van Leeuwaarden. Redundancy scheduling with locally stable compatibility graphs. *arXiv:2005.14566*, 2020.

[29] Céline Comte and Jan-Pieter Dorsman. Pass-and-swap queues. *arXiv:2009.12299*, 2020.

[30] James Cruise, Matthieu Jonckheere, and Seva Shneer. Stability of JSQ in queues with general server-job class compatibilities. *Queueing Systems* 95: 271–279, 2020.

[31] Jim G. Dai and Gideon Weiss. Stability and instability of fluid models for reentrant lines. *Mathematics of Operations Research*, 21(1): 115–134, 1996.

[32] Jim G. Dai. On positive harris recurrence of multiclass queueing networks: a unified approach via fluid limit models. *Annals of Applied Probability*, 5: 49–77, 1995.

[33] Jim G. Dai. A fluid limit model criterion for instability of multiclass queueing networks. *The Annals of Applied Probability*, 6: 751–757, 1996.

[34] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: Simplified data processing on large clusters. *OSDI 2004. The 6th Symposium on Operating System Design and Implementation*, San Francisco, CA, pp. 137–150, 2004.

[35] Jeffrey Dean. Achieving rapid response times in large online services. Google Research 2012. *http://research.google.com/people/jeff/latency.html.*

[36] Ken R. Duffy and Seva Shneer. MDS coding is better than replication for job completion times. *arXiv:1907.11052*, 2019.

[37] Regina Egorova. *Sojourn time tails in processor-sharing systems*, Technische Universiteit Eindhoven. PhD thesis, 2009.

[38] Sergey Foss, Dmitry Korshunov, and Stan Zachary. *An introduction to heavy-tailed and subexponential distributions.* Springer Series in Operations Research and Financial Engineering. Springer, 2nd edition, 2013.

[39] David Gamarnik and Assaf Zeevi. Validity of heavy traffic steady-state approximations in generalized jackson networks. *Annals of Applied Probabilities*, 16(1): 56–90, 2006.

[40] Ayalvadi Ganesh, Sarah Lilienthal, D. Manjunath, Alexandre Proutiére, and Florian Simatos. Load balancing via random local search in closed and open systems. *Queueing Systems*, 71: 321–345, 2012.

[41] Kristen Gardner, Mor Harchol-Balter, Esa Hyytiä, and Rhonda Righter. Scheduling for efficiency and fairness in systems with redundancy. *Performance Evaluation*, 116: 1–25, 2017.

[42] Kristen Gardner, Mor Harchol-Balter, Alan Scheller-Wolf, and Benny van Houdt. A better model for job redundancy: Decoupling server slowdown and job size. *IEEE/ACM Transactions on Networking (TON)*, 25(6): 3353–3367, 2017.

[43] Kristen Gardner, Mor Harchol-Balter, Alan Scheller-Wolf, Mark Velednitsky, and Samuel Zbarsky. Redundancy-d: The power of d choices for redundancy. *Operations Research*, 65: 1078–1094, 2017.

[44] Kristen Gardner, Esa Hyytiä, and Rhonda Righter. A little redundancy goes a long way: Convexity in redundancy systems. *Performance Evaluation*, 131: 22–42, 2019.

[45] Kristen Gardner and Rhonda Righter. Product forms for FCFS queueing models with arbitrary server-job compatibilities: an overview. *Queueing Systems*, 96(1): 3–51, 2020.

[46] Kristen Gardner, Samuel Zbarsky, Sherwin Doroudi, Mor Harchol-Balter, Esa Hyytiä, and Alan Scheller-Wolf. Queueing with redundant requests: exact analysis. *Queueing Systems*, 83(3-4): 227–259, 2016.

[47] Nicolas Gast and Bruno Gaujal. Markov chains with discontinuous drifts have differential inclusions limits. *Performance Evaluation*, 69: 623–642, 2012.

[48] Ronald K. Getoor. Transience and recurrence of markov processes. *Séminaire de probabilités de Strasbourg*, 14: 397–409, 1980.

[49] Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung. The google file system. In *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles*, SOSP '03, pp. 29–43, 2003.

[50] Diego Goldsztajn, Sem C. Borst, Johan S. H. van Leeuwaarden, Debankur Mukherjee, and Philip A. Whiting. Self-learning threshold-based load balancing, *arXiv:2010.15525*, 2020.

[51] Wayne Gray and Deborah Boehm-Davis. Milliseconds matter: An introduction to microstrategies and to their use in describing and predicting interactive behavior. *Journal of experimental psychology. Applied*, 6: 322–35, 2001.

[52] H. Christian Gromoll, Philippe Robert, and Bert Zwart. Fluid limits for processor sharing queues with impatience. *Mathematics of Operations Research*, 33: 375–402, 2008.

[53] Isaac Grosof, Mor Harchol-Balter, and Alan Scheller-Wolf Stability for Two-class Multiserver-job Systems, *arXiv:2010.00631*, 2020.

[54] Matthias Grossglauser and David Tse. Mobility increases the capacity of ad-hoc wireless networks. In *Proceedings the 20th Annual Joint Conference of the IEEE Computer and communications Societies Conference on Computer Communications*, IEEE INFOCOM 2001, 3:1360–1369, 2001.

[55] Varun Gupta, Karl Sigman, Mor Harchol-Balter, and Ward Whitt. Insensitivity for PS server farms with JSQ routing. *ACM SIGMETRICS Performance Evaluation Review*, 35: 24–26, 2007.

[56] Mor Harchol-Balter. *Performance Modeling and Design of Computer Systems: Queueing Theory in Action.* Cambridge University Press, 2013.

[57] Tim Hellemans, Tejas Bodas, and Benny van Houdt. Performance analysis of workload dependent load balancing policies. *ACM SIGMETRICS 2019. International Conference on Measurement and Modeling of Computer Systems*, New York, NY, USA, Volume 3(2), Article 35, 35 pp., 2019.

[58] Tim Hellemans and Benny van Houdt. On the power-of-d-choices with least loaded server selection. In *Proceedings of the ACM on Measurement and Analysis of Computing Systems (POMACS)*, SIGMETRICS 2018, 2(2): Article 27, 22 pp., 2018.

[59] Tim Hellemans and Benny van Houdt. Analysis of redundancy(d) with identical replicas. *ACM SIGMETRICS Performance Evaluation Review*, 46(3): 74–79, 2018.

[60] Tim Hellemans and Benny van Houdt. Performance of redundancy(d) with identical/independent replicas. In *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS)*, SIGMETRICS 2019, 4(2): Article 9, 28 pp., 2019.

[61] Tim Hellemans and Benny van Houdt. Heavy traffic analysis of the mean response time for load balancing policies in the mean field regime, *arXiv:2004.00876*, 2020.

[62] Ane Izagirre, Urtzi Ayesta, and Ina Maria Verloop. Sojourn time approximations for a discriminatory processor sharing queue. In *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS)*, SIGMETRICS 2016, 1(1): Article 5, 31 pp., 2016.

[63] Matthieu Jonckheere. Stability of two interfering processors with load balancing. In *Proceedings of the 3rd International Conference on Performance Evaluation Methodologies and Tools,* ValueTools 2008, Article 49, 10 pp., 2008.

[64] Gauri Joshi, Yanpei Liu, and Emina Soljanin. Coding for fast content download. *2012 50th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, USA, pp. 326-333, 2012.

[65] Gauri Joshi, Yanpei Liu, and Emina Soljanin. On the delay-storage trade-off in content download from coded distributed storage systems, *IEEE Journal on Selected Areas in Communications*, volume 32, 2013.

[66] Gauri Joshi, Emina Soljanin, and Gregory Wornell. Efficient redundancy techniques for latency reduction in cloud systems. In *ACM Transactions on Modeling and Performance Evaluation of Computing Systems (TOMPECS)*, SIGMETRICS 2017, 2(2): Article 12, 30 pp., 2017.

[67] Frank P. Kelly. *Stochastic Networks and Reversibility.* Wiley, Chichester, 1979.

[68] Yusik Kim, Rhonda Righter, and Ronald Wolff. Job replication on multiserver systems. *Advances in Applied Probability*, 41: 546–575, 2009.

[69] Ger Koole and Rhonda Righter. Resource allocation in grid computing. *Journal of Scheduling*, 11: 163–173, 2007.

[70] Anthony E. Krzesinski. Order independent queues. In Richard. J. Boucherie and Nicolaas. M. van Dijk, editors, *Queueing Networks: A Fundamental Approach,* Springer 2011, pp. 85–120, 2012.

[71] Kangwook Lee, Ramtin Pedarsani, and Kannan Ramchandran. On scheduling redundant requests with cancellation overheads. *IEEE/ACM Transactions on Networking (TON)*, 25(2): 1279–1290, 2017.

[72] Kangwook Lee, Nihar B. Shah, Longbo Huang, and Kannan Ramchandran. The MDS queue: Analysing the latency performance of erasure codes. *IEEE Transactions on Information Theory*, 63(5): 2822–2842, 2017.

[73] Nam H. Lee. *A sufficient condition for stochastic stability of an Internet congestion control model in terms of fluid model stability,* UC San Diego. PhD thesis, 2008.

[74] Tapani Lehtonen. On the optimality of the shortest line discipline . *Mathematische Operationsforschung und Statistik, Series Optimization,* Taylor & Francis, 15: 291-299, 1984.

[75] Bin Li, Aditya Ramamoorthy, and Rayadurgam Srikant. Mean-field-analysis of coding versus replication in cloud storage systems. *IEEE INFOCOM 2016. The 35th Annual IEEE International Conference on Computer Communications*, San Francisco, CA, USA, pp. 1–9, 2016.

[76] John D.C. Little, Stephen C. Graves. Little's Law. In Dilip Chhajed and Timothy J. Lowe, editors, *Building Intuition: Insights From Basic, Operations Management Models and Principles,* Springer Science + Business Media, LLC, pp. 81–100 2008.

[77] Gal Mendelson. A lower bound on the stability region of redundancy-d with FIFO service discipline. *Operations Research Letters*, 49(1): 113–120, 2021.

[78] Sean P. Meyn and Richard L. Tweedie. Generalized resolvents and harris recurrence of markov processes. *Contemporary Mathematics*, 149: 227–250, 1993.

[79] Sean P. Meyn and Richard L. Tweedie. *Markov chains and stochastic stability.* Springer-Verlag, 1993.

[80] Michael Mitzenmacher. The power of two choices in randomized load balancing. *IEEE Transactions on Parallel and Distributed Systems*, 12(10): 1094–1104, 2001.

[81] Fernando Paganini, Ao Tang, Andrés Ferragut, and Lachlan Andrew. Network stability under alpha fair bandwidth allocation with general file size distribution. *IEEE Transactions on Automatic Control*, 57(3): 579–591, 2012.

[82] Youri Raaijmakers, Sem C. Borst, and Onno Boxma. Redundancy scheduling with scaled bernoulli service requirements. *Queueing Systems*, 93: 67–82, 2019.

[83] Youri Raaijmakers, Sem C. Borst, and Onno Boxma. Stability of redundancy systems with processor sharing. *Performance Evaluation*, 147, Article 102195, pp. 19, 2021.

[84] Youri Raaijmakers and Sem C. Borst. Achievable stability in redundancy systems. In *Proceedings of the ACM on Measurement and Analysis of Computing Systems (POMACS)*, SIGMETRICS 2020, 4(3), Article 46, 21 pp., 2020.

[85] Martin I. Reiman and Burton Simon. An interpolation approximation for queueing systems with poisson input. *Operations Research*, 36: 454–469, 1988.

[86] Martin I. Reiman and Burton Simon. Light traffic limits of sojourn time distributions in markovian queueing networks. *Stochastic Models*, 4:191–233, 1988.

[87] Philippe Robert. *Stochastic Networks and Queues*. Springer-Verlag, 2003.

[88] Sheldon M. Ross. *Stochastic Processes*. Wiley series in probability and mathematical statistics. Wiley, 1995.

[89] Ron Roth. *Introduction to Coding Theory*. Cambridge University Press, 2006.

[90] Alexander Rumyantsev and Evsey Morozov. Stability criterion of a multiserver model with simultaneous service. *Annals of Operations Research*, 252: 29-–39, 2017.

[91] Sandvine. The mobile Internet phenomena report 2020. *https://www.sandvine.com*. February 2020.

[92] Sandvine. The global Internet phenomena report Covid-19 spotlight. *https://www.sandvine.com*. May 2020.

[93] Christian Schindelhauer. Mobility in wireless networks. In Jiri Wiedermann, Gerard Tel, Jaroslav Pokorny and Maria BielikovaJulius Stuller, editors, *SOFSEM 2006: Theory and Practice of Computer Science*, 3831: 100–116, 2006.

[94] Moshe Shaked and J. George Shantikumar. *Stochastic Orders*, volume 63. Springer edition, 2007.

[95] Christian Sieber, Andreas Blenk, Max Hinteregger, and Wolfgang Kellerer. The cost of aggressive http adaptive streaming: Quantifying youtube's redundant traffic. In *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pp. 1261–1267, 2015.

[96] Christian Sieber, Poul Heegaard, Tobias Hossfeld, and Wolfgang Kellerer. Sacrificing efficiency for quality of experience: Youtube's redundant traffic behavior. *2016 IFIP Networking Conference*, Vienna, Austria, pp. 503-511, 2016.

[97] Florian Simatos and Danielle Tibi. Spatial homogenization in a stochastic network with mobility. *Annals of Applied Probabilities*, 20(1): 312–355, 2010.

[98] Steve Souders. Velocity and the bottom line. *http://radar.oreilly.com/2009/07/velocity-making-your-site-fast.html*, 2009.

[99] StatCounter. Search engine market share worldwide. *https://gs.statcounter.com/search-engine-market-share*,2020.

[100] Yin Sun, C. Emre Koksal, and Ness B. Shroff. On delay-optimal scheduling in queueing systems with replications, *arXiv:1603.07322*, 2016.

[101] Brian S. Thomson, Judith B. Bruckner, Andrew M. Bruckner Elementary Real Analysis. *Prentice-Hall*, 2001.

[102] Jeremy Visschers, Ivo Adan, and Gideon Weiss. A product form solution to a system with multi-type jobs and multi-type servers. *Queueing Systems*, 70: 269–298, 2012.

[103] Ashish Vulimiri, Oliver Michel, P Brighten Godfrey, and Scott Shenker. More is less: Reducing latency via redundancy. In *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, HotNets 2011, 11: 13–18, 2012.

[104] Ashish Vulimiri, Philip Brighten Godfrey, Radhika Mittal, Justine Sherry, Sylvia Ratnasamy, and Scott Shenker. Low latency via redundancy. In *Proceedings of the ACM conference on Emerging networking experiments and technologies*, CoNEXT 2013, pp. 283–294, 2013.

[105] Jean Walrand. Chapter 11 queueing networks. In D.P. Heyman and M.J. Sobel, editors, *Stochastic Models*, volume 2 of *Handbooks in Operations Research and Management Science, Elsevier*, 2: 519 – 603, 1990.

[106] Matei Zaharia, Andy Konwinski, Anthony D. Joseph, Randy Katz, and Ion Stoica. Improving mapreduce performance in heterogeneous environments. In *Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation*, OSDI 2008, pp. 29–42, 2008.

[107] Martin Zubeldia. Delay-optimal policies in partial fork-join systems with redundancy and random slowdowns. In *Proceedings of the ACM on Measurement and Analysis of Computing Systems (POMACS)*, SIGMETRICS 2020, 4(1): Article 2, 49 pp. 2020.

List of figures

List of tables

Abstract

In this thesis, we studied how both redundancy and mobility impact the performance of computer systems and cellular networks, respectively. The general notion of redundancy is that upon arrival each job dispatches copies into multiple servers. This allows exploiting the variability of the queue lengths and server capacities in the system. We consider redundancy models with both identical and i.i.d. copies. When copies are i.i.d., we show that with PS and ROS, redundancy does not reduce the stability region. When copies are identical, we characterize the stability condition for systems where either FCFS, PS, or ROS is implemented in the servers. We observe that this condition strongly depends on the scheduling policy implemented in the system. We then investigate how redundancy impacts the performance by comparing it to a non-redundant system. We observe that both the stability and performance improve considerably under redundancy as the heterogeneity of the server capacities increases. Furthermore, for both i.i.d. and identical copies, we characterize redundancy-aware scheduling policies that improve both the stability and performance. Finally, we identify several open problems that might be of interest to the community.

User mobility in wireless networks addresses the fact that users in a cellular network switch from cell to cell when geographically moving in the system. We control the mobility speed of the users among the servers and analyze how mobility impacts the performance at a user level. We observe that the performance of the system under fixed mobility speed strongly depends on the inherent parameters of the system.

Keywords: Load balancing, stability, performance, redundancy, mobility.

Résumé

Dans cette thèse, nous avons étudié l'impact de la redondance et de la mobilité sur les performances des systèmes informatiques et des réseaux cellulaires, respectivement. La notion générale de redondance est qu'à l'arrivée, chaque tâche envoie des copies dans plusieurs serveurs. Cela permet d'exploiter la variabilité de la longueur des files d'attente et des capacités du serveur dans le système. Nous considérons des modèles de redondance où les tâches ont soit des copies i.i.d. ou des copies identiques. Lorsque des copies sont i.i.d., nous montrons que la région de stabilité n'est pas réduit quand PS ou ROS est mis en œuvre. Lorsque les copies sont identiques, nous caractérisons la condition de stabilité pour les systèmes où FCFS, PS ou ROS est mis en œuvre dans les serveurs. Nous observons que cette condition dépend fortement de la discipline de service. Nous examinons ensuite l'incidence de la redondance sur le rendement en la comparant à celle d'un système où il n'y a pas de redondance. Nous observons que la stabilité et les performances sont considérablement améliorées sous l'effet de la redondance, à mesure que l'hétérogénéité des capacités du serveur augmente. De plus, pour les systèmes avec des copies i.i.d. et des copies identiques, nous caractérisons des disciplines de service prenant en compte la redondance qui peuvent améliorer à la fois la stabilité et les performances du système. Enfin, nous identifions plusieurs problèmes ouverts qui pourraient intéresser la collectivité.

La mobilité des utilisateurs dans les réseaux sans fil rend compte du fait que les utilisateurs d'un réseau cellulaire passent d'une cellule à l'autre lorsqu'ils se déplacent géographiquement dans le système. Nous contrôlons la vitesse de mobilité des utilisateurs parmi les serveurs et analysons comment la mobilité affecte les performances au niveau de l'utilisateur. Nous observons que la performance du système à vitesse de mobilité constante dépend fortement des paramètres inhérents au système.

Mots clés: Équilibrage de charge, stabilité, performance, redondance, mobilité.