Modelling complex stochastic systems

Approaches to management and stability

Patch, B.J.

Publication date
2019

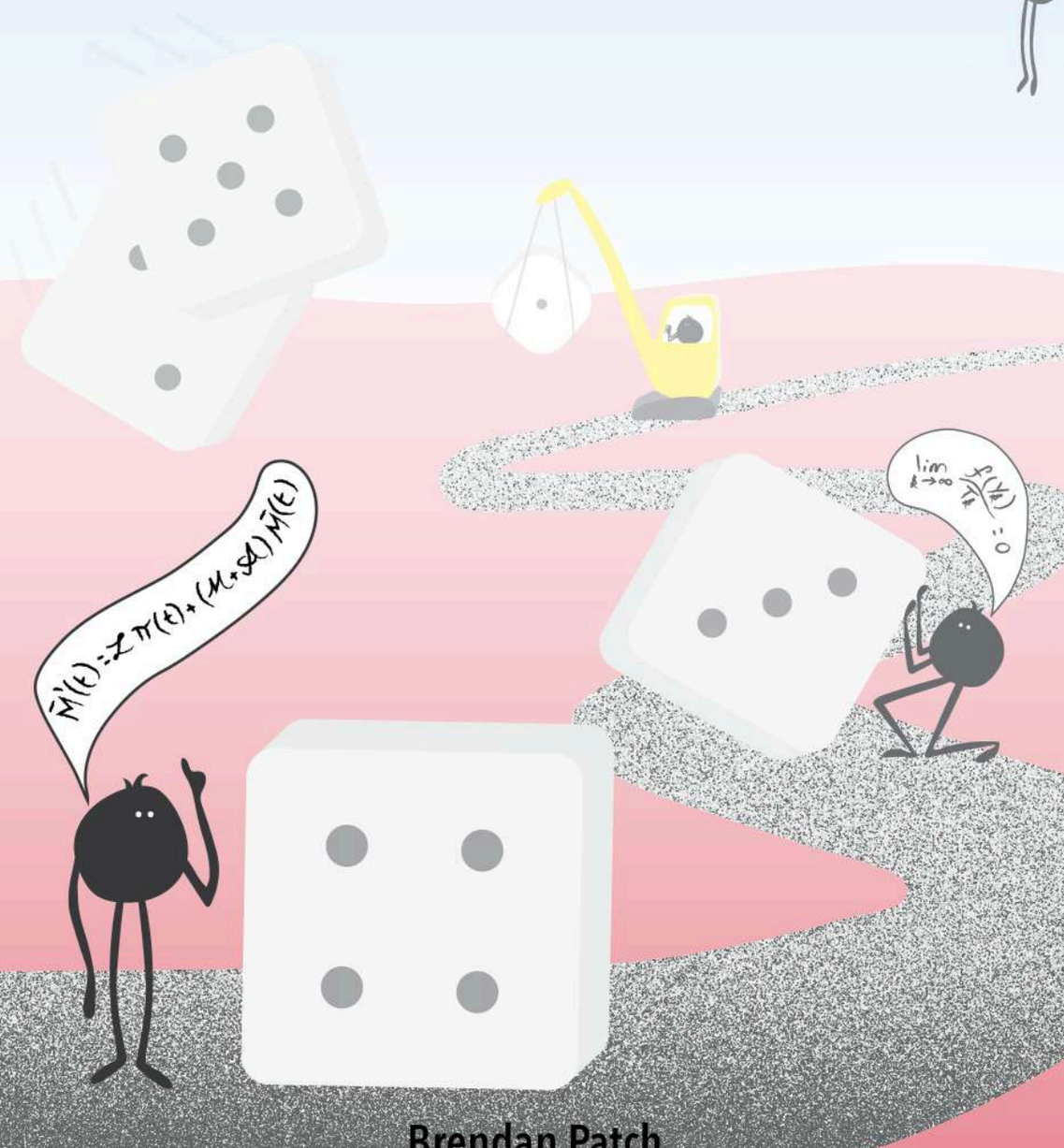
Document Version
Final published version

License
Other

Modelling complex stochastic systems: approaches to management and stability

Brendan Patch

Modelling complex stochastic systems: approaches to management and stability



Brendan Patch

Modelling complex stochastic systems: approaches to management and stability

Brendan Patch

This thesis was prepared within the partnership between the University of Amsterdam and The University of Queensland with the purpose of obtaining a joint doctorate degree. The thesis was prepared in the Faculty of Science at the University of Amsterdam and in the School of Mathematics and Physics at The University of Queensland.

Dit proefschrift is tot stand gekomen binnen een samenwerkingsverband tussen de Universiteit van Amsterdam en The University of Queensland met als doel het behalen van een gezamenlijk doctoraat. Het proefschift is voorbereid in de Faculteit der Natuurwetenschappen, Wiskunde en Informatica van de Universiteit van Amsterdam en de School of Mathematics and Physics van The University of Queensland.

THE UNIVERSITY OF QUEENSLAND

AUSTRALIA

Modelling complex stochastic systems: approaches to management and stability

Brendan Patch
BEc, BFin, MSc

A thesis submitted for the degree of Doctor of Philosophy at
The University of Queensland
School of Mathematics and Physics
in collaboration with the
University of Amsterdam
Korteweg-de Vries Institute for Mathematics
in 2018

Modelling complex stochastic systems: approaches to management and stability

Academisch Proefschrift

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. K.I.J. Maex
ten overstaan van een door het College voor Promoties ingestelde
commissie, in het openbaar te verdedigen in
de UQ St Lucia Campus, Brisbane, Australië
op 11 februari 2019, te 17:00 uur

door

Brendan John Patch

geboren te Canberra

Promotiecommissie

Promotores:

prof. dr. M.R.H. Mandjes
Universiteit van Amsterdam

dr. T. Taimre
The University of Queensland

Copromotor:

dr. N.S. Walton
The University of Manchester

Overige leden:

prof. dr. R. Núñez-Queija
Universiteit van Amsterdam

prof. dr. P.K. Pollett
The University of Queensland

prof. dr. P.J.C. Spreij
Universiteit van Amsterdam

prof. dr. P. Taylor
The University of Melbourne

dr. A.V. den Boer
Universiteit van Amsterdam

dr. I.B. Ziedins
The University of Auckland

prof. dr. J.H. van Zanten
Universiteit van Amsterdam

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

Abstract

This thesis is about coping with variability in outcomes for complex stochastic systems. We focus on systems where jobs arrive randomly throughout time to utilise resources for a random amount of time before departure. The systems we investigate are primarily concerned with the communication and storage of data. The thesis is partitioned into two parts. The first part studies systems where congestion leads to jobs waiting for service (queueing systems) and the second part considers systems where congestion leads to losses due to departures before provision of service (loss systems).

For queueing systems, we are mainly interested in the management objective of ensuring that the expected time a job must wait before entering is finite — a property known as stability. Finite waiting times occur naturally for loss systems due to the balking behaviour of jobs in response to congestion and so our attention in this case turns to the more ambitious goal of managing systems in such a way that the number of lost jobs is minimised.

Each part consists of an introductory chapter providing background knowledge, which is followed by three chapters containing original research. In both parts we progress through these chapters by first applying traditional analytical approaches to novel models and then developing novel simulation-based approaches for models which are out of reach of traditional approaches.

We begin our research on queueing networks in Part 1 by considering a network of infinite-server queues with the special feature that, triggered by specific events, the network population vector may undergo a linear transformation. We use moment generating functions to obtain expressions for transient and stationary moments of the queue size vector and characterise the set of parameters for which the system is stable. A variety of systems fit in the framework developed, such as networks of retrial queues, networks in which jobs can be rerouted when links fail, and storage systems.

In the next chapter of Part 1 we study the recently introduced Queue-Proportional Rate Allocation scheduling algorithm for multihop radio networks. The main contribution is a proof using fluid limit techniques to show that a natural generalisation of this policy to allow weighting of packets at each link, to reflect nonhomogeneous priorities, retains the maximal stability property. We also state a conjecture that in heavy traffic the diffusion-scaled workload process of the network converges weakly to a reflected Brownian motion and that in this weak limit the vector of queue lengths is always proportional to the traffic arrival rate vector.

We conclude Part 1 by devising a simulation-based method for detecting whether a non-negative Markov chain is unstable for a given set of parameter values. More precisely, for a given subset of the parameter space, we develop an algorithm that can decide whether the set has a subset of positive Lebesgue measure for which the Markov chain is unstable. The approach is based on a variant of simulated annealing, and consequently only mild assumptions are needed to obtain rigorous performance guarantees. Our framework leads to a procedure that can perform statistically rigorous tests for instability, which has been extensively tested using several examples of standard and non-standard queueing networks.

We begin our investigation of loss systems in Part 2 by considering a finite-capacity Erlang B model that alternates between active and inactive states according to a two-state modulating Markov process. Jobs arrives to the system as a Poisson process but are blocked from entry when the system is at capacity or inactive. We use Laplace transforms

to derive expressions for the revenue lost during short term planning horizons. These expressions can be used to assess alternative system designs.

In the next chapter of Part 2 we develop a sophisticated loss system type model for cloud computing systems. User demand on the computational resources of cloud computing platforms varies over time. These variations in the arrival process can be predictable or unpredictable, resulting in time-varying and 'bursty' demand fluctuations. Furthermore, jobs can arrive in batches, and users whose demands are not met can be impatient. We demonstrate how to compute the expected revenue loss over a finite time horizon in the presence of all these model characteristics using matrix analytic methods. It is seen that taking these characteristics of fluctuating user demand into account can result in a substantial reduction of losses.

We conclude Part 2 by developing an optimisation framework for a model applicable to mobile cloud edge computing systems. Our model is a stochastic network with blocking: jobs attempt to be processed sequentially at nodes in a network but are lost when they attempt to access a node that is at capacity. The problem is mathematically intractable in general and time consuming to solve using standard simulation methods. Our novel method combines simulation with analytical approximations to quickly obtain high quality solutions. We extensively test our approach using several complex models.

Abstract (Dutch)

Modellering van complexe stochastische systemen: aanpak voor management en stabiliteit

Dit proefschrift gaat over het omgaan met variabiliteit in complexe stochastische systemen. We richten ons op systemen waarbij taken op een willekeurig moment in de tijd arriveren om voor een willekeurige tijdsduur gebruik te maken van diensten alvorens weer te vertrekken. De systemen die we onderzoeken houden zich primair bezig met het doorsturen en opslaan van gegevens. Het proefschrift is verdeeld in twee delen. In het eerste deel worden systemen bestudeerd waarbij congestie ertoe leidt dat taken moeten wachten op service (wachtrijsystemen) en het tweede deel gaat over systemen waarbij congestie leidt tot het verliezen van taken voordat deze voorzien kunnen worden van service (zogenaamde verliessystemen).

Voor wachtrijsystemen zijn we vooral geïnteresseerd in de managementdoelstelling ervoor te zorgen dat de verwachte tijd die een taak moet wachten voordat hij in behandeling wordt genomen eindig is — een eigenschap die stabiliteit wordt genoemd. Eindige wachttijden komen van nature voor in verliessystemen doordat taken het systeem vroegtijdig verlaten als reactie op congestie. Daarom richten we onze aandacht in dit geval op het ambitieuzere doel om systemen zodanig te beheren dat het aantal *verloren* taken tot een minimum wordt beperkt.

Beide delen van het proefschrift bestaan uit een inleidend hoofdstuk met achtergrondkennis, gevolgd door drie hoofdstukken met nieuw onderzoek. In beide delen gaan we door deze hoofdstukken heen door eerst traditionele analytische methoden toe te passen op nieuwe modellen, om vervolgens nieuwe simulatie-gebaseerde methoden te ontwikkelen voor modellen buiten het bereik van de traditionele methoden liggen.

We beginnen ons onderzoek naar wachtrijsystemen in Deel 1 met het bekijken van een netwerk van wachtrijsystemen met oneindig veel servers met de speciale eigenschap dat door bepaalde gebeurtenissen de vector die de netwerk populatie beschrijft een lineaire transformatie kan ondergaan. We gebruiken momentgenererende functies om uitdrukkingen voor tijdsafhankelijke en stationaire momenten van de rijlengte vector te verkrijgen en de verzameling parameters te karakteriseren waarvoor het systeem stabiel is. Een verscheidenheid aan systemen valt onder het daarmee ontwikkelde framework, zoals opslagsystemen, netwerken van *retrial* wachtrijen en netwerken waarin taken kunnen worden omgeleid als koppelingen falen.

In het volgende hoofdstuk van Deel 1 bestuderen we het recent geïntroduceerde *wachtrijproportionele rate allocatie* algoritme voor multihop radionetwerken. De belangrijkste bijdrage is een bewijs dat gebruik maakt van *fluïd* limieten om aan te tonen dat bij een natuurlijke generalisatie van dit beleid, namelijk om taken bij elke koppeling een weging mee te geven om zo rekening te houden met inhomogene prioriteit, de maximale stabiliteitseigenschap behouden blijft. We presenteren ook het vermoeden dat in geval van *heavy traffic* het diffusie-geschaalde werkbelastingsproces van het netwerk zwak convergeert naar een gereflecteerde Brownse beweging en dat in deze zwakke limiet de rijlengte vector altijd proportioneel is aan de aankomstintensiteit vector.

We sluiten Deel 1 af door een simulatie-gebaseerde methode te ontwikkelen voor het vaststellen of voor een bepaalde verzameling parameters een niet-negatieve Markovketen onstabiel is. Om precies te zijn, we ontwikkelen voor een bepaalde deelverzameling van de parameterruimte een algoritme dat kan bepalen of deze een deelverzameling met positieve lebesgue-maat bevat waarvoor de Markovketen onstabiel is. De aanpak is gebaseerd

op een variant van *simulated annealing*, en daarom zijn slechts milde aannames nodig voor rigoureuze prestatiegaranties. Ons raamwerk leidt tot een procedure die statistisch rigoureuze tests voor instabiliteit kan uitvoeren, die uitgebreid is getest op verschillende standaard en niet-standaard wachtrijnetwerken.

We beginnen ons onderzoek naar verliessystemen in Deel 2 door een Erlang B model met eindige capaciteit te beschouwen die alterneert tussen actieve en non-actieve toestand volgens een *2-state modulating Markov proces*. Taken arriveren in het systeem volgens een Poisson proces, maar worden geblokkeerd bij aankomst wanneer het systeem vol of inactief is. We gebruiken laplacetransformaties om uitdrukkingen af te leiden voor het gemaakte verlies tijdens een korte termijn planningshorizon. Deze uitdrukkingen kunnen worden gebruikt om alternatieve systemen te beoordelen.

In het volgende hoofdstuk van Deel 2 ontwikkelen we een geavanceerd model van het type verliessysteem voor *cloud computing* systemen. De vraag van gebruikers naar de computationele bronnen van cloud computing platforms varieert over de tijd. Deze variabiliteit in het aankomstproces kan voorspelbaar of onvoorspelbaar zijn, waarbij dat laatste resulteert in tijdsafhankelijke en 'bursty' patronen in de vraag. Bovendien kunnen taken in *batches* aankomen en gebruikers aan wiens vraag niet (tijdig) wordt voldaan, kunnen ongeduldig raken. We laten zien hoe het verwachte verlies over een eindige tijdshorizon kan worden berekend met behulp van matrixanalysemethoden, waarbij met al deze modelkenmerken rekening wordt gehouden. We merken op dat dit kan resulteren in een substantiële vermindering van verlies.

We sluiten Deel 2 af door een optimalisatie kader te ontwikkelen voor een model dat van toepassing is op mobiele *cloud edge computing systemen*. Ons model is een stochastisch netwerk met blokkering: taken proberen achtereenvolgens te worden verwerkt op knooppunten in een netwerk, maar gaan verloren wanneer ze toegang proberen te krijgen tot een vol knooppunt. Het probleem is in het algemeen niet wiskundig op te lossen en het oplossen met behulp van standaard simulatie technieken is erg tijdrovend. Onze nieuwe methode combineert simulatie met analytische methoden om snel oplossingen van hoge kwaliteit te verkrijgen. We testen onze aanpak uitgebreid met behulp van verschillende complexe modellen.

Declaration by author

This thesis is composed of my original work, and contains no material previously published or written by another person except where due reference has been made in the text. I have clearly stated the contribution by others to jointly-authored works that I have included in my thesis.

I have clearly stated the contribution of others to my thesis as a whole, including statistical assistance, survey design, data analysis, significant technical procedures, professional editorial advice, financial support and any other original research work used or reported in my thesis. The content of my thesis is the result of work I have carried out since the commencement of my higher degree by research candidature and does not include a substantial part of work that has been submitted to qualify for the award of any other degree or diploma in any university or other tertiary institution. I have clearly stated which parts of my thesis, if any, have been submitted to qualify for another award.

I acknowledge that an electronic copy of my thesis must be lodged with the University Library and, subject to the policy and procedures of The University of Queensland, the thesis be made available for research and study in accordance with the Copyright Act 1968 unless a period of embargo has been approved by the Dean of the Graduate School.

I acknowledge that copyright of all material contained in my thesis resides with the copyright holder(s) of that material. Where appropriate I have obtained copyright permission from the copyright holder to reproduce material in this thesis and have sought permission from co-authors for any jointly authored works included in the thesis.

Publications included in this thesis

Journal articles (peer-reviewed)

1. M. Mandjes, **B. Patch**, and N. S. Walton. Detecting Markov chain instability: A Monte Carlo approach, *Stochastic Systems*, 7.2 (2017). pp. 48–62.

2. **B. Patch** and T. Taimre. Transient provisioning and performance evaluation for cloud computing platforms: A capacity value approach, *Performance Evaluation*, 118 (2018). pp. 289–314.

3. D. Fiems, M. Mandjes, and **B. Patch**. Networks of infinite-server queues with multiplicative transitions, *Performance Evaluation*, 123–124 (2018). pp. 35–49.

Conference proceedings (peer-reviewed)

4. **B. Patch**, T. Taimre, and Y. Nazarathy, Performance of faulty loss systems with persistent connections, *ACM SIGMETRICS Performance Evaluation Review*, 43.2 (2015). pp 16–18.

Submitted manuscripts included in this thesis

No manuscripts submitted for publication.

Other publications during candidature

Journal articles (peer-reviewed)

5. **B. Patch**, Y. Nazarathy, and T. Taimre, A correction term for the asymptotic covariance of renewal-reward processes with multivariate rewards, *Statistics and Probability Letters*, 102 (2015). pp. 1–7.

6. W. Merritt, **B. Patch**, V. R. Reddy, and G. J. Syme, Modelling livelihoods and household resilience to droughts using Bayesian networks, *Environment, Development and Sustainability*, 18.2 (2016). pp. 315–346.

Conference proceedings (peer-reviewed)

7. M. Borm, **B. Patch**, T. Taimre, and I. Adan, Evaluation of a self-organized traffic light policy, *Proceedings of the 9th EAI International Conference on Performance Evaluation Methodologies and Tools*, (2016). pp 135–136.

8. Y. Nazarathy, T. Taimre, A. Asanjarani, J. Kuhn, **B. Patch**, and A. Vuorinen, The challenge of stabilizing control for queueing systems with unobservable server states. *5th Australian Control Conference (AUCC)*, Gold Coast, Australia, (2016). pp. 342–347.

Contributions by others to the thesis

In addition to the contributions by others outlined immediately preceding Chapter 2, Chapter 4, Chapter 6, and Chapter 7:

- Chapter 3 is based on ongoing work by the candidate and Neil Walton (University of Manchester); each of these authors have contributed equally to this work.
- Chapter 8 is based on ongoing work by the candidate, Mark Squillante (Mathematical Sciences Department, IBM), and Peter van de Ven (Center for Mathematics and Computer Science (CWI)); each of these authors have contributed equally to this work.

Statement of parts of the thesis submitted to qualify for the award of another degree

No works submitted towards another degree have been included in this thesis.

Research involving human or animal subjects

No animal or human subjects were involved in this research.

Acknowledgments

I would like to thank my supervisors Michel Mandjes (University of Amsterdam (UvA)), Thomas Taimre (University of Queensland (UQ)) and Neil Walton (University of Manchester) for their support throughout my candidature. Their expert guidance has been invaluable throughout the last four years and I have learnt a lot from them. More importantly, due to these three I have enjoyed my academic experience in such a way that I have a strong desire to continue with research.

I would also like to thank the co-authors of the chapters of this thesis, for their indispensable input, guidance and just for generally being a pleasure to work with. Thanks Dieter Fiems (University of Ghent), Yoni Nazarathy (UQ), Mark Squillante (IBM Research) and Peter van de Ven (CWI - Dutch Center for Mathematics and Computer Science).

In addition, I also had many useful discussions about Chapter 4 and Chapter 7 with Yoni Nazarathy, who inspired these directions of research. Peter Taylor (University of Melbourne) provided useful remarks on Chapter 2, Chapter 6 and Chapter 7. Ross McVinish (UQ) provided useful remarks on Chapter 2.

I thank my Doctorate Committee on behalf of University of Amsterdam, consisting of Sindo Nunez Queija (UvA), Phil Pollett (UQ), Peter Spreij (UvA), Peter Taylor, Arnoud den Boer (UvA), Ilze Ziedins (University of Auckland) and Harry van Zanten (UvA) for feedback on earlier versions of this thesis and sacrificing their time towards my examination.

Rosina Muir and Tor Lattimore gave excellent comments on earlier versions of this thesis. Thanks!

All of the chapters which are published in journals have undergone careful reading from anonymous referees, and the feedback from this process has substantially improved the quality of this thesis. I am grateful to all of the referees and editors associated with this process.

This thesis was written in Amsterdam, Brisbane, New York, Manchester, Basel, Zurich, Lausanne and Melbourne, among other places. I would like to thank my hosts and their families in these places. Particularly Neil's wife Aurora Cruz-Cabeza for helping Neil to host me in Basel and Sophie Hautphenne for hosting me in Lausanne.

The logistics of organising a joint doctorate are immense and have been ongoing for the entirety of this four year process. There are many little differences not only between the PhD programs of UQ and UvA, but also in terms of administrative matters generally in Australia and The Netherlands, details which I am now far too familiar with. A big thank you to everyone involved in this process, I feel like we got most of it done right and for the things that were done wrong — we are all only human.

Last but not least I wish to thank my family and friends for their love, encouragement, and ability to distract me from mathematics over the years. You know who you are.

I am particularly grateful to Sarah Hunt in many ways.

Financial support

The research of the candidate was financially supported by an Australian Government Research Training Program scholarship, an ARC Centre of Excellence for Mathematical and Statistical Frontiers scholarship under grant number CE140100049, and the NWO Gravitation Programme Networks under grant number 024.002.003.

The candidate received support for travel from University of Amsterdam, The University of Queensland, ARC Centre of Excellence for Mathematical and Statistical Frontiers, and The Network Center.

Keywords

Markov chains, stability, Monte Carlo algorithm, queueing networks, loss system, stochastic networks, multihop networks, fluid limit, diffusion limit, matrix analytic methods

Australian and New Zealand Standard Research Classifications (ANZSRC)

ANZSRC code: 010206, Operations Research, 40%
ANZSRC code: 010406, Stochastic Analysis and Modelling, 60%

Fields of Research (FoR) Classification

FoR code: 0102, Applied Mathematics, 40%
FoR code: 0104, Statistics, 60%

Contents

Overview, background, and motivation

This thesis is about coping with variability in outcomes for *complex random systems*. The focus is mainly upon systems where jobs arrive randomly throughout time to utilise resources of some sort for a random amount of time before departure. The primary focus is on models for communication and storage of data. We investigate, for example, models of radio networks, cloud file hosting services, and cloud computing platforms. These systems are rich enough that results for specific models are interesting in their own right, while also providing a fruitful environment for the development of novel *approaches* of interest to other application areas where similar models may arise (e.g., ecology, transportation, manufacturing, health care).

Models for communication and storage of data are often concerned with either queueing or loss behaviour, with Part I and Part II of this thesis devoted to these aspects respectively. Queueing occurs when a scarce resource is sought by more users than the amount available and it is possible for users to wait before accessing the resource. In such a case, *management* often entails determining an ideal trade-off between users waiting and investment in more resources. When there are multiple users or resource types the question of how to allocate resources between users also arises. Loss behaviour results when users are unable or unwilling to wait and so the effect of congestion is that users depart without accessing the resource. In this case management faces a trade-off between investing in more resources or losing a higher quantity of users.

In the following we consider a very simple scenario to highlight queueing and loss behaviour. Suppose time is broken into distinct consecutive periods, and at the beginning of each time period with probability $1/2$ an item arrives to be processed, then at the end of each time period if there are any items present, one is chosen at random and with probability $p \in [0, 1]$ is successfully processed and removed from the system. The number of items waiting in the system is clearly dependent on the value of p. Intuitively one might expect that low values of p result in more items waiting compared to values of p closer to 1. If there is an infinite capacity in the system for items to wait for processing, then management of the system may benefit from knowledge of the random fluctuations of this *queue* and how this relates to the time items must wait before being processed. On the other hand, suppose there is a finite capacity such that at most C items may wait for processing at any given time. Furthermore, suppose that if an item arrives to a system which is at capacity, then it departs without receiving service. In this case there is, in some sense, a maximum waiting time for items which enter the system and so management may be more concerned with ensuring that not too many items are *lost* without receiving service at all. From an elementary point of view, Part I of this thesis is concerned with parameters similar to p and Part II of this thesis is concerned with parameters similar to C.

There are many management questions of interest for the systems we investigate in this thesis. It is imperative in many cases to address the *stability* of a system before approaching other management questions. In the trade-off between investment in resources and waiting times, stability can be thought of as determining simply whether a chosen allocation of resources results in finite waiting times. We call systems where workload grows without bound, and consequently newly arriving work never gets processed, unstable. We spend the majority of Part I of this thesis determining conditions under which specific systems of interest are stable and developing methods for detecting instability. One motivation for this is that stability can be considered to be the most modest of long term management objectives. For our simple system, stability is achieved if and only if $p > 1/2$. If indeed an infinite waiting capacity is possible, then management may wish to carefully choose p to ensure that instability does not occur. In many cases, however, such a simple closed form expression for stability is not possible and so more sophisticated approaches are called for. Another motivation for studying stability is that the presence of instability can be indicative that the description of the system on which the model is based has flaws that need to be addressed. For our simple system, considering the finite quantity of items in the world it is unreasonable to assume that the probability of an arrival in each time period does not depend on the number of items being processed. A model assuming that the arrival rate decreases with the length of the queue may be more suitable for addressing the question of the queue's limiting length than the potentially infinite length description employed.

For systems characterised by loss, stability is no longer an issue — these systems are naturally bounded in size by the balking of jobs in response to congestion. For these systems it is natural to assume that finite waiting capacity is closely linked to scarcity of the resources required to generate waiting capacity. Moreover, scarcity is often closely related to price: we suppose that by paying more it is possible for management to increase waiting capacity. Hence there is a trade-off between provisioning capacity and losing jobs. This trade-off is the focus of Part II. For our simple system we may suppose that C units of capacity costs $\theta_c C$ per time unit to provide and that each lost job results in lost revenue of θ_l, and so C should be chosen to reduce the number of lost jobs, but not so high that costs are excessive. Maintaining an optimal balance between capacity and losses results in a more efficient system. For example, in the cloud computing context this means that it may be possible to reduce energy consumption by reducing redundancy, leading to a healthier natural environment and lower energy bills. On a more personal level it may just mean that an important text to a loved one is not lost en-route to its destination.

This thesis has two parts, Part I and Part II. As outlined in more detail in the introductory chapters, Chapter 1 and Chapter 5, these parts each begin by extending already established approaches to novel models of systems with contemporary relevance. Each part of the thesis concludes with a novel simulation-based approach that is applicable for models where already established methods are intractable. The established approaches we use are based on scaling limits, moment generating functions, Laplace–Stieltjes transforms, and matrix analytic methods. Our new simulation-based methods are exciting because they are efficient enough to easily be applicable to current state-of-the-art models and have scope to substantially extend the type of models that can be considered by applied probabilists. Furthermore, in some cases we provide rigorous guarantees on their performance.

Part I

Queueing models and stability

CHAPTER 1

Introduction to queueing models and stability

Queueing networks are an established class of mathematical models with a rich literature. Queueing results underpin and are conversely inspired by developments in areas such as wireless communications, manufacturing, transportation, logistics, and health care. In this chapter we provide a brief overview of selected topics from this literature. To this end we first give an introduction to some classical models and results, which the work in Chapter 2 directly builds upon. We then discuss the concept of stability and show some standard approaches to this topic, which leads us to introduce the more recently developed switch network model, as studied in Chapter 3. The chapter concludes with a discussion of the remaining chapters in this part. Briefly, these remaining chapters: (i) substantially generalise a well known queueing network model to include multiplicative transitions, (ii) provide interesting theoretical properties of a model for multihop radio networks, and (iii) devise a novel simulation-based method for detecting Markov chain instability.

1.1 Queueing networks

Many standard queueing models are characterised by a description of how jobs arrive to the system, the response of jobs to congestion and waiting times, the time it takes for a job to be served, a service discipline, the number of servers, and the ability of jobs to wait.

Consider a system where jobs arrive according to a homogeneous Poisson process (see [1]) with rate $\lambda \in \mathbb{R}_+$ (i.e., the times between arrivals are iid exponentially distributed random variables). The system consists of k servers, each of which can serve a single job at any time. Upon arrival, jobs wait indefinitely for a server to become available. Once a server becomes available a waiting job is selected for service uniformly at random, or if no jobs are waiting, the server idles until a job arrives. Once a job and a server are matched, an exponentially distributed amount of time with mean μ^{-1} passes and then the job leaves the system. Let $(X(t), t \in \mathbb{R}) \equiv X$ be a random process evolving on the state space $\mathcal{S} := \{0, 1, \dots\}$ that records the number of jobs in the system over time (which we assume has been running from time $-\infty$). Let $\mathcal{Q} = (q(i,j), i,j \in \mathcal{S})$ be a collection with $q(i,i) = 0$ and $q(i,j) \geq 0$ for $i \neq j$. Related to this collection are the quantities

$(q(i), i \in \mathcal{S})$ where $q(i) := \sum_{j \in \mathcal{S}} q(i,j)$, which we assume to be finite. Informally, suppose that upon entering a state $i \in \mathcal{S}$ the process remains in state i for an exponentially distributed amount of time with mean $q(i)^{-1}$ and then transitions to state $j \in \mathcal{S}$ with probability $p(i,j) := q(i,j)/q(i)$. Call $q(i,j)$ the transition *rate* from state i to state j. We have that $X(t)$ transitions to $X(t)+1$ and $X(t)-1$ at rates λ and $\mu \min\{k, X(t)\}$ respectively.

A collection of non-negative numbers $(\pi_i, i \in \mathcal{S})$ summing to unity that satisfy the *equilibrium equations*,

$$\pi_i \, q(i) = \sum_{j \in \mathcal{S}} \pi_j q(j,i), \quad \forall \, i \in \mathcal{S}\,, \tag{1.1}$$

is defined to be a *stationary* or *equilibrium* distribution of the system. If $(X(t_1), X(t_2), \ldots, X(t_n))$ has the same distribution as $(X(t_1+\tau), X(t_2+\tau), \ldots, X(t_n+\tau))$ for all $(t_1, t_2, \ldots, t_n, \tau) \in \mathbb{R}^{n+1}$, then we call X *stationary*. Many reasonably behaving processes, such as the one we are considering, are stationary when they have been running since time $-\infty$ (as we assume). If we can find a collection of positive numbers satisfying (1.1) whose sum is infinite, then an equilibrium distribution does not exist for X (see [2, p. 2]).

Although the system just described operates in continuous time, it is sometimes convenient to consider a discrete-time approximation $(X^{(J)}(t), t \in \mathbb{N}_0)$ to systems such as this one that records the state of the system only at time instances immediately after it changes state. For these discrete-time models the equilibrium distribution is defined by a collection of non-negative numbers $(\pi_i^{(J)}, i \in \mathcal{S})$ summing to unity that satisfy

$$\pi_i^{(J)} = \sum_{j \in \mathcal{S}} \pi_j^{(J)} p(j,i), \quad \forall \, i \in \mathcal{S}\,.$$

For Markov processes that almost surely have a finite number of transitions in any finite interval (such as the queueing system described above), when an equilibrium distribution exists for the discrete-time approximation it will also exist for the Markov process under study and vice versa. We use this property in Chapter 2 when establishing the existence of an equilibrium distribution of a continuous-time system, and the property can also be used to extend the framework we develop in Chapter 4 for discrete-time systems to continuous-time systems.

If $(X(t_1), X(t_2), \ldots, X(t_n))$ and $(X(\tau - t_1), X(\tau - t_2), \ldots, X(\tau - t_n))$ have the same distribution for all $t_1, t_2, \ldots, t_n, \tau \in \mathbb{R}$, then we call X *reversible*. It is well known that stationary processes where $q(i,j) = 0$ unless $|i-j| = 1$ are reversible (see e.g., [2, p. 11]). Therefore, since X satisfies this condition, according to Theorem 1.2 in [2] its equilibrium distribution can be identified by finding a collection of non-negative numbers $(\pi_i, i \in \mathcal{S})$ summing to unity that satisfy

$$\pi_i \, q(i,j) = \pi_j \, q(j,i), \quad \forall i, j \in \mathcal{S}\,.$$

These are known as the *detailed balance equations* [2]. For example, for our simple queueing system these are

$$\pi_i \lambda = \pi_{i+1}(i+1)\mu\,, \qquad 0 \le i < k\,, \tag{1.2}$$

$$\pi_i \lambda = \pi_{i+1} k \mu\,, \qquad i \ge k\,. \tag{1.3}$$

The parameter $\varrho = \lambda/\mu$, often referred to as the *traffic intensity*, is intimately related to the stationary distribution of the queue. For $k = 1$ it compares the average workload

imposed on the system to the average processing capability of the system. In this case, when $\varrho < 1$ the detailed balance equations can be solved to find that $\pi_i = (1-\varrho)\,\varrho^i$, where the constant $(1-\varrho)$ follows from the condition $\sum_{i=0}^{\infty} \pi_i = 1$. On the other hand, if ϱ is greater than 1, then more jobs are arriving on average than the system can serve on average throughout time. A reasonable hypothesis in this case is that the size of the queue will grow without bound over time — in which case we call the system *unstable* (a topic we return to in detail in the next section). Indeed, for $\varrho \geq 1$ we have identified a collection of positive numbers that satisfy (1.1) whose sum is infinite, indicating that a stationary distribution for the queue length does not exist in this case, confirming our hypothesis. When $\varrho < 1$ we say that the system is operating in a *subcritical* regime. For $k \to \infty$, for all possible values of ϱ, the detailed balance equations can be solved to find $\pi_i = \mathrm{e}^{-\varrho}\,\varrho^i/i!$. This indicates that a stationary distribution always exists for this system when the number of servers becomes infinitely large. In this case it can be seen that the expected value of the size of the queue in equilibrium is given by ϱ. The case $k = 1$ is known as a *single server queue*, while the case $k \to \infty$ is known as an *infinite server queue*. These are both special cases of the birth–death process introduced by Feller in [3].

Importantly, the stationary distribution of a stochastic system often underpins many performance measures for the system. For queueing systems where an equilibrium distribution exists, waiting is a key characteristic, and so it is common to use the long-term average waiting time in the system of an arbitrary customer, denoted by W, as a performance measure. In [4] it is shown that $L = \overline{\lambda} W$, where $\overline{\lambda}$ is the long-term average effective arrival rate and L is the long-term average length of the queue. This result, known as *Little's Law*, is highly general; in particular it can be applied to networks of queues, and holds regardless of the arrival process distribution, the service distribution, or service discipline [5]. This is extremely important since the vast majority of applications rely on a model consisting of many queues joined together as a *queueing network*.

When the primitive random variables underlying a queueing network are all exponentially distributed, the queueing network is a special case of a more broad class of models called *Markov population processes* [6]. These are vector valued Markov processes where each coordinate records the number of items given some classification as a function of time. In the queueing context each classification usually refers to an individual queue. Let the set of queues be $\mathcal{N} = \{1, \ldots, c\}$, where c is possibly infinite. Also, let the queue sizes at time t be given by a vector $\boldsymbol{X}(t) := (X_1(t), \ldots, X_c(t))$ and the equilibrium queue sizes be given by a vector $\boldsymbol{X} := (X_1, \ldots, X_c)$, both with state space $\mathcal{S} \subset \mathbb{N}_0^c$ and elements denoted by $\boldsymbol{x} = (x_1, \ldots, x_c)$ where each $x_i \in \mathbb{N}_0$. Now, the transition rates between states of this continuous-time Markovian pure-jump process are taken as

$$\begin{aligned} q(\boldsymbol{x}, \boldsymbol{x} + \boldsymbol{e}_i) &= \phi_i(\boldsymbol{x}), \\ q(\boldsymbol{x}, \boldsymbol{x} - \boldsymbol{e}_i) &= \psi_i(\boldsymbol{x}), \\ q(\boldsymbol{x}, \boldsymbol{x} - \boldsymbol{e}_i + \boldsymbol{e}_k) &= \kappa_{i,k}(\boldsymbol{x}), \end{aligned} \tag{1.4}$$

where $\boldsymbol{e}_i$ denotes the ith coordinate vector and ϕ_i, ψ_i, $\kappa_{i,k}$ are functions $\mathcal{S} \to \mathbb{R}_+$. We require $\psi_i(\boldsymbol{x}) = \kappa_{i,k}(\boldsymbol{x}) = 0$ if $x_i = 0$ for all i to ensure that $X_i(t)$ remains non-negative.

Early studies pursuing a closed form expression for the stationary distribution of a Markov population process include those by Jackson [7], Whittle [8], and Kingman [6]. For finite $\mathcal{S}$ the equilibrium equations (1.1) for the model described by (1.4) have a unique solution satisfying the properties of a probability distribution, while for infinite $\mathcal{S}$ they have at most one solution satisfying the properties of a probability distribution [6].

Although finding an explicit general and tractable solution to the equilibrium equations (1.1) that satisfies the properties of a probability distribution for the model described by (1.4) is widely thought to be impossible, progress has been made in certain interesting cases.

Jackson [7] considered the case when $\phi_i(\boldsymbol{x}) = \lambda_i$, $\psi_i(\boldsymbol{x}) = \mu_i \min\{x_i, s_i\}$, and $\kappa_{i,k}(\boldsymbol{x}) = \gamma_{i,k} \min\{x_i, s_i\}$, where λ_i, μ_i, $\gamma_{i,k} \in \mathbb{R}_+$, and $s_i \in \mathbb{N}$. He showed that the stationary distribution of the network factorizes as a product of the marginal stationary distributions of each queue and provided an explicit form of this expression (up to a normalizing constant) in terms of a set of constants that can be derived from a separate set of equations (given below in (1.6)). Whittle [8] showed that Jackson's result still holds for the more general case with $\phi_i(\boldsymbol{x}) = \lambda_i$, $\psi_i(\boldsymbol{x}) = \mu_i f_i(x_i)$, and $\kappa_{i,k}(\boldsymbol{x}) = \gamma_{i,k} f_i(x_i)$, where $f_i(0) = 0$ and $f_i(x) > 0$ for all $x > 0$. Upon taking $s_i \to \infty$ for all i in Jackson's expression or $f_i(x_i) = x_i$ for all i in Whittle's expression, the stationary distribution turns out to be

$$\pi_x = \prod_{i=1}^{c} \frac{\mathrm{e}^{-b_i/\mu_i}(b_i/\mu_i)^{x_i}}{x_i!}, \tag{1.5}$$

where the set $\{b_1, \ldots, b_c\}$ satisfies the system of *traffic equations*

$$b_i = \lambda_i + \sum_{k=1}^{c} b_k \theta_{i,k}, \quad \forall\, i \in \mathcal{N}, \tag{1.6}$$

with $\theta_{i,k} = \gamma_{i,k}/(\mu_i + \sum_j \gamma_{i,j})$ denoting the probability that an item is sent to queue k upon completion of service at queue i. This indicates that in equilibrium the distribution of the number of items at a given queue is independent of the number of items at other queues and follows a Poisson distribution with mean determined by (1.6). Notice that this limit in s_i or choice of f_i results in a network of infinite server queues model, indicating that an equilibrium distribution always exists for such a system (our model in Chapter 2 modifies the network of infinite server queues model in such a way that an equilibrium distribution may not exist). It is also interesting to note that the traffic equations (1.6) depend only on the external arrival rates and the transition probabilities between queues, not on the service rates at the individual queues.

In addition to being able to evaluate performance measures based on the equilibrium distribution, it can also be of tremendous use to system managers to be able to optimise in terms of these measures. Early work by Kleinrock [9] focused on how to best assign service capacity to optimise various performance measures for queueing networks. This research was based on the assumption that service times were exponentially distributed and input processes are Poisson. It was later realised that such an assumption may not hold, yet without these assumptions exact analysis seems to be impossible, and so Pollett [10] developed an approximating technique based on the residual-life approximation. Additionally, there is a very large body of literature on insensitivity to the form of the service distributions provided the mean is kept constant (see e.g., [104] for details). More recent state-of-the-art research [11] focuses on combining exact analytical results that utilise the exponential assumption with simulation to develop fast methods for determining optimal solutions. Before optimisation can be addressed, however, it is often necessary to first determine if and when a particular model will be stable or unstable.

1.2 Stability

A stable queueing network is guaranteed to have sufficient service to cope with the load imposed upon it in the long-run. Intuitively this means that over very long planning horizons the number of jobs in the network remains bounded so that individual jobs expect to experience finite waiting times. More formally it means that it is possible to define a valid probability distribution to describe the system as time goes to infinity. Recall the Markovian population process model above with $\phi_i(\boldsymbol{x}) = \lambda_i$, $\psi_i(\boldsymbol{x}) = \mu_i f_i(x_i)$, and $\kappa_{i,k}(\boldsymbol{x}) = \gamma_{i,k} f_i(x_i)$, where $f_i(0) = 0$ and $f_i(x) > 0$ for all $x > 0$. For this class of systems we saw that a particular choice of f_i results in a model for a system which always has an equilibrium distribution. In fact, by imposing the condition $f_i(x) > 0$ for all i along with some conditions on the set $(\mu_i, \gamma_{i,k})$ that ensure from any state there is a positive probability of eventually visiting any other state we obtain a property known as *irreducibility*. In this thesis we deal exclusively with irreducible systems, and for this type of system there is a strong equivalence, summarised in the following theorem, between the existence of a stationary distribution for a process and the expected time it takes the process to return to any state from the moment it departs the state.

Theorem 1 (Partial restatement of Theorem 3 in Section 6.4 of [12])**.** *An irreducible Markov process evolving on a countable state space has a stationary distribution if and only if the expected return time is finite for some state.*

Before continuing we must make two remarks regarding Theorem 1. Firstly, inspection of the reference for the theorem reveals that the theorem is in fact stated for discrete-time Markov processes evolving on a discrete state space. Given a continuous-time model (e.g. $(X(t),\, t \in \mathbb{R}_+)$) with bounded transition rates it is possible to define a related *skeleton Markov process* (e.g. $(X(\Delta n),\, n \in \mathbb{Z}_+)$) which has a stationary distribution if and only if the continuous-time model does too. This is because the expected return times to each state for the skeleton process will provide upper bounds for the expected return times of the original process. Secondly, the notion of a state having a finite return time is so important it has the special name *positive recurrence*, a name which is adopted by a Markov process when all of its states have this property. Due to Theorem 1, the applied probability community use the terms *stability* and positive recurrence interchangeably with the notion of the existence of a stationary distribution. An *unstable* queueing network is one where a stationary distribution does not exist; for such systems there exists some initial state such that, as time goes to infinity, the number of jobs in the network will go to infinity (and then never come back down) with positive probability (see e.g., [84, p. 53]).

Returning to the Markov population process models introduced earlier, in the Whittle case, define the constants

$$g_i = \sum_{n=1}^{\infty} \frac{(b_i/\mu_i)^n}{\prod_{r=1}^{n} f_i(r)}, \quad i \in \mathcal{N}\,.$$

Inspection of (1.6) hints that the constants b_i, and then also g_i, are related to the total load imposed on any particular queue in the network in equilibrium. In fact, in order for a stationary distribution to exist $g_i < \infty$ is required for all $i \in \mathcal{N}$. For example, consider the case of a sequence of c single server queues in tandem, which occurs when $\phi_1(\boldsymbol{x}) = \lambda$, $\psi_c(\boldsymbol{x}) = \mu_c 1\{x_c > 0\}$, $\kappa_{i,i+1}(\boldsymbol{x}) = \mu_i 1\{x_i > 0\}$ for $1 \leq i < c$, and all other transitions

are 0. In this case $b_i = \lambda$ and so g_i is only finite for $\lambda/\mu_i < 1$. This stability condition is highly intuitive, it essentially says that the total traffic entering the network for each queue must be less than the rate at which work can be processed. Another way to put this is that the traffic intensity λ/μ_i (as discussed earlier in this chapter) must be sub-critical (i.e., less than 1) for all queues. For the most part this resolves the question of stability for many queueing networks; however, in the next two sections we will explore two other types of network where a stationary distribution has not been found and other methods for determining stability are necessary.

1.3 Foster–Lyapunov stability

In this section we will demonstrate the utility of another method for determining stability properties for a stochastic system. Specifically, we present a model where the service allocation depends on the current state of the system. For this model a stationary distribution has not yet been determined, motivating us to state the *Foster–Lyapunov Theorem* which can be used, nonetheless, to approach the question of stability for such a system. Multihop radio network models (see e.g., [13, 14, 15, 16]) are an important class of random network that share many of the features of the continuous-time first-in-first-out (FIFO) networks discussed at length in the previous section. These models are distinct, however, in that they typically have a discrete time index and more sophisticated service disciplines than the processes of Jackson and Whittle specified in the previous section. We focus on this type of model in Chapter 3. A continuous-time variant of a typical system from this area of research could be specified with appropriate ϕ, ψ, and κ; however, we will now instead more directly define the system in discrete time.

There are links contained in a finite countable set $\mathcal{L}$ serving several routes of traffic. Each route r consists of N_r ordered links $\{l_1^{(r)}, \dots, l_{N_r}^{(r)}\} \subset \mathcal{L}$. For each link there is an associated queue that keeps track of the quantity of jobs at that link, and similarly for each route at each link there is a queue. At time $t \in \{0, 1, \dots\}$ we denote these queues as follows: there are $Q_l(t)$ jobs at link l, and moreover there are $X_{l,r}(t)$ jobs at link l on route r. In each time slot $A_r(t)$ route r jobs arrive exogenously to link $l_1^{(r)}$, where $(A_r(t), t \geq 0)$ are iid. We let $\lambda_r := \mathbb{E}A_r(1)$ and store these quantities in a vector denoted by λ. Let λ_Σ be a vector with l-th component $\sum_{r:l\in r} \lambda_r$. In each time period a scheduling policy is determined such that $\mu_{l,r}(t)$ route r jobs are expected to be served at link l. Jobs which are served move to the next link on their route before the next time period begins, or leave the system if they are on the terminal link of the route (i.e., the link with index N_r). We define $\sigma_l(t) = \sum_{r:l\in r} \mu_{l,r}(t)$ and suppose that the set $\{\sigma_l(t)\}$ must be chosen from within a compact convex region $\mathbf{\Lambda} \subset \mathbb{R}_+^{|\mathcal{L}|}$. It was shown in [13] that λ_Σ must be an element of the interior of $\mathbf{\Lambda}$ in order for there to exist a scheduling policy for which the system is stable. Such a policy is called *maximally stable* or *throughput optimal*.

One such maximally stable policy is the celebrated *Backpressure* policy of Tassiulas and Ephremides [13]. In each time period this policy chooses $\{\mu_{l,r}(t)\}$ to solve the optimisation problem:

$$\max \quad \sum_{(l,r)} \left[X_{l_-^{(r)},r}(t) - X_{l^{(r)},r}(t) \right] \mu_{l,r}(t) \quad \text{subject to} \quad \sigma(t) \in \mathbf{\Lambda}\,, \tag{1.7}$$

where $l_-^{(r)}$ is the link before link $l^{(r)}$ on route r and $X_{l_-^{(r)}}(t) = 0$ for all ingress links $l_1^{(r)}$.

Although it is has not been possible yet to explicitly find the stationary distribution of this type of network under this policy, it is still possible to determine that it is stable for λ_Σ strictly an element of the interior of $\mathbf{\Lambda}$. A key result that is used to prove this result is the Foster–Lyapunov theorem.

Suppose we are working with a discrete-time Markov process $(X(t), t \in \mathbb{N})$ evolving on a countable state space $\mathcal{S}$ (which may be vector valued). Then the following theorem, based on ideas for ordinary differential equations, allows us to establish stability of the process by considering how movements over a single time period for states inside and outside of some finite set are related to the overall expected return time for all states.

Theorem 2 (Foster-Lyapunov stability criterion)**.** *Suppose there exists a function $V : \mathcal{S} \to \mathbb{R}_+$ such that for some constants $\varepsilon > 0$ and b, some finite exception set $K \subset \mathcal{S}$, and all $i \in \mathcal{S}$,*

$$\mathbb{E}[V\big(X(t+1)\big) - V\big(X(t)\big) \,|\, X(t) = i] \leq \begin{cases} -\varepsilon, & i \notin K, \\ b - \varepsilon, & i \in K. \end{cases}$$

Then the expected return time to K is finite, and X is a positive recurrent (stable) Markov process.

This theorem can be used in conjunction with Theorem 1 to show that a stationary distribution exists, even when that distribution cannot be explicitly found. In [13] the function $V(\boldsymbol{X}(t)) = \sum_{(l,r)} X_{l,r}(t)^2$ is used in Theorem 2 to prove the Backpressure policy is maximally stable. Identifying an appropriate V is often the main challenge in applying this theorem. In Chapter 2, however, we use Theorem 2 with V being the L^1-norm (of the queue size vector) to prove stability — in this case a major hurdle is identifying an expression for the expected change of the process over time. In Chapter 3 we encounter a model where Theorem 2 cannot be applied directly and a scaling limit approximation must be used to approach the question of stability. In the next section we introduce this scaling limit and show its relation to stability. We do this in the context of some famous counterexamples showing that subcritical traffic intensity is not sufficient for stability for all Markovian population process type queueing networks.

1.4 Fluid scaling limits and stability

The Rybko–Stolyar [17] and Lu–Kumar [18] systems are examples of queueing networks that fit into the dynamics described by (1.4) and can be unstable with subcritical traffic intensity. These examples were introduced independently in the early 1990's. Since these systems exhibit similar dynamics, we will focus only on the Rybko-Stolyar network, which was introduced in a paper that was among the first to use the fluid model technique for determining stability that we introduce in this section. The network arises again in Chapter 4 as an example for our novel simulation-based method for detecting instability.

The Rybko–Stolyar network consists of four queues, displayed in Figure 1.2, grouped into two *stations*, where jobs enter the network at one of the stations and upon completion of service move to the other station for service. At each station there is a queue for newly arriving jobs and a queue for jobs that arrived from the other station, and at each station only one queue may receive service at any particular point in time. Jobs which have already been processed at their ingress station receive priority over jobs being processed for the first time. The system has dynamics in terms of (1.4) as follows: $\phi_1(\boldsymbol{x}) = \phi_4(\boldsymbol{x}) = \lambda$, $\kappa_{1,2}(\boldsymbol{x}) = \mu_1 1\{x_1 > 0, x_2 = 0\}$, $\kappa_{4,3}(\boldsymbol{x}) = \mu_2 1\{x_4 > 0, x_3 = 0\}$, $\psi_2(\boldsymbol{x}) = \mu_2 1\{x_2 > 0\}$,

$\psi_3(\boldsymbol{x}) = \mu_1 1\{x_3 > 0\}$, and no other transitions. From [17] we have that the system is unstable when $\lambda = 1$, $\mu_2 > 0$, and $\mu_1 < 2$.

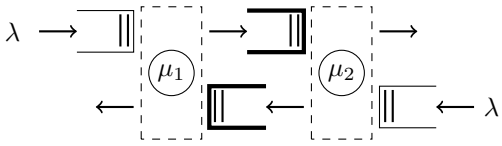


Figure 1.1: The Rybko–Stolyar network.

In Figure 1.2 we see a simulated sample path of this system with $\lambda = 1$, $\mu_2 = 2$, and $\mu_1 = 1.99$. The figure shows very interesting behaviour: the queues do appear to always return to 0, but as time progresses the return time appears to increase. Indeed, with this choice of parameters the system is unstable. Determining the stationary distribution for this system in the stable case in the same way as for the Jackson and Whittle type systems introduced earlier has not yet, to the author's knowledge, been done. Nonetheless, it is still desirable to determine when a stationary distribution exists. The machinery used in [19] and Chapter 3 are based on a particular type of scaling that we will now discuss.

Although the development of a universally applicable and implementable method for determining stability and instability remains an open question, a method relying on *fluid scaling limits* is a popular candidate for such a role. Loosely speaking, in a fluid scaled process, jumps whose size are of order 1 occurring at rate λ are scaled to have a size of order $1/n$ and made to occur at rate λn, and then n is taken to be very large. The initial condition of the scaled stochastic process converges (often by assumption) in distribution to a random vector (possibly of constants) as $n \to \infty$. More formally, given a continuous-time Markov process $(X(t),\, t \geq 0)$, for each n consider the fluid scaled process

$$\widetilde{X}^{(n)} := \left(\frac{X(n\,t)}{n}, \quad t \geq 0 \right) .$$

We call a fixed function $\widetilde{x} \equiv (\widetilde{x}(t),\ t \geq 0)$ a *fluid sample path* if there exists an arbitrary nondecreasing sequence of positive numbers tending to infinity $\mathcal{K}$ and a sequence of sample paths $\{\widetilde{x}^{(n)}\}$ of the corresponding processes $\{\widetilde{X}^{(n)}\}$, such that as $n \to \infty$ along sequence $\mathcal{K}$,

$$\widetilde{x}^{(n)} \to \widetilde{x}\,,$$

uniformly on compact intervals.

As a standard example, consider the individual queue introduced at the beginning of this chapter with $k = 1$. It is possible to show (see e.g., [20, Section 5.7]) that for this single server Markovian queue, under the assumption that fluid scaled paths satisfy $\widetilde{X}^{(n)}(0)/n \to \widetilde{x}(0) \in \mathbb{R}_+$ almost surely (i.e., with probability 1), that almost all (i.e., with probability 1) fluid sample paths are given by

$$\widetilde{x}(t) = \max\{\widetilde{x}(0) + (\lambda - \mu)\,t,\ 0\}\,, \quad t \geq 0\,.$$

It is crucial to note that for $\lambda < \mu$ this function goes to 0 and for $\lambda \geq \mu$ it remains bounded away from 0. We saw earlier that in the former case a stationary distribution exists for this system and in the latter case a stationary distribution does not exist. A rigorous relationship between stability of a stochastic system and the almost sure convergence to 0 sample paths property of its corresponding fluid model has been established in the applied

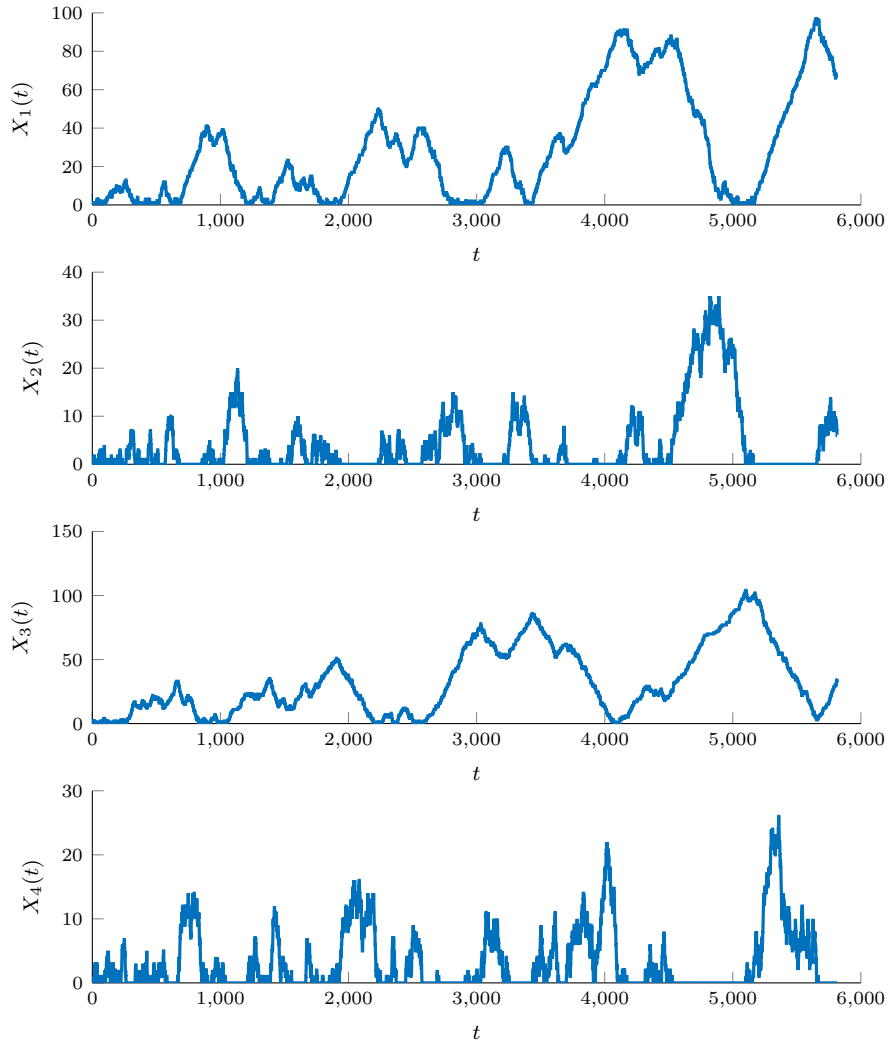


Figure 1.2: Sample path of Rybko-Stolyar system with unstable parameter choices.

probability literature.

The following theorem, rephrased from Dai [21], provides sufficient conditions that can be used to establish stability of a relatively large family of queueing networks, including the simple example just explored and the model studied in Chapter 3.

Theorem 3 (Restatement and rephrase of Theorem 4.2 in [21]). *Given a continuous-time Markov process X satisfying some technical conditions, if there exists a constant $T \in (0, \infty)$ such that for almost all fluid sample paths $\widetilde{x}$ it holds that $|\widetilde{x}(t)| = 0$ for all $t \geq T$ almost surely, then X is positive recurrent.*

Showing that almost all fluid sample paths almost surely go to 0 can be a challenge. A popular method is to find an appropriate Lyapunov function. Fortunately this is often more straightforward than finding a Lyapunov function that deals directly with the system

dynamics, which is the strength of Theorem 3 over Theorem 2. We wish to highlight that this theorem is only sufficient for stability when almost all fluid sample paths go to 0 almost surely. If, in the single server example above, the arrival rate λ and service rate μ were to be switched with probability 1/2 at the first instance that $X(t) = 0$, then a fluid model with random sample paths would result. In this circumstance 1/2 of the fluid sample paths would not go to 0 and so the conditions of Theorem 3 would not be met and a conclusion of stability could not be made. The properties of models which possess more than 1 potential fluid sample path are an active area of research (see e.g., [22, 23, 24]). Concluding *instability* from fluid models is also a fundamental part of the fluid scaling stability framework (see e.g., [25, 26]).

Utilising Theorem 3 in place of Theorem 2 often comes with the added benefit of finding simpler conditions on sample paths at the cost of needing to first establish a fluid model for the stochastic system of interest. Three traditional techniques of proving convergence of Markov processes to their fluid limits are: (i) C-tightness criterion (as used in e.g., [27, 28, 29]), (ii) the martingale representation (as used in e.g., [30]), and (iii) convergence of generators (see e.g., [31] for details). Our proof of convergence to a fluid model in Chapter 3 utilises a martingale, and this type of process (see Appendix A.2) is also a fundamental ingredient of our results in Chapter 4.

The rest of Part I of this thesis presents original research results, which we will now summarise.

1.5 Outline of Part I

Here we provide a summary of the subsequent chapters of Part I of this thesis. Each chapter uses different mathematical methods and has its own distinct focus, yet stability is a prevailing theme throughout. Each remaining chapter in this part utilises different subsets of the ideas introduced in this chapter and is presented in an accessible and self-contained fashion.

Chapter 2 — Networks of infinite-server queues with multiplicative transitions. This chapter considers a network of infinite-server queues with the special feature that, triggered by specific events, the network population vector may undergo a linear transformation. Due to these transformations, the model does not fit into the class of models described by (1.4). For this model we characterize the joint probability generating function in terms of a system of partial differential equations; this system enables the evaluation of (transient as well as equilibrium) moments. We show that several relevant systems fit in the framework developed, such as networks of retrial queues, networks in which jobs can be rerouted when links fail, and storage systems. Numerical examples illustrate how our results can be used to support design problems.

Chapter 3 — Stability of weighted queue-proportional rate allocation. We study a weighted version of the recently introduced Queue-Proportional Rate Allocation scheduling algorithm for multi-hop radio networks. This policy does not require knowledge of traffic arrival rates to make scheduling decisions. We use fluid limit techniques to show that a generalization of this policy to allow weighting of packets at each station, to reflect non-homogeneous priorities, retains the maximal stability property. We do this in a setting where the processing capabilities of the network varies over time according to a finite-state, discrete-time Markov chain. We also apply diffusion scaling to the model

and investigate the properties of the model in heavy traffic. We conjecture that the workload process of the network converges to a reflected Brownian motion, that the vector of queue lengths is always proportional to the traffic arrival rate vector, and that the policy asymptotically minimises workload.

Chapter 4 — Detecting Markov chain instability: a Monte Carlo approach. We devise a Monte Carlo based method for detecting whether a non-negative Markov chain is stable for a given set of parameter values. More precisely, for a given subset of the parameter space, we develop an algorithm that is capable of deciding whether the set has a subset of positive Lebesgue measure for which the Markov chain is unstable. The approach is based on a variant of simulated annealing, and consequently only mild assumptions are needed to obtain performance guarantees.

The theoretical underpinnings of our algorithm are based on a result stating that the stability of a set of parameters can be phrased in terms of the stability of a single Markov chain that searches the set for unstable parameters. Our framework leads to a procedure that is capable of performing statistically rigorous tests for instability, which has been extensively tested using several examples of standard and non-standard queueing networks.

The following publication has been incorporated as Chapter 2.
[32] D. Fiems, M. Mandjes, and **B. Patch**. Networks of infinite-server queues with multiplicative transitions, *Performance Evaluation*, 123–124 (2018). pp. 35–49.

Contributor	Statement of contribution	%
Dieter Fiems	writing of text	33
	proof-reading	33
	supervision, guidance	50
	theoretical derivations	33
	preparation of figures	33
	initial concept	33
Michel Mandjes	writing of text	33
	proof-reading	33
	supervision, guidance	50
	theoretical derivations	33
	preparation of figures	33
	initial concept	33
Brendan Patch	writing of text	33
	proof-reading	33
	theoretical derivations	33
	preparation of figures	33
	initial concept	33

CHAPTER 2

Networks of infinite-server queues with multiplicative transitions

2.1 Introduction

The vast majority of queueing network models studied in the literature are of the form: there is a set of nodes that are fed by streams of external arrivals, and a routing mechanism that determines to which queue served clients are forwarded or whether the client leaves the system altogether. The most common queueing disciplines are of single-server type (entailing that clients may have to wait until they get into service) and of infinite-server type (in which all customers present at a node are served in parallel).

A key feature of the conventional class of models described above is that clients join and leave queues *one by one.* In many applications, however, triggered by specific events, the *full population* of individual queues may move around the network (or leave the system altogether). Particularly in the reliability and availability context, there are many relevant examples of such dynamics. We could for instance think of a data communication network with unreliable nodes: at the moment that a node goes down, all traffic residing at the node may be instantly lost. Another example concerns rerouting: triggered, for instance, by a link failure, clients are moved from one set of resources to an alternative set (the 'backup route'). Due to the fact that they correspond to transitions of the entire population of specific queues, the dynamics of the above two examples do not align with those of conventional queueing models.

Scope, object of study. Motivated by the above examples, the main objective of the present chapter is to analyse queueing networks *with multiplicative transitions.* These multiplicative transitions effectively entail that the network dynamics include transitions by which the network's population vector, say $\boldsymbol{m}$, is (pre-)multiplied by a matrix A with integer-valued, non-negative entries, so that the network population after the transition becomes $A\boldsymbol{m}$. For instance, choosing A to be a diagonal matrix with $[A]_{ii} = 0$ and $[A]_{kk} = 1$ for all $k \neq i$ would correspond to the event of all clients at node i being lost. Relocation of clients can be taken care of in a similar manner: the full population of queue i moving to queue j corresponds to $[A]_{ji} = 1$, $[A]_{kk} = 1$ for all $k \neq i$, and all other entries equal to 0.

In this chapter the queues considered are of *infinite-server* type. This type of queue is particularly relevant in contexts where the sojourn time at a node of each client is not (or hardly) affected by other clients. As such, the model has a broad variety of applications, ranging from the number of websurfers simultaneously present at a set of websites, to the number of messenger RNA molecules simultaneously present in a collection of cells (see e.g., [33]). A specific application that features in the present chapter concerns the optimal design of storage networks. To make the model as widely applicable as possible, we assume that all relevant transition rates (i.e., arrival rates and departure rates) are affected by an external autonomously evolving Markovian environmental process; the resulting model is therefore of a *Markov modulated* nature. As will become clear, in a reliability context such an environmental process can be used to model the state of the nodes of the network (i.e., 'up' or 'down').

Contributions. The chapter has two main contributions. (i) In the first place we set up a general model of a network of infinite-server queues with multiplicative transitions. For this model we derive a system of partial differential equations that describe its time-dependent behaviour (in terms of the probability generating function of the joint queue length distribution), as well as a procedure to evaluate the corresponding moments. The model turns out to have a non-trivial stability condition (under which the system's stationary behaviour is well-defined), which we establish using the expression we found for the time-dependent mean. (ii) In the second place, we point out that various natural, practically relevant models fit in our framework. Most notably, we concentrate on a network of retrial queues, a network with rerouting, and a storage network. Our results can be used to support various design decisions. In the storage system application, for instance, interesting trade-offs can be numerically assessed: files are typically stored on multiple locations so as to mitigate the risk of loss, but evidently one wants to do so without using an unnecessarily large amount of storage space.

Literature. As mentioned above, in typical queueing network models the number of clients per queue changes by one at a time; see e.g. the standard textbooks [2, 34]. Several papers, such as [35, 36, 37, 38, 39, 40], consider queues with batch arrivals and batch services and find product-form results, but these typically neither cover our multiplicative update rule nor allow the transition rates to be affected by an environmental process. We also refer to the related papers [41, 42, 43].

As mentioned above, a relevant special case of our model corresponds to the context of reliability. In many situations, when a network resource (a node or a link) fails, all clients using it will be lost. Such models are known as queueing models with *catastrophes*; for a fairly complete account of such models, we refer to the recent literature review in [44, Section 1]. The models studied are typically (but not always) one-dimensional; interesting contributions include [45, 46].

Queueing models for which the underlying infrastructure alternates between being 'up' and 'down' can be seen as examples of stochastic processes on dynamically evolving graphs. Despite the sizeable literature on random graphs, the body of work on dynamic random graphs is considerably smaller, and (evidently) the body of work covering stochastic processes on dynamic random graphs is even smaller. In recent contributions, results on dynamic random graphs have been reported; see e.g. [47, 48, 49, 50, 51]. This chapter can be regarded as being among the first pieces of work to facilitate describing queueing processes on a randomly evolving graph (but it is noticed that the model we propose is substantially more general, as the multiplicative transitions are not restricted to node

failures and repairs).

As mentioned, our model covers various practically relevant models as special cases. In each of the corresponding application areas there is a large collection of papers and textbooks available; in Section 2.3 we include a number of domain-specific references.

Organization. The rest of the chapter is organized as follows. Section 2.2 presents the model in its generic form, and after some preliminaries, the results in terms of partial differential equations characterizing the joint probability generating function and ordinary linear differential equations characterizing the moments. In addition, the stability condition is provided, under which stationary moments exist, which can be found by solving systems of linear equations. Section 2.3 gives an indication of the width of our framework: we show that it covers a network of retrial queues, a network with rerouting, and a storage network. Section 2.4 demonstrates a couple of design issues that can be resolved by using our machinery. Finally, Section 2.5 provides a discussion and concluding remarks.

2.2 Analysis

This section studies our generic model: a network of infinite-server queues with multiplicative transitions. We first introduce the model, then study its time-dependent behaviour, derive its stability condition, and conclude by commenting on numerical issues.

2.2.1 Model

In this subsection we describe our network of infinite-server queues with multiplicative transitions between the nodes. Let $\mathscr{N} := \{1, \ldots, N\}$ (with $N \in \mathbb{N}$) be the set of infinite-server queues. The object of study is $(\boldsymbol{M}(t))_{t\geqslant 0}$ (with $\boldsymbol{M}(t) \in \mathbb{N}_0^N$), that is, the multivariate queue content process (also sometimes referred to as the network population process). The process $(X(t))_{t\geqslant 0}$ (with $X(t) \in \mathscr{I} := \{1, \ldots, I\}$, where $I \in \mathbb{N}$) is the environment process (or: background process), which evolves autonomously of the queue content process; in our set-up, $(X(t))_{t\geqslant 0}$ is assumed to be an irreducible continuous-time Markov chain.

The following transition rates play a role:

- The rate $\lambda_n^{(i)} \geqslant 0$ is the external arrival rate at queue $n \in \{1, \ldots, N\}$ when the background process $X(\cdot)$ is in $i \in \mathscr{I}$. Note that this entails that the arrival process at each of the queues is a Markov-modulated Poisson process.

- Likewise, the rate $\mu_{nn'}^{(i)} \geqslant 0$ is the departure rate *for every customer present* from queue n to queue n' when the background process $X(\cdot)$ is in $i \in \mathscr{I}$. Here $n \in \mathscr{N}$ and $n' \in \mathscr{N} \cup \{0\}$, where $n' = 0$ corresponds to the client leaving the network. Note that at the queues, the clients are served simultaneously, reflecting the infinite-server nature of each of the queues.

- Define for each pair (i, j) with $i, j \in \mathscr{I}$ such that $i \neq j$ the set $\mathscr{K}_{ij} := \{1, \ldots, K_{ij}\}$ with $K_{ij} \in \mathbb{N}$ given. Let, for each $k \in \mathscr{K}_{ij}$, $A_{ij}^{(k)}$ be an $(N \times N)$-matrix with entries in $\mathbb{N}_0$. The rate $\alpha_{ij}^{(k)} \geqslant 0$ is the rate at which the queue content, say $\boldsymbol{m} \in \mathbb{N}_0^N$, is converted into $A_{ij}^{(k)}\boldsymbol{m}$, and at the same time the environment process $X(\cdot)$ jumps from state i to state j, for $i, j \in \mathscr{I}$. For obvious reasons, we refer to these events as *multiplicative transitions.*

Two issues are worth highlighting. (i) Note that the above description does not explicitly include notation for state transitions of the background process that do not involve multiplication with an A-matrix. It is easily observed, however, that such transitions can be introduced by letting the A-matrix correspond to an identity matrix. (ii) Transitions from i to j with $i = j$ ('self-transitions') are allowed. Our set-up in this respect differs from how continuous-time Markov processes are typically defined; observe that $X(\cdot)$ is nonetheless a continuous-time Markov chain.

Notice that it can be anticipated that this system has a non-trivial stability condition. Observe that if some of the A-matrices have entries larger than 1, the parameters may be such that the network population can grow quickly and eventually explode. When the stability condition applies, however, this cannot happen. We derive the stability condition in Section 2.2.4. Evidently, the system's time-dependent behaviour can be studied regardless of the validity of such a stability condition; this time-dependent behaviour is the topic of Sections 2.2.2–2.2.3. In Section 2.2.5 we comment on the numerical evaluation of the performance measures under study.

2.2.2 System of partial differential equations

The objective of this subsection is to characterize the distribution of $(\boldsymbol{M}(t), X(t)) \in \mathbb{N}_0^N \times \mathscr{I}$. We take the classical approach of setting up a system of partial differential equations for the corresponding transforms. To this end, we first define, for $i \in \mathscr{I}$, $t \geqslant 0$, and $\boldsymbol{\vartheta} \in \mathbb{R}^N$,

$$\varphi_i(\boldsymbol{\vartheta}, t) := \mathbb{E}\big(\mathrm{e}^{\langle \boldsymbol{\vartheta}, \boldsymbol{M}(t)\rangle} I_i(t)\big),$$

with $I_i(t) := 1\{X(t) = i\}$, the indicator function for the event that $X(t)$ equals i; we follow the convention that $\langle \boldsymbol{a}, \boldsymbol{b}\rangle$ denotes the inner product $\sum_{n=1}^N a_n b_n$, where a_n (b_n) denotes the nth element of the vector $\boldsymbol{a}$ ($\boldsymbol{b}$). From standard theory, the quantities $\varphi_i(\boldsymbol{\vartheta}, t)$ uniquely describe the system's probabilistic behaviour.

So as to set up the differential equations, the main idea is to relate the state of the system at time $t + \Delta t$ to the state at time t, for Δt small. We rely on the usual 'Markovian reasoning', meaning that when the environmental process is in state i at time t the following three types of events have to be considered: (i) with a probability of essentially $\lambda_n^{(i)} \Delta t$ there is an external arrival at node n, (ii) with probability $\mu_{nn'}^{(i)} M_n(t)\, \Delta t$ there is a departure from node n to n' (with n' possibly equalling 0, to model the clients that leave the network), and (iii) with probability $\alpha_{ij}^{(k)} \Delta t$ the environmental process jumps to j while simultaneously the network population vector $\boldsymbol{M}(t)$ is instantly replaced by $A_{ij}^{(k)} \boldsymbol{M}(t)$. Working out these transitions in detail, elementary calculations reveal that, as $\Delta t \downarrow 0$,

$$\varphi_i(\boldsymbol{\vartheta}, t + \Delta t) = \varphi_i(\boldsymbol{\vartheta}, t) + \xi_i(\boldsymbol{\vartheta}, t)\Delta t - \zeta_i(\boldsymbol{\vartheta}, t)\Delta t + \mathrm{o}(\Delta t), \tag{2.1}$$

where

$$\begin{aligned}\xi_i(\boldsymbol{\vartheta}, t) \;:=\; & \sum_{n=1}^N \mathrm{e}^{\vartheta_n} \mathbb{E}\big(\mathrm{e}^{\langle \boldsymbol{\vartheta}, \boldsymbol{M}(t)\rangle} I_i(t)\big)\, \lambda_n^{(i)} + \sum_{n=1}^N \sum_{n'=1}^N \mathrm{e}^{-\vartheta_n + \vartheta_{n'}} \mathbb{E}\big(\mathrm{e}^{\langle \boldsymbol{\vartheta}, \boldsymbol{M}(t)\rangle} I_i(t)\, M_n(t)\big)\, \mu_{nn'}^{(i)} \\ & + \sum_{n=1}^N \mathrm{e}^{-\vartheta_n} \mathbb{E}\big(\mathrm{e}^{\langle \boldsymbol{\vartheta}, \boldsymbol{M}(t)\rangle} I_i(t)\, M_n(t)\big)\, \mu_{n0}^{(i)} + \sum_{j=1}^I \sum_{k=1}^{K_{ji}} \mathbb{E}\big(\mathrm{e}^{\langle \boldsymbol{\vartheta}, A_{ji}^{(k)} \boldsymbol{M}(t)\rangle} I_j(t)\big)\, \alpha_{ji}^{(k)}\end{aligned}$$

and

$$\zeta_i(\boldsymbol{\vartheta},t) := \sum_{n=1}^{N}\mathbb{E}\big(\mathrm{e}^{\langle\boldsymbol{\vartheta},\boldsymbol{M}(t)\rangle}I_i(t)\big)\,\lambda_n^{(i)}+\sum_{n=1}^{N}\sum_{n'=1}^{N}\mathbb{E}\big(\mathrm{e}^{\langle\boldsymbol{\vartheta},\boldsymbol{M}(t)\rangle}I_i(t)\,M_n(t)\big)\,\mu_{nn'}^{(i)}+$$
$$\sum_{n=1}^{N}\mathbb{E}\big(\mathrm{e}^{\langle\boldsymbol{\vartheta},\boldsymbol{M}(t)\rangle}I_i(t)\,M_n(t)\big)\,\mu_{n0}^{(i)}+\sum_{j=1}^{I}\sum_{k=1}^{K_{ij}}\mathbb{E}\big(\mathrm{e}^{\langle\boldsymbol{\vartheta},\boldsymbol{M}(t)\rangle}I_i(t)\big)\alpha_{ij}^{(k)}.$$

To understand the structure of $\xi_i(\boldsymbol{\vartheta},t)$ and $\zeta_i(\boldsymbol{\vartheta},t)$, note that their first terms reflect the external arrivals, the second terms the routing to other queues, the third terms clients leaving the network, and the fourth terms the multiplicative transitions.

The next step is to rewrite the expressions for $\xi_i(\boldsymbol{\vartheta},t)$ and $\zeta_i(\boldsymbol{\vartheta},t)$ in terms of partial derivatives with respect to the arguments ϑ_n, $n\in\mathscr{N}$. We thus obtain, with A^{T} denoting the transpose of the matrix A,

$$\xi_i(\boldsymbol{\vartheta},t) = \sum_{n=1}^{N}\mathrm{e}^{\vartheta_n}\varphi_i(\boldsymbol{\vartheta},t)\,\lambda_n^{(i)}+\sum_{n=1}^{N}\sum_{n'=1}^{N}\mathrm{e}^{-\vartheta_n+\vartheta_{n'}}\frac{\partial}{\partial\vartheta_n}\varphi_i(\boldsymbol{\vartheta},t)\,\mu_{nn'}^{(i)}+$$
$$\sum_{n=1}^{N}\mathrm{e}^{-\vartheta_n}\frac{\partial}{\partial\vartheta_n}\varphi_i(\boldsymbol{\vartheta},t)\,\mu_{n0}^{(i)}+\sum_{j=1}^{I}\sum_{k=1}^{K_{ji}}\varphi_j\big((A_{ji}^{(k)})^{\mathrm{T}}\boldsymbol{\vartheta},t\big)\alpha_{ji}^{(k)}$$

and

$$\zeta_i(\boldsymbol{\vartheta},t) = \sum_{n=1}^{N}\varphi_i(\boldsymbol{\vartheta},t)\,\lambda_n^{(i)}+\sum_{n=1}^{N}\sum_{n'=1}^{N}\frac{\partial}{\partial\vartheta_n}\varphi_i(\boldsymbol{\vartheta},t)\,\mu_{nn'}^{(i)}+$$
$$\sum_{n=1}^{N}\frac{\partial}{\partial\vartheta_n}\varphi_i(\boldsymbol{\vartheta},t)\,\mu_{n0}^{(i)}+\sum_{j=1}^{I}\sum_{k=1}^{K_{ij}}\varphi_i(\boldsymbol{\vartheta},t)\alpha_{ij}^{(k)}.$$

We proceed in the common way: by subtracting $\varphi_i(\boldsymbol{\vartheta},t)$ from both sides in (2.1), dividing by Δt, and taking the limit as $\Delta t\downarrow 0$, we arrive at the following result.

Proposition 1. *The transforms $\varphi_i(\boldsymbol{\vartheta},t)$, for $i\in\mathscr{I}$, satisfy the system of partial differential equations:*

$$\frac{\partial}{\partial t}\varphi_i(\boldsymbol{\vartheta},t)=\varphi_i(\boldsymbol{\vartheta},t)\sum_{n=1}^{N}\big(e^{\vartheta_n}-1\big)\,\lambda_n^{(i)}+\sum_{n=1}^{N}\sum_{n'=1}^{N}\big(e^{-\vartheta_n+\vartheta_{n'}}-1\big)\frac{\partial}{\partial\vartheta_n}\varphi_i(\boldsymbol{\vartheta},t)\,\mu_{nn'}^{(i)}+$$
$$\sum_{n=1}^{N}\big(e^{-\vartheta_n}-1\big)\frac{\partial}{\partial\vartheta_n}\varphi_i(\boldsymbol{\vartheta},t)\,\mu_{n0}^{(i)}+\sum_{j=1}^{I}\sum_{k=1}^{K_{ji}}\varphi_j\big((A_{ji}^{(k)})^{T}\boldsymbol{\vartheta},t\big)\alpha_{ji}^{(k)}-\varphi_i(\boldsymbol{\vartheta},t)\sum_{j=1}^{I}\sum_{k=1}^{K_{ij}}\alpha_{ij}^{(k)}\,. \tag{2.2}$$

From this relation moments can be evaluated by differentiation and setting $\boldsymbol{\vartheta}=\mathbf{0}$, as we demonstrate in the next subsection.

2.2.3 Moments

We now find an explicit expression for the I mean queue content vectors (each of them in $\mathbb{R}_0^N$)

$$\overline{\boldsymbol{M}}_i(t) = \big[\mathbb{E}(M_n(t)I_i(t))\big]_{n=1}^N,$$

$i \in \mathscr{I}$. In addition to playing a central role in our performance evaluation framework, these expressions also allow us to establish the stability condition for this type of queueing network; see Section 2.2.4.

The first step is to identify the transient distribution of the environmental process $X(\cdot)$. To this end, we let $\pi_i(t) := \mathbb{P}(X(t) = i)$ for $i \in \mathscr{I}$; this means that $\boldsymbol{\pi}(t) = [\pi_i(t)]_{i=1}^I$ is the transient distribution vector of the background process. Setting $\boldsymbol{\vartheta} = \mathbf{0}$ in (2.2) yields a (homogeneous) system of coupled linear differential equations:

$$\pi_i'(t) \quad = \quad \sum_{j=1}^{I}\sum_{k=1}^{K_{ji}} \pi_j(t)\alpha_{ji}^{(k)} - \pi_i(t)\sum_{j=1}^{I}\sum_{k=1}^{K_{ij}} \alpha_{ij}^{(k)}\,, \quad i \in \mathscr{I}.$$

These are the Kolmogorov forward equations for the background process. They can be compactly rewritten as

$$\boldsymbol{\pi}'(t) = \overline{\mathscr{A}}^{\mathrm{T}}\boldsymbol{\pi}(t)$$

with $\overline{\mathscr{A}} = [\overline{\alpha}_{ij}]_{i,j=1}^I$ the corresponding generator matrix with elements, for $i, j \in \mathscr{I}$ and $i \neq j$,

$$\overline{\alpha}_{ij} = \sum_{k=1}^{K_{ij}} \alpha_{ij}^{(k)}\,, \quad \overline{\alpha}_{ii} = -\sum_{i' \neq i} \overline{\alpha}_{ii'}\,.$$

We thus find $\boldsymbol{\pi}(t) = \exp(\overline{\mathscr{A}}^T t)\boldsymbol{\pi}(0)$ (which, of course, also follows directly from the fact that $X(t)$ is a continuous-time Markov chain with transition rate matrix $\overline{\mathscr{A}}$). Observe that $\overline{\mathscr{A}}$ is a transition rate matrix (i.e., a matrix with negative diagonal elements and row sums equal to zero). This entails that, for any $t \geqslant 0$, $\boldsymbol{\pi}(t)$ is a probability distribution on $\mathscr{I}$.

Our next objective is to identify the first moments of the queue sizes. To obtain these quantities we differentiate the full equation (2.2) with respect to each of the ϑ_n ($n \in \mathscr{N}$). Plugging in $\boldsymbol{\vartheta} = \mathbf{0}$ then leads, after some straightforward but tedious algebra, to the (non-homogeneous) system of linear differential equations:

$$\overline{\boldsymbol{M}}_i'(t) = \mathscr{L}_i\pi_i(t) + \mathscr{M}_i\overline{\boldsymbol{M}}_i(t) + \sum_{j=1}^{I}\mathscr{A}_{ji}\overline{\boldsymbol{M}}_j(t)\,, \quad i \in \mathscr{I},$$

with the matrices $\mathscr{L}_i$, $\mathscr{M}_i$, and $\mathscr{A}_{ij}$ defined as follows.

- Firstly, $\mathscr{L}_i := \big[\lambda_n^{(i)}\big]_{n=1}^N$, i.e., a column vector with the arrival rates in the different queues when the background process is in state $i \in \mathscr{I}$.

- Secondly,

$$\mathscr{M}_i := \big[\mu_{n'n}^{(i)} + 1\{n = n'\}\overline{\mu}_n^{(i)}\big]_{n,n'=1}^N, \quad \text{with } \overline{\mu}_n^{(i)} = -\sum_{n'=0}^{N}\mu_{nn'}^{(i)},$$

is the matrix with the departure rates between the different queues when the background process is in state $i \in \mathscr{I}$.

- In addition, we introduce some notation for the multiplicative update process:

$$\mathscr{A}_{ij} := \sum_{k=1}^{K_{ij}} \alpha_{ij}^{(k)} A_{ij}^{(k)}, \quad \mathscr{A}_{ii} := \sum_{k=1}^{K_{ii}} \alpha_{ii}^{(k)} A_{ii}^{(k)} - \overline{\alpha}_i \mathbb{I}_N,$$

for $i, j \in \mathscr{I}$, $i \neq j$, with $\mathbb{I}_N$ denoting the $(N \times N)$-dimensional identity matrix, and with

$$\overline{\alpha}_i := \sum_{j=1}^{I} \sum_{k=1}^{K_{ij}} \alpha_{ij}^{(k)}.$$

Moreover, the above set of equations can be combined into a single equation. To this end, we define $\overline{\boldsymbol{M}}(t)$ to be the vector $[\overline{\boldsymbol{M}}_i(t)]_{i=1}^I$ of dimension $J := IN$. Also $\mathscr{A} := [\mathscr{A}_{ji}]_{i,j=1}^I$ and $\mathscr{M} := \operatorname{diag}([\mathscr{M}_i]_{i=1}^I)$, which are $(J \times J)$-dimensional matrices. Finally, $\mathscr{L} := \operatorname{diag}([\mathscr{L}_i]_{i=1}^I)$ is a matrix of dimension $J \times I$. We thus obtain the following representation of the set of differential equations; we refer to [42] where a similar method is used for a simpler (but related) model.

Proposition 2. *For any $t \geqslant 0$,*

$$\overline{\boldsymbol{M}}'(t) = \mathscr{L}\boldsymbol{\pi}(t) + (\mathscr{M} + \mathscr{A})\overline{\boldsymbol{M}}(t). \tag{2.3}$$

Solving the differential equation for the transient moment vector (2.3) leads to the explicit solution (in terms of integrals over matrix-exponentials):

$$\begin{aligned} \overline{\boldsymbol{M}}(t) &= \mathrm{e}^{(\mathscr{M}+\mathscr{A})t}\overline{\boldsymbol{M}}(0) + \int_0^t \mathrm{e}^{(\mathscr{M}+\mathscr{A})(t-s)}\mathscr{L}\boldsymbol{\pi}(s)\,\mathrm{d}s \\ &= \mathrm{e}^{(\mathscr{M}+\mathscr{A})t}\overline{\boldsymbol{M}}(0) + \int_0^t \mathrm{e}^{(\mathscr{M}+\mathscr{A})(t-s)}\mathscr{L}\,\mathrm{e}^{\overline{\mathscr{A}}^T s}\boldsymbol{\pi}(0)\,\mathrm{d}s. \end{aligned} \tag{2.4}$$

The stationary means follow from equating $\overline{\boldsymbol{M}}'(t)$ to $\mathbf{0}$ and defining $\boldsymbol{\pi}$ as the solution to $\overline{\mathscr{A}}^{\mathrm{T}}\boldsymbol{\pi} = \mathbf{0}$, so that the stationary mean $\overline{\boldsymbol{M}}$ is given by

$$\overline{\boldsymbol{M}} = -(\mathscr{M} + \mathscr{A})^{-1}\mathscr{L}\boldsymbol{\pi}. \tag{2.5}$$

Note, however, that this reasoning tacitly assumes that the underlying queueing network is stable, an issue we return to in Section 2.2.4.

Along the same lines higher moments of the queue sizes can be found as well. The higher transient moments can be phrased in terms of a (non-homogeneous) system of linear differential equations. The procedure to identify them is of a recursive nature, as determining the n-th transient moment requires knowledge of the first $n-1$ transient moments. Similarly, higher stationary moments follow as solutions to linear equations (under the stability condition), where finding the n-th stationary moment requires the first $n-1$ stationary moments being available. For analogous procedures in a related context, see [52].

2.2.4 Stability

As it turns out, Proposition 2 facilitates the provision of conditions for the ergodicity of the Markov chain. Before proceeding to stating and proving our stability result, we first define ω to be spectral abscissa of $\mathscr{M}+\mathscr{A}$, that is

$$\omega := \max\{\operatorname{Re}\lambda : \lambda \in \operatorname{spec}(\mathscr{M}+\mathscr{A})\}$$

where $\operatorname{spec}(\mathscr{M}+\mathscr{A})$ is the set of eigenvalues of $\mathscr{M}+\mathscr{A}$.

Theorem 4. *The Markov chain $\boldsymbol{Z}(\cdot) \equiv (\boldsymbol{M}(t), X(t))_{t\geq 0}$ is ergodic provided ω is negative.*

Proof. To prove the claim, we study the ergodicity of the *skeleton Markov chain* $\{\boldsymbol{Z}(\Delta n); n \in \mathbb{N}_0\}$ for some $\Delta > 0$. Obviously, if the skeleton Markov chain is ergodic for some $\Delta > 0$, so is $\boldsymbol{Z}(\cdot)$, as the mean recurrence time for any state of the skeleton chain is an upper bound for the mean recurrence time of the original chain $\boldsymbol{Z}(\cdot)$.

Appealing to [53, Proposition I.5.3], it suffices to show that for some $\varepsilon > 0$, $\Delta > 0$, and $\mathfrak{M} \geqslant 0$, the following mean drift condition holds:

$$\mathbb{E}_{(\boldsymbol{m},i)}\big(\|\boldsymbol{M}(\Delta)\|_1\big) - \|\boldsymbol{m}\|_1 < -\varepsilon$$

for all $\boldsymbol{m}$ with $\|\boldsymbol{m}\|_1 > \mathfrak{M}$ and all $i \in \mathscr{I}$; the subscript $(\boldsymbol{m}, i)$ indicates that the expectation is conditional on $\boldsymbol{Z}(0) = (\boldsymbol{m}, i)$.

Define $L := \sum_{i=1}^{I}\sum_{n=1}^{N}\lambda_n^{(i)}$. From the differential equation for the first moment (2.3), we derive the following bound:

$$\begin{aligned}
&\mathbb{E}_{(\boldsymbol{m},i)}\big(\|\boldsymbol{M}(\Delta)\|_1\big) - \|\boldsymbol{m}\|_1 \\
&= \big\|\overline{\boldsymbol{M}}(\Delta)|_{\boldsymbol{M}(0)=\boldsymbol{m},\boldsymbol{\pi}(0)=\boldsymbol{e}_i}\big\|_1 - \|\boldsymbol{m}\|_1 \\
&= \left\|\mathrm{e}^{(\mathscr{M}+\mathscr{A})\Delta}(\boldsymbol{e}_i\otimes\boldsymbol{m}) + \int_0^\Delta \mathrm{e}^{(\mathscr{M}+\mathscr{A})(\Delta-s)}\mathscr{L}\boldsymbol{\pi}(s)\mathrm{d}s\right\|_1 - \|\boldsymbol{m}\|_1 \\
&\leqslant \|\mathrm{e}^{(\mathscr{M}+\mathscr{A})\Delta}\|_1\|\boldsymbol{m}\|_1 + \int_0^\Delta \|\mathrm{e}^{(\mathscr{M}+\mathscr{A})(\Delta-s)}\|_1\, L\,\mathrm{d}s - \|\boldsymbol{m}\|_1 \\
&\leqslant \|\boldsymbol{m}\|_1\gamma\mathrm{e}^{\omega^\star\Delta} + \gamma\, L\int_0^\Delta \mathrm{e}^{\omega^\star(\Delta-s)}\mathrm{d}s - \|\boldsymbol{m}\|_1 \\
&\leqslant \|\boldsymbol{m}\|_1\gamma\mathrm{e}^{\omega^\star\Delta} - \frac{\gamma}{\omega^\star}\,L\,(1-\mathrm{e}^{\omega^\star\Delta}) - \|\boldsymbol{m}\|_1 =: g(\Delta)\,,
\end{aligned}$$

for $\omega < \omega^\star < 0$ and where $\gamma > 0$ is a constant; see [54, Proposition 11.18.8] for the bound on the norm of the matrix exponential. Clearly, with $R := \gamma L/\omega^\star < 0$,

$$\lim_{\Delta\to\infty} g(\Delta) = -R - \|\boldsymbol{m}\|_1,$$

which is negative for $\|\boldsymbol{m}\|_1 > -R > 0$. Hence, for any choice of $\mathfrak{M} > -R$, there exists a Δ such that the drift condition of the skeleton chain holds for $\|\boldsymbol{m}\|_1 > \mathfrak{M}$. The skeleton chain $\{\boldsymbol{Z}(\Delta n)\}_{n\in\mathbb{N}_0}$ is therefore ergodic, which is inherited by the original Markov process. □

Remark 3. At first sight it may look unnatural that the stability condition is in terms of the matrices $\mathscr{M}$ and $\mathscr{A}$ only, and does not involve the external arrival rate matrix $\mathscr{L}$. To

get a feel for the underlying intuition, let us consider the simplest network possible: an isolated infinite-server queue, with external arrival rate $\lambda \geqslant 0$, exponential holding times with mean $\mu^{-1} \geqslant 0$, and a multiplicative transition from state $m \in \mathbb{N}_0$ to $a_k m$ (with $a_k \in \mathbb{N}_0$) with rate $\alpha_k \geqslant 0$ $(k = 1, \ldots, K)$. Then, using the results of Section 2.2.3, the mean number of clients in the queue at time t, denoted by $\overline{M}(t)$, satisfies the differential equation

$$\overline{M}'(t) = \lambda + \sum_{k=1}^{K} \alpha_k (a_k - 1)\overline{M}(t) - \mu \overline{M}(t);$$

observe that the process goes up by one with rate λ, jumps from m to $a_k m$ (leading to a net change of $(a_k - 1)m$) with rate α_k, and goes down by one with rate μm. To ensure stability, the mean number in the system should not explode. This leads to a stability condition that does not involve λ, viz. (in self-evident vector notation) $\langle \boldsymbol{\alpha}, \boldsymbol{a} - \mathbf{1} \rangle < \mu$. $\Diamond$

2.2.5 Efficient evaluation of performance metrics

In many applications, the performance of the system during a finite time interval $[0, T]$ is expressed in terms of quantities of the form

$$v(T) := \sum_{i=1}^{I} \sum_{n=1}^{N} \varrho_{n,i} \overline{M}_{n,i}(T), \quad w(T) := \int_0^T \sum_{i=1}^{I} \sum_{n=1}^{N} \varrho_{n,i} \overline{M}_{n,i}(t) \mathrm{d}t,$$

for some vector $\boldsymbol{\varrho} \in \mathbb{R}^J$ and $T \geqslant 0$, with $\overline{M}_{n,i}(t) := \mathbb{E}\big(M_n(t) I_i(t)\big)$; we provide various examples of such performance metrics in Section 2.3. In this section we point out how to efficiently compute the vectors $\overline{\boldsymbol{M}}(T)$ and $\int_0^T \overline{\boldsymbol{M}}(t)\mathrm{d}t$.

We first study $\overline{\boldsymbol{M}}(T)$; note that $v(T)$ then follows upon evaluation of $\langle \boldsymbol{\varrho}, \overline{\boldsymbol{M}}(T) \rangle$. The first term of expression (2.4) is a matrix-exponential, for which standard evaluation techniques have been developed; see e.g. [55]. The second term reads $B(T) \cdot \boldsymbol{\pi}(0)$, with

$$B(T) := \int_0^T \mathrm{e}^{(\mathscr{M}+\mathscr{A})(T-s)} \mathscr{L} \mathrm{e}^{\overline{\mathscr{A}}^T s} \mathrm{d}s.$$

By [56, Thm. 1], $B(T)$ equals the $(J \times I)$-dimensional top right corner matrix of $\mathrm{e}^{\mathscr{C}T}$, where

$$\mathscr{C} := \begin{bmatrix} \mathscr{M} + \mathscr{A} & \mathscr{L} \\ \mathbb{O}_{I,J} & \overline{\mathscr{A}}^T \end{bmatrix}$$

(with $\mathbb{O}_{I,J}$ defined as an all-zeros matrix of dimension $I \times J$). We thus end up with the following observation for any $T \geqslant 0$:

$$\overline{\boldsymbol{M}}(T) = \mathrm{e}^{(\mathscr{M}+\mathscr{A})T} \overline{\boldsymbol{M}}(0) + \big[\mathbb{I}_J, \mathbb{O}_{J,I}\big] \cdot \mathrm{e}^{\mathscr{C}T} \cdot \begin{bmatrix} \mathbb{O}_{J,I} \\ \mathbb{I}_I \end{bmatrix} \boldsymbol{\pi}(0). \tag{2.6}$$

Now we explain how to evaluate $\int_0^T \overline{\boldsymbol{M}}(t)\mathrm{d}t$, which facilitates the computation of

$$w(T) = \int_0^T \langle \boldsymbol{\varrho}, \overline{\boldsymbol{M}}(t) \rangle \mathrm{d}t = \left\langle \boldsymbol{\varrho}, \int_0^T \overline{\boldsymbol{M}}(t)\mathrm{d}t \right\rangle.$$

Due to (2.6),

$$\int_0^T \overline{\boldsymbol{M}}(t)\mathrm{d}t = \int_0^T \mathrm{e}^{(\mathscr{M}+\mathscr{A})t}\mathrm{d}t \cdot \overline{\boldsymbol{M}}(0) + \left[\mathbb{I}_J, \mathbb{O}_{J,I}\right] \cdot \int_0^T \mathrm{e}^{\mathscr{C}t}\mathrm{d}t \cdot \begin{bmatrix} \mathbb{O}_{J,J} \\ \mathbb{I}_I \end{bmatrix} \boldsymbol{\pi}(0).$$

Define the matrices

$$\mathscr{C}_1 := \begin{bmatrix} \mathbb{O}_{J,J} & \mathbb{I}_J \\ \mathbb{O}_{J,J} & \mathscr{M}+\mathscr{A} \end{bmatrix}, \quad \mathscr{C}_2 := \begin{bmatrix} \mathbb{O}_{J+I,J+I} & \mathbb{I}_{J+I} \\ \mathbb{O}_{J+I,J+I} & \mathscr{C} \end{bmatrix},$$

which are of dimensions $2J \times 2J$ and $2(J+I) \times 2(J+I)$, respectively. Again applying [56, Thm. 1], we arrive at

$$\begin{aligned}\int_0^T \overline{\boldsymbol{M}}(t)\mathrm{d}t =& \left[\mathbb{I}_J, \mathbb{O}_{J,J}\right] \cdot \mathrm{e}^{\mathscr{C}_1 T} \cdot \begin{bmatrix} \mathbb{O}_{J,J} \\ \mathbb{I}_J \end{bmatrix} \boldsymbol{M}(0) + \\ & \left[\mathbb{I}_J, \mathbb{O}_{J,I}\right] \cdot \left[\mathbb{I}_{J+I}, \mathbb{O}_{J+I,J+I}\right] \cdot \mathrm{e}^{\mathscr{C}_2 T} \cdot \begin{bmatrix} \mathbb{O}_{J+I,J+I} \\ \mathbb{I}_{J+I} \end{bmatrix} \begin{bmatrix} \mathbb{O}_{J,J} \\ \mathbb{I}_I \end{bmatrix} \boldsymbol{\pi}(0).\end{aligned}$$

This can be rewritten in the following more compact form or any $T \geqslant 0$,

$$\int_0^T \overline{\boldsymbol{M}}(t)\mathrm{d}t = \left[\mathbb{I}_J, \mathbb{O}_{J,J}\right] \cdot \mathrm{e}^{\mathscr{C}_1 T} \cdot \begin{bmatrix} \mathbb{O}_{J,J} \\ \mathbb{I}_J \end{bmatrix} \boldsymbol{M}(0) + \left[\mathbb{I}_J, \mathbb{O}_{J,J+2I}\right] \cdot \mathrm{e}^{\mathscr{C}_2 T} \cdot \begin{bmatrix} \mathbb{O}_{2J+I,I} \\ \mathbb{I}_I \end{bmatrix} \boldsymbol{\pi}(0)\,. \tag{2.7}$$

2.3 Retrial queues, rerouting, storage systems

In this section we show the power of the framework introduced in the previous section, by pointing out how it facilitates the modelling of all sorts of relevant phenomena. We specifically focus on: (i) systems in which nodes go down but in which lost customers attempt re-entry, (ii) systems in which customers are rerouted when one of the links along the route goes down, and (iii) storage systems.

2.3.1 Retrial queues

In this subsection we consider a network of faulty service stations. Each of the stations alternates between being 'up' and 'down'. While a station is in the 'up' state it processes clients as a standard infinite-server queue. Upon going down, all clients present at a service station move instantly to an associated retrial location, from where they (independently of each other) try to re-enter the service station or renege. For an in-depth account of related retrial models, we refer to [57]. We note that, to the best of our knowledge, the literature does not cover the model we study here.

We now point out how this retrial mechanism fits in the framework that we set up in the previous section. Let the components 1 up to N° of $\boldsymbol{M}(\cdot)$ correspond to the service stations in the network, and the components $N^\circ+1$ up to $2N^\circ =: N$ to the associated retrial locations. Here we assume that the up-time of station $n \in \{1, \ldots, N^\circ\}$ is exponentially distributed with parameter $\gamma_n^{(\mathrm{u})}$, and the corresponding down-time is exponentially distributed with parameter $\gamma_n^{(\mathrm{d})}$. We thus have constructed an environmental process of dimension $I = 2^{N^\circ}$, where each state of this process corresponds to the particular set of

stations that are up (and consequently also the set of stations that are down). In the sequel we let $S(i)$ denote the set of stations that are up when the environmental process is in state i. (It is noted that the above model can be extended in a straightforward manner to the situation in which the up-times and down-times stem from phase-type distributions. Similarly, Markov-modulated arrivals can be dealt with.)

We let λ_n be the arrival rate at station n; note that clients arriving at station n when it is down are immediately placed in the corresponding retrial pool (which is component $N^\circ + n$ of $\boldsymbol{M}(\cdot)$). Also, let $\mu_{nn'}$ denote the rate of being routed (after service completion) from node n to node n' (with $n' = 0$ corresponding, as always, with the client leaving the network). The rate κ_n is the retrial rate at the n-th retrial location (i.e., component $N^\circ + n$ of $\boldsymbol{M}(\cdot)$), and ν_n the corresponding renege rate (reflecting clients that leave the network from a retrial location, i.e., before being served, e.g. due to impatience).

Let us now describe how the above parameters translate into the rates of the framework of the previous section. Suppose the environmental process is in state i. Let us first consider the external arrivals. Define $1_n(i) := 1\{n \in S(i)\}$. For $n = 1, \ldots, N^\circ$, the external arrival rates when the environmental process is in state i, are given by

$$\lambda_n^{(i)} = \lambda_n \, 1_n(i), \quad \lambda_{n+N^\circ}^{(i)} = \lambda_n \, (1 - 1_n(i)).$$

Regarding the service completions, we have for the service stations (with $n, n' = 1, \ldots, N^\circ$)

$$\mu_{nn'}^{(i)} = \mu_{nn'} \, 1_{n'}(i), \quad \mu_{n,n'+N^\circ}^{(i)} = \mu_{nn'} \, (1 - 1_{n'}(i)), \quad \mu_{n0}^{(i)} = \mu_{n0},$$

and for the retrial locations (again with $n = 1, \ldots, N^\circ$)

$$\mu_{n+N^\circ,n}^{(i)} = \kappa_n \, 1_n(i), \quad \mu_{n+N^\circ,0}^{(i)} = \nu_n.$$

We now consider the transitions related to the stations alternating between the active and inactive mode. As it turns out, for all $i, j \in \mathscr{I}$ we have that K_{ij} equals 0 or 1. We distinguish the cases:

- Suppose that for a $j \neq i$ and some $n \in \{1, \ldots, N^\circ\}$ we have $S(i) = S(j) \cup \{n\}$; then the background process jumps from i to j after an exponentially distributed time with rate $\alpha_{ij} = \gamma_n^{(\mathrm{u})}$. Note that this transition corresponds to station n failing, and thus clients being moved to the corresponding retrial location. The vector $\boldsymbol{M}(t)$ is pre-multiplied by a $(N \times N)$-dimensional matrix A_{ij} consisting of a 0 on position (n, n), a 1 on position $(n + N^\circ, n)$, all diagonal entries except the n-th being 1, and all other entries being 0.

- Suppose on the other hand that for $i \neq j$ and some $n \in \{1, \ldots, N^\circ\}$ we have $S(j) = S(i) \cup \{n\}$; then the background process jumps from i to j with rate $\alpha_{ij} = \gamma_n^{(\mathrm{d})}$, without the network population vector changing. This transition corresponds to station n having been repaired.

In the above cases $K_{ij} = 1$; for all other i, j we have $K_{ij} = 0$.

This framework has the potential to support various design issues. In the network described, an objective may be to keep the fraction of lost clients (due to reneging) below some tolerable level, say, ε. To this end, define $Z_a(t)$ as the total number of clients arrived

in $[0,t]$ and $Z_\ell(t)$ as the number of clients lost. With $\boldsymbol{\lambda}$ defined in the evident way,

$$\mathbb{E}\, Z_a(t) = \int_0^t \sum_{i=1}^{I} \sum_{n=1}^{N} \pi_i(s)\lambda_n^{(i)} \mathrm{d}s = \int_0^t \langle \boldsymbol{\lambda}, \overline{\boldsymbol{\pi}}(s)\rangle \mathrm{d}s.$$

Likewise, with $\boldsymbol{\nu}$ defined appropriately (i.e., a vector of which the first IN° entries equal to 0 and the second IN° entries equal the appropriate ν_n),

$$\mathbb{E}\, Z_\ell(t) = \int_0^t \sum_{i=1}^{I} \sum_{n=N^\circ+1}^{N} \mathbb{E}\big(M_n(s) I_i(s)\big)\nu_n \,\mathrm{d}s = \int_0^t \langle \boldsymbol{\nu}, \overline{\boldsymbol{M}}(s)\rangle \mathrm{d}s. \tag{2.8}$$

The numerical evaluation of the above performance metrics is facilitated by (2.7).

Suppose that for a given time horizon T the service requirement is $\mathbb{E}\, Z_\ell(T) \leqslant \varepsilon \cdot \mathbb{E}\, Z_a(T)$. If for given repair rates $\boldsymbol{\gamma}^{(\mathrm{d})} \equiv (\gamma_1^{(\mathrm{d})}, \ldots, \gamma_{N^\circ}^{(\mathrm{d})})$ this condition is not met, one may wonder by how much the repairs have to be sped up to meet the service requirement. A relevant optimization problem is then

$$\min_{\boldsymbol{\gamma}^{(\mathrm{d})}} \langle \boldsymbol{\gamma}^{(\mathrm{d})}, \mathbf{1}\rangle, \quad \text{subject to } \mathbb{E}\, Z_\ell(T) \leqslant \varepsilon \cdot \mathbb{E}\, Z_a(T).$$

2.3.2 Rerouting

Routing concerns the selection of a path along which traffic is transmitted. To make the service level more robust, one may adopt the policy that when a network element fails, traffic using that network element is routed along an alternative route. For a textbook treatment of routing in communication networks, we refer to e.g. [58].

Our present framework can be used to track the number of clients that use the different direct and indirect routes. The clients along these routes correspond to the customers of our framework and the queues (i.e., the components of $\boldsymbol{M}(\cdot)$) record the quantity of clients utilizing each of the direct and indirect routes. More formally, the rerouting model can be cast in our framework as follows. Let there be N° origin-destination pairs, each connected by a direct route (consisting of one link) as well as an indirect route (consisting of two links). Let the direct link used by the n-th origin-destination pair be labelled by n, and let $\mathscr{N}(n) := \{n_1(n), n_2(n)\}$ (both elements being contained in $\{1, \ldots, N^\circ\} \setminus \{n\}$) be the links of the corresponding indirect route. We thus have $N = 2N^\circ$ queues, the first N° queues corresponding to the number of clients on the direct routes and the second N° queues corresponding to the number of clients on the indirect routes. The parameters $\gamma_n^{(\mathrm{u})}$ and $\gamma_n^{(\mathrm{d})}$ correspond to the up-time and down-time of link n. Clients for origin-destination pair n arrive according to a Poisson process with rate λ_n, and stay in the system for an exponential time with parameter μ_n. We again stress that various generalizations are possible, such as phase-type up- and down-times and Markov modulated arrival processes; these extensions are conceptually very similar to the set-up we describe here, but notationally burdensome.

Each of the N° links can be up or down, so that the background process has $I = 2^{N^\circ}$ states. Let $S(i)$ be the set of links that are up when the background process is in state i. Again, define $1_n(i) := 1\{n \in S(i)\}$.

Suppose the background process is in state i. Then, for $n = 1, \ldots, N^\circ$,

$$\lambda_n^{(i)} = \lambda_n\, 1_n(i), \quad \lambda_{n+N^\circ}^{(i)} = \lambda_n\, (1 - 1_n(i))\, 1_{n_1(n)}(i)\, 1_{n_2(n)}(i).$$

All $\mu_{nn'} = 0$ for $n, n' \in \{1, \ldots, N\}$, and $\mu_{n0} = \mu_{n+N^\circ,0} = \mu_n$.

We now consider the transitions corresponding to links going down (and coming up again). For all $i, j \in \mathscr{I}$ we have that K_{ij} equals 0 or 1. We distinguish between the cases:

- Suppose that for a $j \neq i$ and some $n \in \{1, \ldots, N^\circ\}$ we have $S(i) = S(j) \cup \{n\}$; then the background process jumps from i to j after an exponentially distributed time with rate $\alpha_{ij} = \gamma_n^{(\mathrm{u})}$. Note that this transition corresponds to link n failing, and thus clients using this route as a direct route move to the indirect route (if available) and clients using this link as part of their indirect route are lost. The queue content vector is pre-multiplied by a $(N \times N)$-dimensional matrix A_{ij} consisting of (i) a 0 on position (n, n), (ii) a 1 on position $(n + N^\circ, n)$ but only if $\mathscr{N}(n) \subseteq S(i)$ (where it is noted that if $\mathscr{N}(n) \not\subseteq S(i)$, then files are lost), (iii) a 0 on position $(n' + N^\circ, n' + N^\circ)$ if $\{n\} \subseteq \mathscr{N}(n')$ (corresponding to files that are lost), (iv) all other diagonal entries being 1, and (v) all other entries being 0.

- Suppose on the other hand that for $i \neq j$ and some $n \in \{1, \ldots, N^\circ\}$ we have $S(j) = S(i) \cup \{n\}$; then the background process jumps from i to j with rate $\alpha_{ij} = \gamma_n^{(\mathrm{d})}$. This transition corresponds to link n having been repaired. The queue content vector is pre-multiplied by a $(N \times N)$-dimensional matrix A_{ij} consisting of (i) a 0 on position $(n + N^\circ, n + N^\circ)$, (ii) a 1 on position $(n, n + N^\circ)$, (iii) all other diagonal entries being 1, and (iv) all other entries being 0.

In the above cases $K_{ij} = 1$; for all other i, j we have $K_{ij} = 0$.

Again, our model can be used to study design questions. As indicated above, clients are lost if both the direct route and the indirect route are unavailable. Compared to using only direct routes, the option of indirect routes evidently reduces the number of lost clients, but this comes at the price of the servers being more intensively used. Let $Z_\ell(t)$ denote, as before, the number of clients lost in $[0, t]$; see Equation (2.8). In addition, let $Z_s(t)$ be the amount of link resources used in $[0, t]$:

$$\mathbb{E}\, Z_s(t) = \int_0^t \sum_{i=1}^{I} \sum_{n=1}^{N^\circ} \mathbb{E}\big(M_n(s) I_i(s)\big)\, \mathrm{d}s + 2 \int_0^t \sum_{i=1}^{I} \sum_{n=N^\circ+1}^{N} \mathbb{E}\big(M_n(s) I_i(s)\big)\, \mathrm{d}s;$$

again (2.7) can be used to numerically evaluate this quantity.

With a time horizon T, let ϱ_ℓ be the cost of losing a client and ϱ_s the per time unit network element utilisation cost, so that the total cost is

$$\varrho_\ell \cdot \mathbb{E}\, Z_\ell(T) + \varrho_s \cdot \mathbb{E}\, Z_s(T). \tag{2.9}$$

Its value can be compared to the value of the same objective function in the system without rerouting. Typically, for small $\varrho := \varrho_\ell / \varrho_s$ the system without rerouting is to be preferred, whereas for large ϱ rerouting pays off. To optimally design the system, it would be useful to have knowledge of the critical value $\varrho^\star$ (i.e., the value for which both mechanisms have the same cost).

2.3.3 Applications to storage networks

In storage networks information is typically stored at multiple locations (e.g. on multiple data storage servers), so as to mitigate the danger of files being lost. A relevant design issue concerns developing a policy that controls the fraction of files lost without unnecessarily

replicating them. For a general account of various aspects of storage networks, see e.g. [59].

Consider a system with K storage locations, each of which can be either 'up' or 'down'. Let the up-time of location $k \in \{1, \ldots, K\}$ be exponentially distributed with parameter $\gamma_k^{(\mathrm{u})}$, and let the corresponding down-time be exponentially distributed with parameter $\gamma_k^{(\mathrm{d})}$. We thus have constructed an environmental process of dimension $I = 2^K$, where each state corresponds to the set of locations that are up (and consequently also the set of locations that are down). We let, for any $i \in \{1, \ldots, I\}$, the set $U(i)$ denote the locations that are up when the environmental process is in state i. We order the I states such that the state 1 corresponds to all locations up, the states 2 up to $K+1$ to all situations with one location down, etc., so that state 2^K corresponds to all locations being down.

Files can be stored on any subset of the locations; there are $N = 2^K - 1$ of these. We let $S(n)$ denote the locations involved in the n-th subset, for $n \in \{1, \ldots, N\}$. These are ordered in the same way as above: queue 1 corresponds to files stored at all locations, the queues 2 up to $K+1$ to files stored at all-but-one locations, etc., so that queues $2^K - K$ up to $2^K - 1$ correspond to files stored on just a single location (which are lost if this location fails).

We now argue that this model is covered by the general multiplicative-transition framework that we introduced in the previous section. Consider the situation that the environmental process is in state i. Let λ_n be the (constant) arrival rate that is intended to be stored at the set of locations $S(n)$. However, if i is such that this is not possible (because some of the locations are down), it is only stored at the subset of $S(n)$ that is up. This means that, with $V(i,n) := \{n' : S(n') \cap U(i) = S(n)\}$, external arrivals to subset n occur at rate

$$\lambda_n^{(i)} = \sum_{n' \in V(i,n)} \lambda_{n'}.$$

During operations, files may be copied to additional locations, may be deleted from locations or may be deleted completely. Therefore, files hop from queue n to n' with rate $\mu_{nn'}^{(i)}$ (with $n' = 0$ corresponding to files leaving the network).

We now consider the multiplicative transitions. Two cases are to be distinguished.

- Suppose that for some $j \in \{1, \ldots, I\}$, that is assumed to be different from the current environmental state i, it holds that $U(i) = U(j) \cup \{k\}$; then the background process jumps from i to j after an exponentially distributed time with rate $\gamma_k^{(\mathrm{u})}$ (note that this transition corresponds to the event that location k finishes its up-time, i.e., goes down). Simultaneously the N-dimensional queue content vector is pre-multiplied by a matrix A_{ij} that is defined as follows. It has a zero on the diagonal positions that correspond to subsets that contain location k (i.e., n such that $\{k\} \subseteq S(n)$). In the same column, it has a one on the position n' such that $S(n') = S(n) \setminus \{k\}$ (if any).

- Suppose that for $i \neq j$ we have $U(j) = U(i) \cup \{k\}$; then the background process jumps from i to j with rate $\gamma_k^{(\mathrm{d})}$ (without any change in the network population vector; this transition corresponds to the event that location k finishes its down-time, i.e., becomes functioning again).

Recalling that the entries $2^K - K$ up to $2^K - 1$ of $\boldsymbol{M}(\cdot)$ correspond to files stored at just

a single location, we can evaluate the mean number of lost files in $[0,t]$ as

$$\mathbb{E}\, Z_\ell(t) = \int_0^t \sum_{i=1}^{I} \sum_{n=2^K-K}^{2^K-1} \mathbb{E}\big(M_n(s) I_i(s)\big) \gamma^{(u)}_{n-2^K+K+1} \mathrm{d}s,$$

which can be numerically evaluated using (2.7).

Consider for example the case of $K = 2$ locations, so that $I = 4$ and $N = 3$. In self-evident notation we code the 4 states of the background process as

$$\{1,2,3,4\} \equiv \{\{1,2\},\{1\},\{2\},\emptyset\}$$

(with the left-hand side in the previous display being in terms of the elements $i \in \mathscr{I}$, and the right-hand side in terms of the corresponding $U(i)$). Then

$$A_{12} = A_{34} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad A_{13} = A_{24} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \end{pmatrix},$$

whereas the other A-matrices equal I_3 (note that A_{12} and A_{34} correspond to location 2 going down, and A_{13} and A_{24} to location 1 going down). For all $i,j \in \mathscr{I}$ we have that K_{ij} equals 0 or 1. The transition rates α_{ij} are given by

$$\alpha_{12} = \alpha_{34} = \gamma_2^{(\mathrm{u})}, \quad \alpha_{13} = \alpha_{24} = \gamma_1^{(\mathrm{u})}, \quad \alpha_{21} = \alpha_{43} = \gamma_2^{(\mathrm{d})}, \quad \alpha_{31} = \alpha_{42} = \gamma_1^{(\mathrm{d})};$$

the corresponding values of K_{ij} are equal to 1 (whereas K_{ij} equals 0 for other i,j).

2.4 Numerical experiments

To illustrate the potential of our results, in this section we provide two numerical examples: one on a retrial queue, and another one on storage networks.

2.4.1 Retrial queue

In this first example, we consider a single retrial system, i.e., a two-queue network consisting of a service station and a retrial location. The service station alternates between being 'up' and 'down', with the corresponding durations being exponentially distributed with parameters $\gamma^{(u)}$ and $\gamma^{(d)}$, respectively. Clients arrive with rate λ and their service times are exponentially distributed with mean μ^{-1}. The rate at which customers in the retrial queue attempt to reenter service is κ, where the corresponding renege rate is ν.

We now cast this system in the terminology of our overarching model. The background process can be in two states (so that $I = 2$); we let state 1 correspond to the station being functioning, and state 2 to the station being down. The dimension of $\boldsymbol{M}(\cdot)$ is $N = 2$; the first component corresponds to the queue at the service station, whereas the second component corresponds to the retrial pool. The matrices A_{12} and A_{21} are given by

$$A_{12} = \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix}, \quad A_{21} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

The arrival rates are $\lambda_n^{(i)} = \lambda$ for (i,n) equalling $(1,1)$ or $(2,2)$, and otherwise 0. In addition, $\mu_{21}^{(1)} = \kappa$, $\mu_{20}^{(1)} = \mu_{20}^{(2)} = \nu$, $\mu_{10}^{(1)} = \mu$, whereas the other departure rates are 0. Also, $\alpha_{12} = \gamma^{(u)}$ and $\alpha_{21} = \gamma^{(d)}$.

Let $\overline{M}_{ni}(t)$ be the mean number in queue n when the background process is in state i at time t; observe that we constructed our model such that $\overline{M}_{12}(t) = 0$ for all $t \geqslant 0$. The time-dependent means follow from solving a system of linear differential equations:

$$\begin{aligned}
\overline{M}'_{11}(t) &= \lambda\pi_1(t) - (\mu + \gamma^{(u)})\overline{M}_{11}(t) + \kappa\overline{M}_{21}(t),\\
\overline{M}'_{21}(t) &= \gamma^{(d)}\overline{M}_{22}(t) - (\kappa + \nu + \gamma^{(u)})\overline{M}_{21}(t),\\
\overline{M}'_{22}(t) &= \lambda\pi_2(t) + \gamma^{(u)}\overline{M}_{11}(t) + \gamma^{(u)}\overline{M}_{21}(t) - (\nu + \gamma^{(d)})\overline{M}_{22}(t).
\end{aligned}$$

We now present the stationary means $\overline{M}_{11}$, $\overline{M}_{21}$, and $\overline{M}_{22}$. Let $\Gamma := \gamma^{(u)} + \gamma^{(d)}$, $\pi_1 = \gamma^{(d)}/\Gamma = 1 - \pi_2$. Sending $t \to \infty$, and letting the derivatives in the above differential equations be equal to 0, we obtain

$$\overline{M}_{21} = \frac{\lambda\,\gamma^{(u)}}{\Gamma\eta}\left(\frac{\mu + \gamma^{(u)} + \gamma^{(d)}}{\mu + \gamma^{(u)}}\right), \quad \eta := (\kappa + \nu + \gamma^{(u)})\frac{\nu + \gamma^{(d)}}{\gamma^{(d)}} - \kappa\frac{\gamma^{(u)}}{\mu + \gamma^{(u)}} - \gamma^{(u)},$$

and

$$\overline{M}_{11} = \frac{1}{\mu + \gamma^{(u)}}\left(\kappa\overline{M}_{21} + \lambda\frac{\gamma^{(\mathrm{d})}}{\Gamma}\right), \quad \overline{M}_{22} = \frac{\kappa + \nu + \gamma^{(u)}}{\gamma^{(d)}}\overline{M}_{21}.$$

We now consider the model's loss ratio ℓ, defined as the long-run fraction of clients leaving the network without being served (i.e., due to reneging). With $\overline{M}_{21}$ and $\overline{M}_{22}$ as computed above,

$$\ell = \frac{\nu}{\lambda}\big(\overline{M}_{21} + \overline{M}_{22}\big).$$

Experiment 1. To control the loss ratio, the service provider may opt for speeding up the repair times. The above formulas allow us to determine the smallest $\gamma^{(d)}$ such that the loss ratio ℓ is below some maximally allowed value $\ell^\star$; observe that ℓ is decreasing in $\gamma^{(d)}$. It requires elementary calculus to verify that

$$\lim_{\gamma^{(d)}\to\infty} \overline{M}_{21} = \frac{\lambda\gamma^{(\mathrm{u})}}{\kappa\mu + \nu\mu + \nu\gamma^{(u)}}, \quad \lim_{\gamma^{(d)}\to\infty} \overline{M}_{22} = 0,$$

so that

$$\lim_{\gamma^{(d)}\to\infty} \ell = \frac{\nu\gamma^{(\mathrm{u})}}{\kappa\mu + \nu\mu + \nu\gamma^{(u)}};$$

this expression increases in $\gamma^{(u)}$ and ν and decreases in μ and κ, as expected.

Observe that it in general cannot be guaranteed that there is a $\gamma^{(d)}$ such that $\ell \leqslant \ell^\star$: the parameters can be such that $\ell > \ell^\star$ for all $\gamma^{(d)}$. This is because even very short down-times lead to the event of clients simultaneously moving to the retrial queue, where the effect of clients reneging starts to kick in.

In the numerical experiment we chose $\lambda = 100$, $\kappa = 2$, $\nu = 2$, $\mu = 1$ and $\gamma^{(\mathrm{u})} = 0.1$. First suppose that the loss ratio should remain below 10%. One needs to take $\gamma^{(\mathrm{d})}$ larger than 2.1496, as illustrated by Fig. 2.1 (left panel). Suppose, on the contrary, that the target is 1%, then this cannot be achieved by increasing $\gamma^{(\mathrm{d})}$; based on the above results, we conclude that even by making the repairs very fast, the loss ratio will (for these values

of λ, κ, ν, μ and $\gamma^{(\mathrm{u})}$) never get below $0.2/4.2 \approx 4.76\%$ (corresponding to the horizontal dashed line in the graph).

Experiment 2. An alternative way to control ℓ is by making the up-times longer, i.e., by decreasing $\gamma^{(u)}$. It is readily verified that

$$\lim_{\gamma^{(u)} \downarrow 0} \overline{M}_{21} = \lim_{\gamma^{(u)} \downarrow 0} \overline{M}_{22} = 0,$$

so that the loss rate ℓ will be below any critical value $\ell^\star$ for $\gamma^{(u)}$ small enough.

In our numerical experiment we again chose $\lambda = 100$, $\kappa = 2$, $\nu = 2$, $\mu = 1$, but now we fix $\gamma^{(\mathrm{d})} = 0.5$. We wonder whether in this scenario a loss ratio below 1% can be achieved by tuning $\gamma^{(\mathrm{u})}$. Fig. 2.1 (right panel) shows that this is indeed the case: as it turns out, $\gamma^{(\mathrm{u})}$ should be below 0.0037.

In practice, one may want to find the most cost effective pair $(\gamma^{(u)}, \gamma^{(d)})$ such that the performance requirement is met. With $\varrho^{(u)}$ the cost of making the mean up-times one unit longer, and $\varrho^{(d)}$ the cost of making the hazard rate corresponding to the down-times one unit larger, a relevant optimization problem could be of the type

$$\min_{\gamma^{(u)}, \gamma^{(d)}} \frac{\varrho^{(u)}}{\gamma^{(u)}} + \varrho^{(d)} \gamma^{(d)}, \quad \text{subject to } \ell \leqslant \ell^\star.$$

2.4.2 A storage system

In this example we show how to analyse a specific storage system; it has some elements in common with the class of models that was introduced in Section 2.3.3, but there are a few notable differences. Files arrive according to a Poisson process with rate λ. With probability $1 - p$ the file will be offered 'basic service', and with probability p 'premium service'. A *basic file* is randomly allocated to one of the two locations (say A and B), where it will stay until that location goes 'down'. In our example copies of files are never deleted (except through a failure of the storage location). A *premium file* is randomly allocated to one of the locations, but is then copied at rate μ to the other location. When a location goes 'down' in the premium case, upon repair files are again copied back (also at rate μ, that is). The locations' up- and down-times are independent and statistically

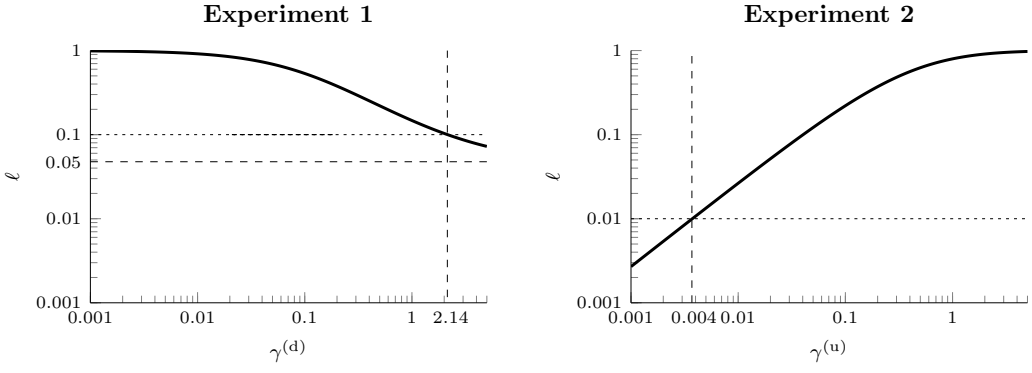


Figure 2.1: Retrial queue: loss ratio ℓ, Experiments 1 and 2.

identical; up-times (down-times, respectively) are exponentially distributed with rate $\gamma^{(u)}$ ($\gamma^{(d)}$). In this system there are five queues to be kept track of: premium files on location A, premium files on location B, premium files on locations A and B, basic files on location A, and basic files on location B.

Experiment 1. The parameters we picked are: $\lambda = 10\,000$ (i.e., on average 10 000 files arrive per day), $\mu = 24$ (i.e., it takes on average an hour before a stored file is copied to a second location), $\gamma^{(u)} = 0.01$ (i.e., each of the storage locations are functional on average for consecutive periods of 100 days), and $\gamma^{(d)} = 2$ (i.e., it takes 12 hours to repair a storage location). We let the system start empty at time 0, with both locations being 'up' (but other initial conditions are handled in the precise same way).

The first graphs, Figure 2.2, show, for $T = 1$ (i.e., one day), the expected number of lost files $\mathbb{E}\, Z_\ell(T)$, and the expected integral of the number of stored files, $\mathbb{E}\, Z_s(T)$, as functions of the fraction of premium files p. In the previous section we pointed out how these metrics can be evaluated, but the computation of $\mathbb{E}\, Z_\ell(T)$ can be done more efficiently, relying on the following idea; the performance measure $\mathbb{E}\, Z_s(T)$ can be dealt with analogously.

The idea is to append one coordinate to the state space; the resulting extra component $M_{N+1}(t)$ records the number of files lost in $[0, t]$ (which can be seen as a queue with zero departure rate). The transform of the vector $(\boldsymbol{M}(t), M_{N+1}(t)) \in \mathbb{N}_0^{N+1}$ (jointly with the state of the environmental process) can be characterized in the precise same way as that of just $\boldsymbol{M}(t)$, i.e., by setting up a system of partial differential equations. This provides us with an expression for $\mathbb{E}\, Z_\ell(T)$ of the form (2.4). Observe that it entails that we can evaluate the quantity $\mathbb{E}\, Z_\ell(T)$, which can be evaluated using (2.6); in this way we avoid evaluating integrals of the type of (2.8).

The graphs in Fig. 2.2 show, for $T = 1$, that $\mathbb{E}\, Z_s(T)$ increases in p (left panel), whereas $\mathbb{E}\, Z_\ell(T)$ decreases in p (right panel), as expected.

Experiment 2. We now consider a cost function that is a linear combination of $\mathbb{E}Z_\ell(t)$ and $\mathbb{E}Z_s(t)$, i.e., (7). In this case the optimal design amounts to minimizing the objective function (7) with respect to the fraction $p \in [0, 1]$. We denote the optimal p by $p^\star$. Let ϱ_ℓ and ϱ_s again respectively correspond to the cost of a lost file and the cost of a unit of storage per unit time; let $\varrho := \varrho_\ell/\varrho_s$. Clearly, $p^\star = 0$ for $\varrho \downarrow 0$ (as losing files is not penalized), whereas $p^\star = 1$ for $\varrho \uparrow \infty$ (as storing files is not penalized). Bearing in mind the shapes of $\mathbb{E}Z_\ell(t)$ and $\mathbb{E}Z_s(t)$, as depicted in Fig. 2.2, the optimization of a linear

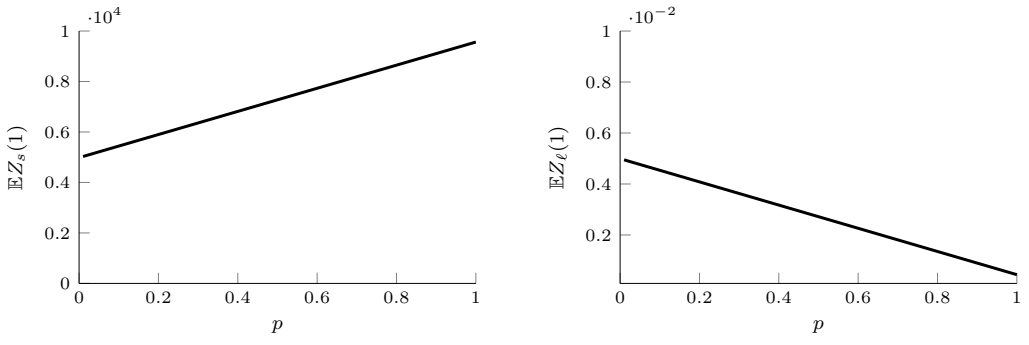


Figure 2.2: Storage system: $\mathbb{E}\, Z_s(1)$ and $\mathbb{E}\, Z_\ell(1)$ as functions of p, Experiment 1.

combination of $\mathbb{E}Z_\ell(t)$ and $\mathbb{E}Z_s(t)$ leads to $p^\star$ equalling either 0 or 1. The left panel of Fig. 2.3 shows the region in which the optimal $p^\star$ is 0 or 1, for combinations of $\gamma^{(\mathrm{d})}$ and ϱ; here ϱ_s is fixed (equal to one), and also $\gamma^{(\mathrm{u})} = 1$ (and all other parameters as in Experiment 1). In the right panel of Fig. 2.3 we show a similar picture, but now with $\gamma^{(\mathrm{u})}$ on the horizontal axis.

Experiment 3. We now vary the value of the repair rate $\gamma^{(\mathrm{d})}$ with the goal of achieving a predetermined performance target. For any value of p we compute the minimally required repair rate (defined as $\overline{\gamma}^{(\mathrm{d})}$) from $\gamma^{(\mathrm{d})} \in [0, 24]$, in an attempt to ensure that the loss fraction $\mathbb{E}Z_\ell(T)/(2\lambda)$ is below 0.05 (where we pick $T = 2$). Observe that the constraint $\overline{\gamma}^{(\mathrm{d})} \leqslant 24$ amounts to imposing the requirement that repairs must be expected to take at least 1 hour to perform.

Inspection of Fig. 2.4 immediately reveals that for p smaller than 0.5 we are unable to achieve our desired loss fraction using only the available changes in $\gamma^{(\mathrm{d})}$. Indeed it is conceivable that for small p there does not exist a repair rate such that the loss fraction goes below 0.05, a phenomenon similar to that which we earlier saw in Experiment 1 for the retrial queue. As p is increased within $[0, 0.5]$ we see an approximately linear decrease in the loss fraction resulting from the increased proportion of files being placed in the premium category (where they are unlikely to become lost). For $p \in [0.5, 0.8]$, we observe that we are able to achieve our desired loss fraction; moreover the storage location can be repaired increasingly slowly if more files are multiply stored (i.e., when p increases). This effect initially results in a very rapid decrease in the repair rate but has less of an impact as p is increased closer to 0.8, at which point the repair rate can no longer be traded off against increased duplication. Notice that the mechanism by which $\gamma^{(\mathrm{d})}$ decreases basic file losses is by reducing the portion of $[0, T]$ during which *both* storage locations are inoperable; this variable has no effect on basic files which are accepted into the system only to be lost due to a failure later. Hence, focusing on basic files, it can be seen that eventually the effect of $\gamma^{(\mathrm{d})}$ on the portion of $[0, T]$ for which both storage locations are inoperable becomes negligible compared to the reductions in losses from increasing p. The result of this is that for $p > 0.8$ the loss fraction continues to decrease approximately linearly as more files are placed in the very safe premium category, as we saw for $p < 0.5$.

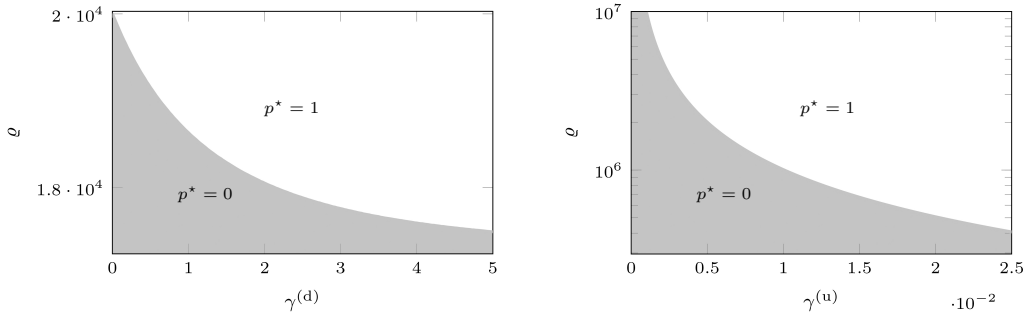


Figure 2.3: Storage system: areas in which $p^\star$ equals 0 and 1, for different values of the rates $\gamma^{(\mathrm{d})}$ and $\gamma^{(\mathrm{u})}$ on the horizontal axis and 'cost ratio' ϱ on the vertical axis, Experiment 2. The scale on the vertical axis is logarithmic.

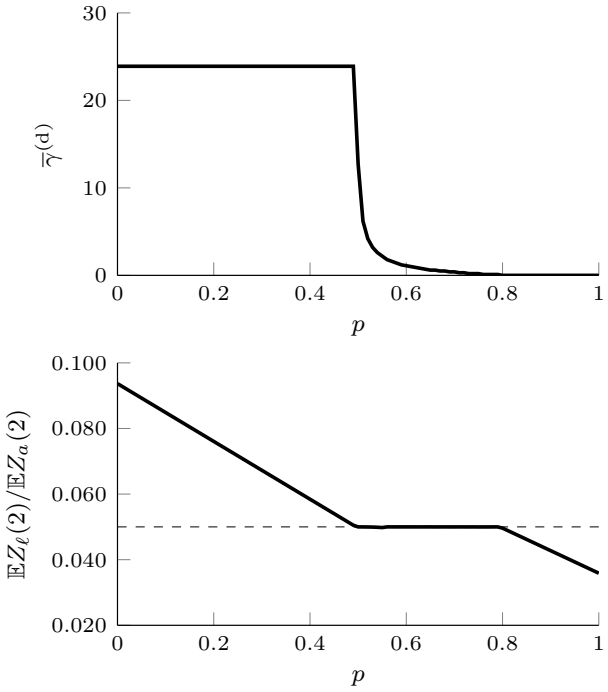


Figure 2.4: Storage system: $\gamma^{(d)}$ and the corresponding proportion of files lost for different values of the fraction p, Experiment 3.

2.5 Discussion and concluding remarks

In this chapter we studied a network of Markov-modulated infinite-server queues with the distinguishing feature that it also incorporates events by which the network population vector makes *multiplicative transitions* (at which it changes from $\boldsymbol{m}$ to $A\boldsymbol{m}$, for some matrix A). As we argued, the resulting framework covers various relevant models as special cases; for example, it enables the modelling of retrial queues, networks with rerouting, and storage systems.

Our results for the system's transient behaviour are in terms of (i) a system of partial differential equations describing the moment generating function of the network population vector, and (ii) a procedure to compute moments. In these expressions time t may approach ∞ so as to obtain the corresponding stationary behaviour, under the proviso that the stability condition applies.

Future research. The model we have developed triggers various intriguing research questions. In the first place, one may wonder whether under a specific scaling of the parameters one could find a weak limit for its transient or stationary behaviour. Such a procedure has been developed in [60, 52] for Markov-modulated infinite-server queues *without* multiplicative transitions. For that model the limiting process (after scaling the arrival rates and the environmental process) is a multivariate Ornstein-Uhlenbeck process. In this diffusion limit all limiting marginal distributions (and the model's stationary distribution, too) are asymptotically Normal. For our model however, *with* multiplicative

transitions that is, it is anticipated that there is no limiting process of diffusion type, due to the possibly large jumps caused by the multiplicative transitions; cf. [61]. In particular, the marginal distributions are expected to be asymmetric.

Scaling the external arrival rates by a common factor, say K, it is seen from (2.5) that the stationary mean also grows proportionally to K. Calling the stationary distribution under this scaling $\boldsymbol{M}^{(K)}$, one may want to asymptotically characterize large-deviation probabilities of the type

$$p_K := \mathbb{P}\left(\frac{\boldsymbol{M}^{(K)}}{K} \in S\right),$$

for K large and a set S that does not contain $\mathbb{E}\,\boldsymbol{M}^{(K)}/K = -(\mathscr{M}+\mathscr{A})^{-1}\mathscr{L}\boldsymbol{\pi}$. It is not clear how such asymptotics can be found; observe that due to the multiplicative transitions the model does not fit in the Freidlin-Wentzell framework [62], so that standard large-deviation techniques are likely to fail.

Other challenges lie in the application of our techniques to develop design principles for various sorts of operational networks. For instance for storage networks, one may want to develop an optimal replication policy, striking a proper balance between controlling the risk of files being lost and excessively using storage space.

CHAPTER 3

Stability of weighted queue-proportional rate allocation

3.1 Introduction

Multi-hop switched queueing networks are useful models for studying congestion and delay for wireless ad hoc networks, internet routers, call centres, data centres, and other complex communication systems. We consider a setting where jobs sequentially utilise stations in a network along fixed routes. There are constraints on which stations can serve jobs simultaneously, which evolve randomly throughout time. We consider a scheduling policy that makes decisions using only information on the number of jobs at each station.

In practice, a highly desirable fundamental feature of any scheduling policy is that, whenever possible, it results in bounded queue sizes over very long time horizons — a property called maximal stability. When the average rate at which jobs arrive to each station is known, implementing a random policy that allocates an average service rate to each queue that dominates the corresponding arrival rate is a simple way to achieve stability. More precisely the maximal property states that an equilibrium distribution will exist under a policy whenever it is possible to obtain an equilibrium distribution using the naive random policy just described that uses explicit knowledge of the arrival rate vector. For many networks of interest arrival rates are not known in advance, a problem which is often compounded by randomly varying rates.

The seminal work of Tassiulas and Ephremides [13] introduced the maximally stable BackPressure algorithm that makes a decision without using arrival rate information. In addition to stability, a key feature of this policy is that it utilises local information on queue sizes to make a scheduling decision. That is, in order to decide the priority of each route at each station it needs to know the number of jobs on that route at a neighbouring station. Therefore, the policy must maintain a queue for each route at each station. This may not be efficient, or even feasible, in many practical situations where the number of these queues can rapidly increase as the network grows. One benefit, however, is that it is possible to prioritise different stations in the network by incorporating weights into this set of information. This is useful, for example, when the penalties for longer queues are not homogeneous across stations in the network.

More recently introduced maximally stable algorithms include the ProportionalScheduler [63, 64] and Queue-Proportional Rate Allocation (QPRA) [16] policies. The primary

advantage that these policies have over BackPressure is that they do not distinguish between the types of jobs at each station and consequently scale much better with network size. However, these policies do not allow station priorities to be incorporated into the scheduling decision. In this chapter we present a natural generalisation of QPRA to allow station priorities.

Our policy, which we call Weighted QPRA (WQPRA), can be described roughly as follows: for a set of first-in, first-out stations $\mathcal{L}$, a vector of queue lengths $(Q_l,\, l \in \mathcal{L}) \in \mathbb{N}_0^{|\mathcal{L}|}$, a vector of station priorities $(\gamma_l,\, l \in \mathcal{L})$ where $\gamma_l \geq 1$ for all l, and a convex set of schedules $\overline{\mathbf{\Lambda}}^\star$, WQPRA chooses a mean per-station service vector $\boldsymbol{\sigma}$ from the boundary of $\overline{\mathbf{\Lambda}}^\star$ such that

$$\begin{aligned} &\sigma_l = 0\,, \quad \text{whenever } Q_l = 0\,, \\ &\frac{\sigma_l}{\gamma_l Q_l} = \frac{\sigma_{l'}}{\gamma_{l'} Q_{l'}}\,, \quad \text{whenever } Q_l > 0 \text{ and } Q_{l'} > 0\,. \end{aligned}$$

This corresponds to choosing $\boldsymbol{\sigma}$ proportional to $\boldsymbol{\gamma Q}$ (where this is element-wise multiplication of vectors). The service allocated to a station is then distributed uniformly at random between jobs (potentially of different routes) waiting for service at the station. Another generalisation is that we allow the set $\overline{\mathbf{\Lambda}}^\star$ to vary randomly over time to reflect, for example, random channel quality variations in the ad hoc network setting.

We prove that maximal stability holds for this policy by employing a fluid model analysis of the system. This generalises the work in [65]. The proof uses a Lyapunov function to determine important properties of the fluid limit sample paths, and the results of Dai [21].

For this type of network, heavily loaded settings are of high interest, since in these cases congestion is pervasive and enabling priorities between stations is paramount. Exact analysis of the network is unavailable and so it is natural to pursue a tractable approximation. To this end, we develop a diffusion limit approximation for the network following the arguments of Stolyar in [66].

For a set of routes using the network $\mathcal{R}$, let $X_{l,r}(t)$ be the number of jobs at station l of route r at time t. With arrival rate vector $(\lambda_r,\, r \in \mathcal{R})$ we call

$$Z(t) = \sum_{(l,r)} \frac{\lambda_r}{\gamma_l} X_{l,r}(t)$$

the workload of the system. In this ongoing work we conjecture that in heavily loaded settings the diffusion scaled workload process converges weakly to a reflected Brownian motion (RBM).

Another key feature of the limiting model is that the evolution of the queue lengths is restricted to a specific invariant manifold, a property known as state space collapse. Interestingly, the position of the queue length vector on the invariant manifold is determined by a RBM that describes the limiting value of the workload. In order to identify the invariant manifold we study the fluid model with critical arrival rate settings, combining the arguments of [16] and [66]. We are able to show weak convergence to Brownian motions for the arrival processes and the process governing the available scheduling policies using standard functional central limit theorem arguments. We then sketch our plans to use the attraction property of the fluid sample paths to the invariant model to solve several Skorohod problems and consequently obtain weak convergence of the workload process

to a reflected Brownian motion. Importantly, the location of the invariant manifold is a function of the station priority vector $\boldsymbol{\gamma}$, meaning that adjustments to priorities have a clear and strong impact on the evolution of the system. Notably, we also aim to show that the workload of the system under WQPRA stochastically dominates the workload under any other scheduling policy, a property called workload minimisation.

The rest of this chapter is organised as follows. In the next section we formally define the model. In Section 3.3 a formal definition of stability is provided. We then explicitly give the WQPRA algorithm in Section 3.4. Section 3.5 states the assumptions that our work operates under. In Section 3.6 we state a diffusion limit conjecture. More detail is added to the model and intuition behind the conjecture given in Section 3.7. In Section 3.8 we study the fluid sample paths of the model and prove the maximal stability property. Section 3.9 sketches a proof of our diffusion limit conjecture and Section 3.10 proves some technical supporting lemmas used in other parts of the chapter.

3.2 System model

We define multi-hop switched queueing networks operating in a random environment. The key feature of these discrete-time queueing networks is that they have restrictions on which queues can be served simultaneously. In our random environment setting these restrictions vary between time slots.

We assume that the network operates in slotted time. Each time slot is indexed by a non-negative integer $t \in \mathbb{N}_0$. Our theoretical results often rely on a continuous time version of the discrete time processes that we work with — in such cases we interpolate the discrete time process linearly between $\lfloor t \rfloor$ and $\lfloor t \rfloor + 1$, where $\lfloor t \rfloor$ denotes the largest integer no greater than t.

We let a finite countable set $\mathcal{L}$ index the set of stations forming a network. Assume that there are $L = |\mathcal{L}|$ stations present in the network and that there are a positive integer number $R \in \mathbb{Z}_+$ routes utilising the network. Each route consists of $N_r \in \mathbb{Z}_+$ consecutive indexed stations $r = \left(l_1^{(r)}, \ldots, l_{N_r}^{(r)}\right)$. For $k = 1, \ldots, N_r - 1$, a route r job served at station $l_k^{(r)}$ next goes to station $l_{k+1}^{(r)}$. Jobs served at the terminal station for their route $l_{N_r}^{(r)}$ depart the network. We denote by $l_-^{(r)}$ the previous (upstream) station to station $l^{(r)}$ on route r. We use $n_{l,r}$ to denote the index of station l on route r; that is, for station $l_i^{(r)}$ of route r, we have $n_{l,r} = i$. In general $n_{l,r} \in \{1, \ldots, N_r\}$. We denote the length of the longest route by $N^{\max} = \max_r\{N_r\}$.

Jobs of each route at each station are recorded in a separate queue of infinite buffer capacity. Let $\boldsymbol{\mathcal{S}}$ index this set of queues and let $S = |\boldsymbol{\mathcal{S}}|$. We call $(l, r) \in \mathcal{S}$ such that $n_{l,r} = 1$ ingress queues, and (l, r) such that $n_{l,r} > 1$ are internal queues. The process

$$(\boldsymbol{X}(t),\, t \in \mathbb{N}_0)$$

evolving on $\mathbb{N}_0^S$ keeps track of the lengths of the queues in the set $\boldsymbol{\mathcal{S}}$ and the process $(\boldsymbol{Q}(t),\ t \in \mathbb{N}_0)$ evolving on $\mathbb{N}_0^L$ keeps track of the total number of jobs queueing at each station. Specifically, at time t there are $X_{l,r}(t)$ jobs at station l on route r, moreover there are $Q_l(t)$ jobs at station l, thus

$$Q_l(t) = \sum_{r: l \in r} X_{l,r}(t)\,, \quad \forall\, t \geq 0\,.$$

Note that each queue indexed by an element of $\boldsymbol{\mathcal{S}}$ corresponds to a pair (l,r). Later, in order to state our main result, it will be convenient to consider the following partition of this set of queues. We group queues $(l,r) \in \boldsymbol{\mathcal{S}}$ into subsets $\boldsymbol{\mathcal{S}}_{i,j}$ which contain jobs that: (i) are being processed for the i-th time, and (ii) which exit the network after being processed by j stations. That is, we denote the set of all elements of $\boldsymbol{\mathcal{S}}$ such that $n_{l,r} = i$ and $N_r = j$ by $\boldsymbol{\mathcal{S}}_{i,j}$. Let $S_{i,j} = |\boldsymbol{\mathcal{S}}_{i,j}|$.

The processes

$$(\boldsymbol{X}_{i,j}(t),\, t \in \mathbb{N}_0)\,, \quad j = 1, \ldots, N^{\max}\,, \quad i = 1, \ldots, j\,,$$

evolving on $\mathbb{N}_0^{S_{i,j}}$ keep track of the lengths of the queues in the sets $\boldsymbol{\mathcal{S}}_{i,j}$.

At the end of each time slot, a discrete random number $A_r(t)$ of route r jobs arrive exogenously to station $l_1^{(r)}$, where $(A_r(t),\, t \geq 0)$ are independent and identically distributed (iid), and these processes are independent across routes. For each queue $(l,r) \in \boldsymbol{\mathcal{S}}$ denote $\lambda_r = \mathbb{E}A_r(1)$, which is the expected number of jobs arriving to route r in each time slot. We store these quantities in a vector of length S denoted $\boldsymbol{\lambda}$. (Note that many elements of this vector are identical.) It is convenient to partition these arrival rates according to the sets $\boldsymbol{\mathcal{S}}_{1,j}$. For $j = 1, 2, \ldots, N^{\max}$, denote by $\boldsymbol{\lambda}_j$ the vector of length $S_{1,j}$ which contains the arrival rates of routes which have length j.

The processing restrictions of the network are random and follow a countable, irreducible, finite state, discrete time background Markov chain $(M(t),\, t \in \mathbb{N}_0)$ evolving on $\mathcal{M}$. In each time slot this Markov chain is in one of the states $m \in \mathcal{M}$. The stationary distribution of M is denoted $(\pi_m,\, m \in \mathcal{M})$, where

$$\pi_m > 0,\, \forall\, m \in \mathcal{M} \quad \text{and} \quad \sum_{m \in \mathcal{M}} \pi_m = 1\,.$$

Associated with each state of the background chain is a finite countable set of feasible station schedules $K(m)$. Each feasible station schedule is a set of stations that can process jobs simultaneously. Let $\boldsymbol{\Lambda}(m)$ denote the convex hull of $K(m)$ (note that since sets of feasible schedules are finite, these convex sets are necessarily compact). We suppose that for each time t it is possible to specify a *station scheduling policy* in the form of a distribution on $K(M(t))$ conditional on $M(t) = m$. The resulting station scheduling policy is then represented by a random vector with support on $K(m)$, of length L, denoted

$$\boldsymbol{R}(t,m) := (R_l(t,m)\,,\, l \in \boldsymbol{\mathcal{L}})\,,$$

where $R_l(t,m) = 1$ if station l is scheduled for processing in time slot t and $R_l(t,m) = 0$ otherwise.

For ease of exposition we assume that each station can process at most one job in each time slot. For each queue (l,r), let the random variable $R_{l,r}(t,m)$ indicate if a job is scheduled for processing from that queue at time t. Notice that $\sum_{r:l \in r} R_{l,r}(t,m) = R_l(t,m)$, and this is either 0 or 1. In the same way that a station scheduling policy is a random distribution on $K(m)$, we also suppose that it is possible to specify a queue scheduling policy as a distribution for the random vector $(R_{l,r}(t,m)\,,\, l \in \boldsymbol{\mathcal{S}})$. For WQPRA this distribution is conditional on $\boldsymbol{R}(t,m)$, but this is not necessarily the case (e.g., BackPressure [13] and static service split policies as described in the next section).

Let $D_{l,r}(t) \leq X_{l,r}(t)$ be the number of jobs departing from queue (l,r) in time slot t. For all queues, if $R_{l,r}(t,m) = 1$ and $X_{l,r}(t) \geq 1$, then $D_{l,r}(t) = 1$; otherwise $D_{l,r}(t) = 0$.

At the end of each time slot, jobs which have been served either move to the next station along their route or depart the system if they are at the terminal station for their route. As a result, the queueing dynamics of this network can be described by the recurrence relation

$$X_{l,r}(t+1) = \begin{cases} X_{l,r}(t) + A_r(t) - D_{l,r}(t)\,, & n_{l,r} = 1\,, \\ X_{l,r}(t) + D_{l^{(r)}_-,r}(t) - D_{l,r}(t)\,, & n_{l,r} > 1\,, \end{cases}$$

for any pair (l,r) and $t \geq 0$.

3.3 System stability region

We call the system *stable* if the Markov chain $\boldsymbol{X}$ is positive recurrent. By this we mean that there exists a non-empty set of states which is reached with probability 1 within finite expected time from any initial state. Stability implies the existence of a stationary probability distribution.

Suppose $\phi := (\phi_m,\ m \in \mathcal{M})$ is a collection with $\phi_m := (\phi_{m,(l,r)},\ (l,r) \in \boldsymbol{\mathcal{S}})$ being a probability measure such that when the background chain is in state m, $R_{l,r}(t,m) = 1$ with probability $\phi_{m,(l,r)}$, and with probability $1 - \sum_{(l,r)} \phi_{m,(l,r)}$ does not serve any of the queues. This is known as a static service split policy since it is independent of information such as the size of the queue length vector. Let the expected service allocated to queue (l,r) when the background chain is in state m under this policy be denoted by $\mu_{l,r}(t,m)$, so that the long-term service rate allocated to queue (l,r) is

$$\nu_{l,r}(\phi) = \sum_{m \in \mathcal{M}} \pi_m\, \mu_{l,r}(t,m)\,.$$

Let $\boldsymbol{\nu}(\phi) = (\nu_{l,r}(\phi),\ (l,r) \in \mathcal{S})$.

Proposition 4. *For existence of a scheduling policy Φ under which the system is stable, condition*

$$\boldsymbol{\lambda} \leq \boldsymbol{\nu}(\phi) \quad \textit{for at least one SSS rule } \phi \tag{3.1}$$

is necessary, and condition

$$\boldsymbol{\lambda} < \boldsymbol{\nu}(\phi) \quad \textit{for at least one SSS rule } \phi \tag{3.2}$$

is sufficient.

The proof of this proposition is a simple extension to the multihop setting of the proof given for Theorem 1 in [67]. For necessity, if the system is stable under some policy ϕ, then under that policy $\phi_{m,(l,r)}$ is the average fraction of time slots when queue (l,r) is allocated while the background state is m. These values form the set for which (3.1) must hold, since otherwise at least one of the queues would diverge with probability 1. Sufficiency follows simply from observing that (3.2) implies the long term average service rate is greater than the long term average arrival rate for all queues.

Motivated by Proposition 4 we call the set of all arrival rate vectors $\boldsymbol{\lambda}$ such that (3.2) holds the *maximal stability region* or *maximum throughput region* and denote it by $\boldsymbol{\Lambda}^0$. We denote the closure of this set by $\boldsymbol{\Lambda}^\star$, and call this the *capacity region*. At some points in this chapter it is necessary to denote the maximal stability region in terms of the per-station aggregate arrival rate vector $\left(\sum_{r:l\in r} \lambda_r\right)_{l\in\mathcal{L}}$. To this end, let $\overline{\boldsymbol{\Lambda}}^0$ be the

set of aggregate arrival vectors $\left(\sum_{r:l\in r}\lambda_r\right)_{l\in\mathcal{L}}$ such that (3.2) holds for the corresponding underlying arrival rate vector $\boldsymbol{\lambda}$, and denote the closure of this set by $\overline{\boldsymbol{\Lambda}^\star}$.

3.4 Weighted queue-proportional rate allocation scheduler

In this section we present a natural generalisation of the throughput-optimal scheduling algorithm of [16]. Our generalisation is to weight stations so that priorities between stations can also be incorporated into the transmission decision. We denote the weight of station l by $\gamma_l \geq 1$ and store these weights in $\boldsymbol{\gamma}$, a vector of length L. In [16] the case $\gamma_l = 1$ for all $l \in \mathcal{L}$ is studied.

Given $\boldsymbol{Q}(t)$, $M(t)$, and weights $\boldsymbol{\gamma}$:

(i) First determine a station scheduling policy for time slot t by finding the vector $\boldsymbol{\sigma}(t, m)$ of length L such that

$$\begin{aligned}&\sigma_l(t, M(t)) = 0\,, \quad \text{whenever } Q_l(t) = 0\,,\\ &\frac{\sigma_l(t, M(t))}{\gamma_l\, Q_l(t)} = \frac{\sigma_{l'}(t, M(t))}{\gamma_{l'} Q_{l'}(t)}\,, \quad \text{whenever } Q_l(t) > 0 \text{ and } Q_{l'}(t) > 0\,,\end{aligned}$$

where $\boldsymbol{\sigma}(t, M(t))$ lies on the boundary of $\boldsymbol{\Lambda}(M(t))$. Let the distribution of $\boldsymbol{R}(t, m)$ have expected value $\boldsymbol{\sigma}(t, m)$ and support on $K(M(t))$.

(ii) Serve jobs at each station according to the Serve-In-Random-Order (SIRO) queueing discipline, i.e., serve jobs at each station l uniformly at random. This implies that the average departure rate of each route at station l is proportional to the number of jobs of its route, i.e.,

$$\begin{aligned}&\mu_{l,r}(t, m) = 0\,, \quad \text{whenever } X_{l,r}(t) = 0\,,\\ &\frac{\mu_{l,r}(t, m)}{X_{l,r}(t)} = \frac{\mu_{l,r'}(t, m)}{X_{l,r'}(t)}\,, \quad \text{whenever } X_{l,r}(t) > 0 \text{ and } X_{l,r'}(t) > 0\,.\end{aligned}$$

3.5 Key assumptions

Consider a sequence of systems, indexed by $k \in \mathcal{K} = \{k_1, k_2, \dots\}$, where $k_i > 0$ for all i and $k_i \uparrow \infty$ as $i \to \infty$. For the remainder of this chapter $k \to \infty$, unless specified otherwise, means that k goes to infinity by taking values from the sequence $\mathcal{K}$, or some subsequence of $\mathcal{K}$. We denote quantities pertaining to the k-th system with a superscript (k).

Our assumption of iid input processes also applies to every system in this sequence, that is

$$A_r^{(k)}(t)\,, \quad t = 1, 2, \dots, \quad \text{are iid.} \tag{3.3}$$

We also make some additional assumptions on the input flows that allow us to apply a functional central limit theorem (FCLT) later. They are:

$$\operatorname{Var} A_r^{(k)}(1) \;\to\; s_r^2 \geq 0\,, \quad k \to \infty\,, \quad \text{and} \tag{3.4}$$

$$\mathbb{E}[A_r^{(k)}(1)^2\, \mathrm{I}\{A_r^{(k)}(1) > z\}] \leq \eta(z)\,, \tag{3.5}$$

where η is a fixed function such that $\eta(z) \to 0$ as $z \to \infty$, and $\mathrm{I}\{B\}$ is the indicator function for the event B.

Define $\boldsymbol{\zeta}$ to be a vector of length S with each component corresponding to a queue $(l,r) \in \boldsymbol{\mathcal{S}}$ and taking the value λ_r/γ_l. We define $\boldsymbol{\zeta}_{i,j}$ similarly for each $\boldsymbol{\mathcal{S}}_{i,j}$. Then, to conclude this section, for $t \geq 0$, let

$$Z(t) := \boldsymbol{\zeta} \cdot \boldsymbol{X}(t) = \sum_{(l,r)\in\boldsymbol{\mathcal{S}}} \frac{\lambda_r}{\gamma_l} X_{l,r}(t) \quad \text{and} \quad Z_{i,j}(t) := \boldsymbol{\zeta}_{i,j} \cdot \boldsymbol{X}_{i,j}(t) = \sum_{(l,r)\in\boldsymbol{\mathcal{S}}_{i,j}} \frac{\lambda_r}{\gamma_l} X_{l,r}(t)\,.$$

The former is a process that keeps track of the *workload* of the system, and the latter keeps track of the workload in each of the sets $\boldsymbol{\mathcal{S}}_{i,j}$.

Define $\boldsymbol{\mu}_{i,j}$ to be a vector of length $S_{i,j}$ containing the elements of $\boldsymbol{\mu}$ corresponding to the set of queues $\boldsymbol{\mathcal{S}}_{i,j}$.

We are now able to introduce the following function of the background chain state:

$$\mu^\star_{i,j}(m) = \max_{\boldsymbol{\mu}_{i,j}\in\boldsymbol{\Lambda}(m)} \boldsymbol{\zeta}_{i,j} \cdot \boldsymbol{\mu}_{i,j}\,, \tag{3.6}$$

where we abuse notation slightly by taking $\boldsymbol{\mu}_{i,j} \in \boldsymbol{\Lambda}(m)$ to mean a choice of $\boldsymbol{\mu} \in \boldsymbol{\Lambda}(m)$ restricted to the queues contained in $\boldsymbol{\mathcal{S}}_{i,j}$. This function gives the maximum possible amount of workload that could potentially be served in one time slot by the sets of queues $\boldsymbol{\mathcal{S}}_{i,j}$ when the background chain is in state m.

Using this function we can further let

$$\mu^\star_{i,j} = \sum_{m\in\mathcal{M}} \pi_m\, \boldsymbol{\mu}^\star_{i,j}(m)\,,$$

be the maximum average possible service rates available to workload for these sets of queues.

Now, our heavy traffic assumptions are as follows: we suppose that, as $k \to \infty$, the arrival rate vector $\boldsymbol{\lambda}^{(k)}$ converges to some fixed vector with strictly positive elements $\boldsymbol{\lambda} \in \mathbb{R}^S_+$, lying on the boundary of the capacity region $\boldsymbol{\Lambda}^\star$:

$$\boldsymbol{\lambda}^{(k)} \to \boldsymbol{\lambda} \in \boldsymbol{\Lambda}^\star\,. \tag{3.7}$$

In addition we suppose that the convergence (3.7) is such that

$$k\,(\boldsymbol{\zeta}_{1,j} \cdot \boldsymbol{\lambda}^{(k)}_j - \boldsymbol{\zeta}_{1,j} \cdot \boldsymbol{\lambda}_j - \mu^\star_{1,j}) \to a_j\,, \tag{3.8}$$

where $a_j \in \mathbb{R}$ for $j = 1, \ldots, N^{\max}$.

The processes $\boldsymbol{X}^{(k)}$, $\boldsymbol{Q}^{(k)}$, and $Z^{(k)}$ and their equivalents restricted to $\boldsymbol{\mathcal{S}}_{i,j}$ record queue lengths and workload in the network when the arrival rate vector is $\boldsymbol{\lambda}^{(k)}$.

Finally, we also make a *no-excess-resources* (NER) assumption:

$$\mu^\star_{i,j} - \mu^\star_{i-1,j} = 0\,, \qquad\qquad j = 2, \ldots, N^{\max}, \quad i = 2, \ldots, j\,. \tag{3.9}$$

The NER assumption states that the maximum average workload that can be processed by queues in $\boldsymbol{\mathcal{S}}_{i,j}$ and queues in $\boldsymbol{\mathcal{S}}_{i-1,j}$ is the same. This assumption ensures that the near congestion behaviour created at ingress stations in heavy traffic is conveyed to queues which are deeper in the network, which is essential for our convergence results to hold.

3.6 Diffusion limit

We begin by applying *diffusion scaling* to the processes $\boldsymbol{X}^{(k)}$, $\boldsymbol{Q}^{(k)}$, and $Z^{(k)}$:

$$\widetilde{\boldsymbol{x}}^{(k)}(t) := k^{-1}\boldsymbol{X}^{(k)}(k^2 t)\,, \quad \widetilde{\boldsymbol{q}}^{(k)}(t) := k^{-1}\boldsymbol{Q}^{(k)}(k^2 t)\,, \quad \widetilde{z}^{(k)}(t) := k^{-1}Z^{(k)}(k^2 t)\,, \quad t \geq 0\,. \tag{3.10}$$

We call the set of vectors $(c\,\boldsymbol{\zeta},\, c \geq 0)$, which are proportional to $\boldsymbol{\zeta}$, the *invariant manifold*, and call the elements of this set *invariant points*. Denote the invariant manifold by $\mathcal{V}$. We provide a more formal definition of the invariant manifold in Section 3.8. We assume that the initial states of the scaled processes converge in probability to an invariant point

$$\widetilde{\boldsymbol{x}}^{(k)}(0) \xrightarrow{\mathsf{p}} \widetilde{\boldsymbol{x}}(0) := \widetilde{z}^{(k)}(0)\,\boldsymbol{\zeta}\,, \tag{3.11}$$

where $\xrightarrow{\mathsf{p}}$ denotes convergence in probability.

We also apply diffusion scaling to $\boldsymbol{X}_{i,j}^{(k)}$ and $Z_{i,j}^{(k)}$, for $j = 1, \ldots, N^{\max}$ and $i = 1, \ldots, j$:

$$\widetilde{\boldsymbol{x}}_{i,j}^{(k)} := k^{-1}\boldsymbol{X}_{i,j}^{(k)}(k^2 t)\,, \quad \widetilde{z}_{i,j}^{(k)}(t) := k^{-1}Z_{i,j}^{(k)}(k^2 t)\,, \quad t \geq 0\,.$$

For $j = 1, \ldots, N^{\max}$ define the Brownian motions

$$\widetilde{w}_{1,j} = \big(\widetilde{z}_{1,j}^{(k)}(0) + a_j\, t + \varrho_{1,j}\, B(t),\, t \geq 0\big)\,, \tag{3.12}$$

all driven by a standard (zero drift, diffusion coefficient) Brownian motion B, where a_j is the parameter in (3.8), and

$$\varrho_{1,j}^2 := \widehat{s}_{1,j}^2 + \sum_{\boldsymbol{\mathcal{S}}_{1,j}} \zeta_{l,r}^2\, s_r^2\,,$$

where s_r^2 is the parameter given in (3.4) and $\widehat{s}_{1,j}^2$ depends on $\boldsymbol{\zeta}$ and the stationary distribution of the (background) Markov chain M, and is specified later in (3.27). These processes correspond to workload entering the system at ingress stations.

For the Brownian motion processes $\widetilde{w}_{1,j}$ defined in (3.12), define the corresponding reflected Brownian motions (RBMs)

$$\widetilde{z}_{1,j}(t) := \widetilde{w}_{1,j}(t) + \widetilde{y}_{1,j}(t)\,, \quad t \geq 0\,, \tag{3.13}$$

where

$$\widetilde{y}_{1,j}(t) := -\left[0 \wedge \inf_{0 \leq u \leq t} \widetilde{w}_{1,j}(u)\right]\,, \quad t \geq 0\,, \tag{3.14}$$

are regulation processes. These processes correspond to workload service 'wasted' by queues in the sets $\boldsymbol{\mathcal{S}}_{1,j}$. In Section 3.9 we sketch what this means more precisely.

Proceeding, we introduce some analogous relationships for $\widetilde{z}_{i,j} = \widetilde{w}_{i,j} + \widetilde{y}_{i,j}$ with $j = 2, \ldots, N^{\max}$ and $i = 2, \ldots, j$. Define

$$\widetilde{w}_{i,j} = \big(\widetilde{z}_{i,j}(0) + \widetilde{y}_{i-1,j}(t) + \widehat{s}_{i,j}^2\, B(t),\, t \geq 0\big)\,, \tag{3.15}$$

where B is the *same* Brownian motion used in (3.12), with corresponding regulation processes

$$\widetilde{y}_{i,j}(t) := -\left[0 \wedge \inf_{0 \leq u \leq t} \widetilde{w}_{i,j}(u)\right]\,, \quad t \geq 0\,. \tag{3.16}$$

Here the processes $\widetilde{w}_{i,j}$ correspond to workload which is output from queues in $\boldsymbol{\mathcal{S}}_{i-1,j}$ as input to queues in $\boldsymbol{\mathcal{S}}_{i,j}$, and $\widetilde{y}_{i,j}$ remains to correspond to workload service 'wasted' by queues in $\boldsymbol{\mathcal{S}}_{i,j}$. The parameter $\widetilde{s}^2_{i,j}$ also depends on the stationary distribution of the (background) Markov chain M, and is specified later in (3.34).

For $1 \le j \le N^{\max}$ and $1 \le i \le j$ we will sketch a proof in Section 3.9 that

$$\widetilde{z}^{(k)}_{i,j} \overset{\text{w}}{\to} \widetilde{z}_{i,j} = \widetilde{w}_{i,j} + \widetilde{y}_{i,j}\,, \tag{3.17}$$

where $\overset{\text{w}}{\to}$ denotes convergence in distribution of stochastic processes, that is, weak convergence of their distributions. Upon bringing all of these processes together as

$$\widetilde{z} := \sum_{j=1}^{N^{\max}} \sum_{i=1}^{j} \widetilde{z}_{i,j}\,, \tag{3.18}$$

we obtain the driving process for our *diffusion limit* and *state space collapse* conjecture, which we can now state.

Conjecture 5. *Consider the sequence of systems indexed by $k \in \mathcal{K}$ as described above, where the scheduling rule for the system is WQPRA. Assume that conditions* (3.3)–(3.5), (3.7)–(3.9), *and* (3.11) *hold.*

(a) Then, as $k \to \infty$,

$$\widetilde{z}^{(k)} \overset{\text{w}}{\to} \widetilde{z}\,, \tag{3.19}$$

as described in (3.13)–(3.18). *Moreover, the following state space collapse holds:*

$$\widetilde{\boldsymbol{x}}^{(k)} \overset{\text{w}}{\to} \widetilde{\boldsymbol{x}} := \widetilde{z}\,\boldsymbol{\zeta}\,. \tag{3.20}$$

(b) The WQPRA rule is asymptotically optimal in that it minimizes the workload process. More precisely, the workload process $\widetilde{z}^k_\Phi$ corresponding to an arbitrary scheduling discipline Φ is such that, for any time $t \ge 0$ and any $u \ge 0$,

$$\liminf_{k\to\infty} \mathbb{P}(\widetilde{z}^k_\Phi(t) > u) \ge \mathbb{P}(\widetilde{z}(t) > u)\,. \tag{3.21}$$

3.7 Additional system model detail and intuition behind main conjecture

In this section we define several additional stochastic processes associated with the system for each value of the scaling parameter k. These can be used to prove our main conjecture. Combined with the previously defined model, the process

$$\boldsymbol{\mathcal{Z}}^{(k)} := (\boldsymbol{X}^{(k)}, \boldsymbol{Q}^{(k)}, Z^{(k)}, W^{(k)}, Y^{(k)}, \overline{\boldsymbol{A}}^{(k)}, \overline{\boldsymbol{D}}^{(k)}, \boldsymbol{G}^{(k)})$$

and its counterparts corresponding to restrictions of station–route pairs in some $\boldsymbol{\mathcal{S}}_{i,j}$, with many of its components summarised in Table 3.1 and all defined shortly, describe the system evolution in more detail.

We defined the processes $\boldsymbol{X}^{(k)}$, $\boldsymbol{Q}^{(k)}$, and $Z^{(k)}$ previously. The process $(A^{(k)}_r(t),\, t \ge 0)$ describes the arrivals in each time slot to route r. The cumulative version of this process

Table 3.1: Descriptions of some components of $\boldsymbol{\mathcal{Z}}^{(k)}$.

Component	Brief description
$\boldsymbol{X}^{(k)}$	$(X_{l,r}^{(k)},\ (l,r) \in \boldsymbol{\mathcal{S}})$ Lengths of station–route queues.
$\boldsymbol{Q}^{(k)}$	$(Q_l^{(k)},\ l \in \mathcal{L})$ Lengths of per-station queues.
$Z_{i,j}^{(k)}$	Workload in system at queues in $\boldsymbol{\mathcal{S}}_{i,j}$.
$Z^{(k)}$	Total workload in system.
$Y_{i,j}^{(k)}$	Cumulative wasted workload service at queues in $\boldsymbol{\mathcal{S}}_{i,j}$.
$Y^{(k)}$	Cumulative total wasted workload service.
$\overline{A}_r^{(k)}$	Cumulative arrivals to route r.
$\overline{\boldsymbol{A}}_j^{(k)}$	$(A_r^{(k)},\ (l,r) \in \boldsymbol{\mathcal{S}}_{1,j})$
$\overline{\boldsymbol{D}}^{(k)}$	$(\overline{D}_{l,r}^{(k)},\ (l,r) \in \boldsymbol{\mathcal{S}})$ Cumulative departures from station–route queues.
$\overline{\boldsymbol{D}}_{i,j}^{(k)}$	$(D_{l,r}^{(k)},\ (l,r) \in \boldsymbol{\mathcal{S}}_{i,j})$.
$H_{i,j}^{(k)}$	Cumulative potential workload service for queues in $\boldsymbol{\mathcal{S}}_{i,j}$.
$\boldsymbol{G}^{(k)}$	$(G_m^{(k)},\ m \in \mathcal{M})$ Cumulative time background process in state m.

is

$$\overline{A}_r^{(k)} := \left(\sum_{\tau=0}^{t} A_r^{(k)}(\tau)\,,\ t \geq 0 \right),$$

which records the total arrivals up to and including time slot t. We denote the diffusion scaling of these processes by

$$\widetilde{a}_r^{(k)}(t) := k^{-1} \overline{A}^{(k)}(k^2 t)\,, \quad t \geq 0\,.$$

We will find it convenient to group these according to the index of the terminal station of the route (i.e., into the sets $\boldsymbol{\mathcal{S}}_{1,j}$, $j = 1, \ldots, N^{\max}$),

$$\overline{\boldsymbol{A}}_j^{(k)} := (A_r^{(k)},\ (l,r) \in \boldsymbol{\mathcal{S}}_{1,j})\,.$$

Similarly, $D_{l,r}^{(k)}(t)$ is the number of packets to be served from queue (l,r) in time slot t, so that

$$\overline{D}_{l,r}^{(k)} := \left(\sum_{\tau=0}^{t} D_{l,r}^{(k)}(\tau)\,,\ t \geq 0 \right),$$

which we also group into $\overline{\boldsymbol{D}}_{i,j}^{(k)} := (D_{l,r}^{(k)},\ (l,r) \in \boldsymbol{\mathcal{S}}_{i,j})$.

The processes

$$G_m^{(k)} := \left(\sum_{\tau=0}^{t} 1\{M(\tau) = m\},\ t \geq 0 \right), \quad m \in \mathcal{M}\,,$$

record the total number of time slots the system spends in each of the different processing modes $m \in \mathcal{M}$. We combine these with the maximum possible rates at which workload

can be served for the set of queues $\boldsymbol{\mathcal{S}}_{i,j}$, defined earlier in (3.6), and denoted by $\mu^\star_{i,j}(m)$, as follows:

$$H^{(k)}_{i,j} := \left(\sum_{m \in \mathcal{M}} \mu^\star_{i,j}(m) G^{(k)}_m(t),\, t \geq 0 \right). \tag{3.22}$$

We denote the diffusion scaling of these processes by

$$\widetilde{h}^{(k)}_{i,j}(t) := k^{-1} H^{(k)}_{i,j}(k^2 t)\,, \quad t \geq 0\,.$$

For $i = 1$, and $j = 1, \ldots, N^{\max}$ these processes give the potential amount of workload that could be served by time t by queues in $\boldsymbol{\mathcal{S}}_{1,j}$. For $j = 1$ this corresponds to the workload of single station routes and for $j > 1$ this corresponds to workload that progresses onwards to receive more service.

Importantly, the processes

$$W^{(k)}_{1,j}(t) := Z^{(k)}_{1,j}(0) + \boldsymbol{\zeta}_{1,j} \cdot \overline{\boldsymbol{A}}^{(k)}_j(t) - H^{(k)}_{1,j}(t)\,, \quad t \geq 0\,, \tag{3.23}$$

do not depend on the scheduling discipline employed — they are a function of the initial workload and model primitives only. We apply diffusion scaling to these processes:

$$\widetilde{w}^{(k)}_{1,j}(t) := k^{-1} W^{(k)}_{1,j}(k^2 t)\,, \quad t \geq 0\,.$$

We will next see that, following from standard theory, the resulting processes converge weakly to a Brownian motion as defined in (3.12), i.e.,

$$\widetilde{w}^{(k)}_{1,j} \overset{\text{w}}{\to} \widetilde{w}_{1,j}\,. \tag{3.24}$$

Due to assumptions (3.3)–(3.4) we can employ a standard FCLT for each input process (i.e., per-route input into queues in $\boldsymbol{\mathcal{S}}_{1,j}$ for each j):

$$\left(\widetilde{a}^{(k)}_r(t) - \lambda^{(k)}_r k t,\, t \geq 0 \right) \overset{\text{w}}{\to} (s_r\, B(t),\, t \geq 0)\,, \tag{3.25}$$

where B is a standard (zero drift, diffusion coefficient) Brownian motion. In a similar fashion, from the FCLT for Markov chains, we have the weak convergence

$$\left(\widetilde{h}^{(k)}_{1,j}(t) - \mu^\star_{1,j} k t,\, t \geq 0 \right) \overset{\text{w}}{\to} (\varrho_{1,j}\, B(t),\, t \geq 0)\,, \tag{3.26}$$

where

$$\varrho^2_{i,j} := \lim_{n \to \infty} n^{-1} \mathbb{E}\left[\left(\sum_{t=0}^{n} \mu^\star_{i,j}(M(t)) \cdot \boldsymbol{\zeta}_{i,j} - \mu^\star_{i,j} t \right)^2 \right]. \tag{3.27}$$

We can combine these two limit results with assumption (3.8) to obtain $\widetilde{w}^{(k)}_{1,j} \overset{\text{w}}{\to} \widetilde{w}_{1,j}$ as defined in (3.12).

This brings us to the first non-trivial part of our analysis. For each $t \geq 0$ we are now able to define

$$Y^{(k)}_{1,j}(t) := H^{(k)}_{1,j}(t) - \boldsymbol{\zeta}_{1,j} \cdot \overline{\boldsymbol{D}}^{(k)}_{1,j}(t)\,, \tag{3.28}$$

which is the total amount of workload service that is *wasted* by queues in $\boldsymbol{\mathcal{S}}_{1,j}$ by time t.

We also apply diffusion scaling to this process:

$$\widetilde{y}_{1,j}^{(k)}(t) := k^{-1} Y_{1,j}^{(k)}(k^2 t)\,, \quad t \geq 0\,. \tag{3.29}$$

Combining (3.23) and (3.28) with the definition of $Z_{1,j}^{(k)}(t)$ given earlier, observe the relationship

$$Z_{1,j}^{(k)}(t) = W_{1,j}^{(k)}(t) + Y_{1,j}^{(k)}(t)\,, \quad t \geq 0\,, \tag{3.30}$$

and futhermore $\widetilde{z}_{1,j}^{(k)} = \widetilde{w}_{1,j}^{(k)} + \widetilde{y}_{1,j}^{(k)}$. Note we already found an expression for the limiting values of $\widetilde{w}_{1,j}^{(k)}$; now we need to do the same for $\widetilde{y}_{1,j}^{(k)}$. In fact, we will show that $\widetilde{y}_{1,j}^{(k)}$ converges weakly to the regulation process $\widetilde{y}_{1,j}$ defined in (3.14). We do this in Section 3.9 by solving a Skorohod problem (see Appendix A.1) that applies to every possible sample path of the process. The intuition behind (3.12)—(3.14) should now be clear.

We now provide intuition for (3.15)–(3.17), this intuition is similar to that just provided for (3.12)–(3.14).

For $j = 2, \ldots, N^{\max}$ and $i = 2, \ldots, j$ we have

$$W_{i,j}^{(k)}(t) := Z_{i,j}^{(k)}(0) + H_{i-1,j}^{(k)}(t) - \overline{Y}_{i-1,j}^{(k)}(t) - H_{i,j}^{(k)}(t)\,, \quad t \geq 0\,. \tag{3.31}$$

These processes are analogous to (3.23) with the exogenous workload arrival processes $\boldsymbol{\zeta}_{1,j} \cdot \boldsymbol{A}_j^{(k)}$ replaced by the workload output from $\boldsymbol{\mathcal{S}}_{i-1,j}$ processes $H_{i-1,j}^{(k)} - \overline{Y}_{i-1,j}^{(k)} \equiv \boldsymbol{\zeta}_{i-1,j} \cdot \overline{\boldsymbol{D}}_{i-1,j}^{(k)}$.

Again, observe the relationships

$$Z_{i,j}^{(k)}(t) = W_{i,j}^{(k)}(t) + Y_{i,j}^{(k)}(t)\,, \quad t \geq 0\,. \tag{3.32}$$

The diffusion scalings of (3.31) and (3.32) are denoted using $\widetilde{w}_{i,j}^{(k)}$ and $\widetilde{z}_{i,j}^{(k)}$.

Using the FCLT we will find

$$\left(\widetilde{h}_{i-1,j}^{(k)}(t) - \widetilde{h}_{i,j}^{(k)}(t),\, t \geq 0\right) \overset{\mathrm{w}}{\to} \left(\widehat{\varrho}_{i,j}\, B(t),\, t \geq 0\right), \tag{3.33}$$

where

$$\widehat{\varrho}_{i,j}^2 := \varrho_{i-1,j}^2 + \varrho_{i,j}^2 + 2 \lim_{n\to\infty} n^{-1} \mathbb{E}\left[\left(\sum_{t=0}^{n} \mu_{i-1,j}^{\star}(M(t)) \cdot \boldsymbol{\zeta}_{i-1,j} - \mu_{i-1,j}^{\star} t\right)\left(\sum_{t=0}^{n} \mu_{i,j}^{\star}(M(t)) \cdot \boldsymbol{\zeta}_{i,j} - \mu_{i,j}^{\star} t\right)\right], \tag{3.34}$$

(with $\varrho_{i,j}^2$ corresponding to the variance terms defined in (3.27)) follows from the standard formula for variances of sums of dependent random variables. From this we will obtain $\widetilde{w}_{i,j}^{(k)} \overset{\mathrm{w}}{\to} \widetilde{w}_{i,j}$ as defined in (3.15). We prove formally in the next section that the corresponding workload departures are given by (3.16).

The preceding discussion was about how to show $\widetilde{z}_{i,j}^{(k)} \overset{\mathrm{w}}{\to} \widetilde{z}_{i,j}$ for $j = 1, \ldots, N^{\max}$, $i = 1, \ldots, j$ with an explicit method for finding for $\widetilde{z}_{i,j}$ so that $\widetilde{z}^{(k)} \overset{\mathrm{w}}{\to} \widetilde{z}$ (as defined in (3.18)) can be found, leading to (3.19), which is the first part of Conjecture 5. Some more detail is needed to rigorously obtain the complete statement of Conjecture 5, (3.19) and (3.20), we do this in Section 3.9. The sketch proof provided in Section 3.9 relies on the

fluid sample paths of our model, which we will now investigate in detail.

3.8 Some properties of fluid sample paths for the WQPRA scheduler

In this section we study sequences of the process $\boldsymbol{\mathcal{Z}}^{(k)}$ under *fluid scaling*. That is, in this section we consider a sequence of systems $\{\boldsymbol{z}^{(k)}, k \in \mathcal{K}_f\}$ where $\mathcal{K}_f$ is a non-decreasing sequence of positive numbers, where:

$$\boldsymbol{z}^{(k)}(t) := k^{-1}\boldsymbol{\mathcal{Z}}^{(k)}(kt)\,, \quad t \geq 0\,.$$

Importantly, the sequences $\mathcal{K}_f$ and those used for the heavy-traffic limit and diffusion scaling $\mathcal{K}$ are not necessarily related.

In particular this scaling applies to all of the processes described in Table 3.1. We denote the fluid scaled processes with lower case letters, for example, for $\boldsymbol{X}^{(k)}$, $\boldsymbol{Q}^{(k)}$, and $Z^{(k)}$:

$$\boldsymbol{x}^{(k)}(t) := k^{-1}\boldsymbol{X}^{(k)}(kt)\,, \quad \boldsymbol{q}^{(k)}(t) := k^{-1}\boldsymbol{Q}^{(k)}(kt), \quad z^{(k)}(t) := k^{-1}Z^{(k)}(kt)\,, \quad t \geq 0\,. \tag{3.35}$$

The following lemma gives the system of equations that these processes follow for k large enough.

Lemma 6. *Under the WQPRA algorithm, with probability 1, there exists a positive sequence $\mathcal{K}_f$ such that as $k \to \infty$ along $\mathcal{K}_f$ the process $\boldsymbol{z}^{(k)}$ and all of its constituent processes converge uniformly over compact intervals, where the limiting functions are Lipschitz continuous in $[0,\infty)$, to a fluid sample path $\boldsymbol{z}$.*

Specifically, we have for $j = 1,\ldots,N^{\max}$, $i = 1,\ldots,j$, $r \in \mathcal{R}$, and $m \in \mathcal{M}$:

$$\boldsymbol{x}^{(k)}(t) \to \boldsymbol{x}(t)\,, \quad \boldsymbol{q}^{(k)}(t) \to \boldsymbol{q}(t)\,, \quad \overline{a}_r^{(k)}(t) \to \lambda_r\, t\,, \quad g_m^{(k)}(t) \to \pi_m\, t\,, \qquad t \geq 0\,, \tag{3.36}$$

$$h_{i,j}^{(k)}(t) \to \mu_{i,j}^{\star}\, t\,, \quad w^{(k)}(t) \to z(0) + \boldsymbol{\zeta}\cdot\boldsymbol{\lambda}\, t - h(t) = z(0) = w(0), \qquad t \geq 0\,, \tag{3.37}$$

$$w_{i,j}^{(k)}(t) \to z_{i,j}(0) + \boldsymbol{\zeta}_{i,j}\cdot\boldsymbol{\lambda}_j\, t - h_{i,j}(t) = z_{i,j}(0) = w_{i,j}(0)\,, \qquad t \geq 0\,, \tag{3.38}$$

$$z^{(k)}(t) \to z(0) + \boldsymbol{\zeta}\cdot\boldsymbol{x}(t) + y(t) = z(0) + y(t)\,, \qquad t \geq 0\,, \tag{3.39}$$

$$z_{i,j}^{(k)}(t) \to z_{i,j}(0) + \boldsymbol{\zeta}_{i,j}\cdot\boldsymbol{x}_{i,j}(t) + y_{i,j}(t) = z_{i,j}(0) + y_{i,j}(t)\,, \qquad t \geq 0\,. \tag{3.40}$$

Lipschitz continuity of the limiting functions implies they are differentiable for almost all $t \geq 0$. Let $\mathcal{T}$ be the set of time instances where these functions are differentiable. Additionally, the following equations hold for all $t \in \mathcal{T}$:

$$\begin{aligned} q_l(t) &= \sum_{r:l\in r} x_{l,r}(t), \quad \forall\, l \in \mathcal{L}\,, \\ \frac{\mathrm{d}}{\mathrm{d}t}x_{l,r}(t) &= \lambda_{l,r} + \mu_{l_-^{(r)},r}(t) - \mu_{l,r}(t)\,, \quad \forall\,(l,r) \in \boldsymbol{\mathcal{S}}\,, \end{aligned} \tag{3.41}$$

where $\lambda_{l,r} = \lambda_r$ *for* $n_{l^{(r)},r} = 1$ *and* 0 *otherwise,* $\boldsymbol{\mu}(t)$ *satisfies*

$$\begin{aligned} \mu_{l^{(r)},r}(t) &= 0, \text{ for } n_{l^{(r)},r} = 1\,, \\ \mu_{l,r}(t) &= \frac{x_{l,r}(t)}{q_l(t)}\sigma_l(t)\,, \end{aligned} \tag{3.42}$$

and $\sigma(t)$ *lies on the boundary of the capacity region* $\overline{\boldsymbol{\Lambda}}^\star$ *satisfying*

$$\begin{aligned} &\sigma_l(t) = 0\,, \text{ whenever } q_l(t) = 0\,, \\ &\frac{\sigma_l(t)}{\gamma_l q_l(t)} = \frac{\sigma_{l'}(t)}{\gamma_{l'} q_{l'}(t)}\,, \text{ whenever } q_l(t) > 0 \text{ and } q_{l'}(t) > 0\,. \end{aligned} \tag{3.43}$$

The proof of this lemma is standard and straightforward, yet somewhat technical; a proof following techniques developed in [68, 21, 69] and used in [16] is given in Section 3.10. We often omit the time index in the remainder of this chapter for brevity.

Notice from this we can more formally define the invariant manifold. Observe that the differential equations defined by (3.41)–(3.43) are stationary when $\mu_{l,r} = \lambda_r$ for all l, and so we define the invariant manifold $\mathcal{V}$ by $\boldsymbol{x}$ such that

$$\begin{aligned} x_{l,r} &= \lambda_r \frac{q_l}{(\boldsymbol{\lambda}_\Sigma)_l}\,, && \forall\,(l,r) \in \mathcal{S}\,, \in \boldsymbol{\mathcal{S}} \\ \frac{q_l\gamma_l}{(\boldsymbol{\lambda}_\Sigma)_l} &= \frac{q_{l'}\gamma_{l'}}{(\boldsymbol{\lambda}_\Sigma)_{l'}}\,, && \forall\,(l,r),\,(l',r') \in \boldsymbol{\mathcal{S}}\,. \end{aligned}$$

The following lemma generalises Lemma 3 in [16] to the WQPRA setting.

Lemma 7. *Assume* $\theta\left(\sum_{r:l\in r}\lambda_r\right)_{l\in\mathcal{L}} \in \overline{\boldsymbol{\Lambda}}^\star$ *for some* $\theta > 0$. *If* $\boldsymbol{x} \neq 0$ *and* $(l^*, r^*) \in \arg\max_{(l,r)} \gamma_l x_{l,r}/\lambda_r$, *then* $\mu_{l^*,r^*} \geq \theta\lambda_{r^*}$.

Proof. Since $\boldsymbol{x} \neq 0$, we have $x_{l^*,r^*} > 0$. Assume $\mu_{l^*,r^*} < \theta\lambda_{r^*}$. Then, for any other $(l,r) \in \boldsymbol{\mathcal{S}}$ ($r \neq r^*$ or $l \neq l^*$), there are two cases:

(i) If $x_{l,r} = 0$, then $\mu_{l,r} = 0$.

(ii) If $x_{l,r} > 0$, then

$$\mu_{l,r} = \frac{\gamma_l x_{l,r}}{\gamma_{l^*} x_{l^*,r^*}}\mu_{l^*,r^*} = \frac{\gamma_l x_{l,r}/\lambda_r}{\gamma_{l^*} x_{l^*,r^*}/\lambda_{r^*}}\frac{\lambda_r}{\lambda_{r^*}}\mu_{l^*,r^*} \leq \frac{\lambda_r}{\lambda_{r^*}}\mu_{l^*,r^*} < \theta\lambda_r\,.$$

Combining (i) and (ii) we have $\mu_{l,r} < \theta\lambda_r$ for any pair (l,r). Therefore, for each link l, if $q_l > 0$ then we have $\sigma_l = \sum_{r:l\in r}\mu_{l,r} < \theta\sum_{r:l\in r}\lambda_r$; if $q_l = 0$, then $\sigma_l = 0 < \theta\sum_{r:l\in r}\lambda_r$. Hence we have $\sigma_l < \theta\sum_{r:l\in r}\lambda_r$ for each station $l \in \mathcal{L}$. This contradicts the fact that the allocated service rate vector $(\sigma_l)_{l\in\mathcal{L}}$ lies on the boundary of the capacity region $\overline{\boldsymbol{\Lambda}}^\star$, since $\theta\left(\sum_{r:l\in r}\lambda_r\right)_{l\in\mathcal{L}} \in \overline{\boldsymbol{\Lambda}}^\star$. □

The next two lemmas are standard technical results from [16] needed to prove Theorem 5.

Lemma 8 (Lemma 1 in [16])**.** *If* $f(x) = \max_{i=1,\ldots,K} f_i(x)$ *and* $f_i(x)$ *for all* i *are locally*

Lipschitz continuous, then we have

$$\frac{D^+}{\mathrm{d}x^+} f(x) \leq \max_{i \in \mathcal{K}} \left\{ \frac{D^+}{\mathrm{d}x^+} f_i(x) \right\},$$

where $\mathcal{K} := \{i \,|\, f_i(x) = f(x)\}$ *and* $\frac{D^+}{\mathrm{d}x^+} f(x)$ *is defined as*

$$\frac{D^+}{\mathrm{d}x^+} f(x) := \limsup_{u \downarrow 0} \frac{f(x+u) - f(x)}{u}.$$

Lemma 9 (Lemma 2 in [16])**.** *Let* $g : [0, \infty) \to [0, \infty)$ *be a locally Lipschitz continuous function.*

(i) Assume that $g(0) = 0$ *and* $\frac{D^+ g(t)}{\mathrm{d}t^+} \leq 0$ *whenever* $g(t) > 0$*. Then* $g(t) = 0$ *for all* $t \geq 0$*.*

(ii) Assume that $g(0) > 0$ *and* $\frac{D^+ g(t)}{\mathrm{d}t^+} \leq -\delta$ *for some* $\delta > 0$ *whenever* $g(t) > 0$*. Then there exists a* $T \geq 0$ *such that* $g(t) = 0$ *for all* $t \geq T$*.*

We now investigate the stability of the system through an analysis of the equations given in Lemma 6.

Consider the Lyapunov function

$$V(\boldsymbol{x}(t)) := \max_{(l,r)} \frac{\alpha^{N_r - n_{l,r}}}{\lambda_r / \gamma_l} x_{l,r}(t), \quad t \geq 0, \tag{3.44}$$

where α is a parameter strictly greater than 1.

The proof of the following result provides a drift condition on the Lyapunov function just defined. The drift condition is one of the primary ingredients needed to show that WQPRA is maximally stable. Specifically, the proof shows that whenever the arrival rate vector is an element of the interior of the capacity region, all fluid sample paths of the system almost surely go to 0 and remain there. By Theorem 4.2 in [21] this is sufficient for maximal stability.

Theorem 5. *The WQPRA algorithm achieves maximal stability.*

Proof of Theorem 5. During this proof the arrival rate vector is assumed to be in $\boldsymbol{\Lambda}^0$ (as defined in Section 3.3). We will show that for some $\delta > 0$, whenever $V(\boldsymbol{x}(t)) > 0$, then

$$\frac{D^+}{\mathrm{d}t^+} V(\boldsymbol{x}(t)) < -\delta. \tag{3.45}$$

This implies the result according Lemma 9 and Theorem 4.2 in [21]. Given any arrival rate vector $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}^0$, by definition there always exists a real number $\varepsilon > 0$ such that

$$(1+\varepsilon) \left(\sum_{r: l \in r} \lambda_r \right)_{l \in \mathcal{L}} \in \overline{\boldsymbol{\Lambda}}^\star.$$

According to Lemma 8 we have

$$\frac{D^+}{\mathrm{d}t^+} V(\boldsymbol{x}(t)) \leq \max_{(\bar{l}, \bar{r}) \in \boldsymbol{\mathcal{S}}} \frac{\alpha^{N_{\bar{r}} - n_{\bar{l},\bar{r}}}}{\lambda_{\bar{r}} / \gamma_{\bar{l}}} \frac{\mathrm{d}}{\mathrm{d}t} x_{\bar{l},\bar{r}}(t), \tag{3.46}$$

where

$$\overline{\boldsymbol{\mathcal{S}}} := \left\{ (\bar{l},\bar{r}) \,:\, V(\boldsymbol{x}(t)) = \frac{\alpha^{N_{\bar{r}}-n_{\bar{l},\bar{r}}}}{\lambda_{\bar{r}}/\gamma_{\bar{l}}} x_{\bar{l},\bar{r}}(t) \right\}.$$

Assume $V(\boldsymbol{x}) > 0$, so that $x_{\bar{l},\bar{r}} > 0$. Let $(l^*, r^*) \in \arg\max_{(l,r)} \gamma_l x_{l,r}/\lambda_r$. Then

$$\frac{\alpha^{N_{\bar{r}}-n_{\bar{l},\bar{r}}}}{\lambda_{\bar{r}}/\gamma_{\bar{l}}} x_{\bar{l},\bar{r}} \geq \frac{\alpha^{N_{r^*}-n_{l^*,r^*}}}{\lambda_{r^*}/\gamma_{l^*}} x_{l^*,r^*} \geq \frac{x_{l^*,r^*}}{\lambda_{r^*}/\gamma_{l^*}}, \tag{3.47}$$

where the first inequality follows by definition and the second uses the facts that $\alpha > 1$ and $1 \leq n_{l^*,r^*} \leq N_{r^*}$.

Under WQPRA we have

$$\begin{aligned}\mu_{\bar{l},\bar{r}} &= \frac{\gamma_{\bar{l}}\, x_{\bar{l},\bar{r}}}{\gamma_{\bar{l}}\, q_{\bar{l}}}\sigma_{\bar{l}} = \frac{\gamma_{\bar{l}}\, x_{\bar{l},\bar{r}}}{\gamma_{l^*}\, q_{l^*}}\sigma_{l^*} = \frac{\gamma_{\bar{l}}\, x_{\bar{l},\bar{r}}}{\gamma_{l^*}\, x_{l^*,r^*}}\mu_{l^*,r^*} = \frac{\gamma_{\bar{l}}\, x_{\bar{l},\bar{r}}/\lambda_{\bar{r}}}{\gamma_{l^*}\, x_{l^*,r^*}/\lambda_{r^*}}\frac{\lambda_{\bar{r}}}{\lambda_{r^*}}\mu_{l^*,r^*} \\ &\geq \frac{1}{\alpha^{N_{\bar{r}}-n_{\bar{l},\bar{r}}}}\frac{\lambda_{\bar{r}}}{\lambda_{r^*}}\mu_{l^*,r^*} \geq \frac{1}{\alpha^{N_{\bar{r}}-n_{\bar{l},\bar{r}}}}\lambda_{\bar{r}}(1+\varepsilon),\end{aligned} \tag{3.48}$$

where the first three equalities follow from (3.42), (3.43), and (3.42) (again). The first inequality follows from (3.47) and the second inequality uses Lemma 7. The proof of the required drift condition (3.45) splits into two cases depending on whether $\bar{l}$ is an ingress station for route $\bar{r}$ or not:

(i) If $\bar{l}$ is the ingress station for route $\bar{r}$ (i.e., $n_{\bar{l},\bar{r}} = 1$), then

$$\begin{aligned}\frac{\alpha^{N_{\bar{r}}-1}}{\lambda_{\bar{r}}/\gamma_{\bar{l}}}\frac{\mathrm{d}}{\mathrm{d}t}x_{\bar{l},\bar{r}} &= \frac{\alpha^{N_{\bar{r}}-n_{\bar{l},\bar{r}}}}{\lambda_{\bar{r}}/\gamma_{\bar{l}}}\left(\lambda_{\bar{r}} - \mu_{\bar{l},\bar{r}}\right) \\ &\leq \alpha^{N_{\bar{r}}-1}\left(\gamma_{\bar{l}} - \frac{\gamma_{\bar{l}}}{\alpha^{N_{\bar{r}}-1}}(1+\varepsilon)\right) \\ &= \alpha^{N_{\bar{r}}-1}\gamma_{\bar{l}} - \gamma_{\bar{l}}(1+\varepsilon) \\ &\leq \gamma_{\bar{l}}\left(\alpha^{N^{\max}-1} - (1+\varepsilon)\right),\end{aligned}$$

where the first inequality uses (3.48) and the second follows by definition.

(ii) If $\bar{l}$ is not the ingress station for route $\bar{r}$ (i.e., $n_{\bar{l},\bar{r}} \geq 2$), then

$$\begin{aligned}\frac{\alpha^{N_{\bar{r}}-n_{\bar{l},\bar{r}}}}{\lambda_{\bar{r}}/\gamma_{\bar{l}}}\frac{\mathrm{d}}{\mathrm{d}t}x_{\bar{l},\bar{r}} &= \frac{\alpha^{N_{\bar{r}}-n_{\bar{l},\bar{r}}}}{\lambda_{\bar{r}}/\gamma_{\bar{l}}}\left(\mu_{\bar{l}_-,\bar{r}} - \mu_{\bar{l},\bar{r}}\right) \\ &= \frac{\alpha^{N_{\bar{r}}-n_{\bar{l},\bar{r}}}}{\lambda_{\bar{r}}}\left(\frac{\gamma_{\bar{l}_-}\, x_{\bar{l}_-,\bar{r}}}{\gamma_{\bar{l}}\, x_{\bar{l},\bar{r}}} - 1\right)\mu_{\bar{l},\bar{r}} \\ &\leq \frac{\alpha^{N_{\bar{r}}-n_{\bar{l},\bar{r}}}}{\lambda_{\bar{r}}}\left(\frac{1}{\alpha} - 1\right)\mu_{\bar{l},\bar{r}} \\ &\leq \left(\frac{1}{\alpha} - 1\right)(1+\varepsilon),\end{aligned}$$

where the second equality follows from (3.42) and (3.43) (as above), the first inequality uses the definition of $(\bar{l},\bar{r})$, and the second inequality uses (3.48) and $\alpha > 1$.

We can then select $\alpha > 1$ small enough such that the result holds. □

3.9 Sketch of diffusion limit proof

This section sketches a proof of Conjecture 5. In this preliminary exposition we have attempted to provide an outline of our approach and sufficient detail to convince the reader that Conjecture 5 is true. Our sketch of the heavy traffic SSC property, which relies on the attraction property of fluid sample paths, follows the general approach developed in [70] and [71]; which was later used in [66]. For purely technical reasons, for the purposes of our analysis we interpolate the values of all our discrete time stochastic processes to all real numbers $t \geq 0$ by linear interpolation between $\lfloor t \rfloor$ and $\lfloor t \rfloor + 1$, where $\lfloor t \rfloor$ denotes the largest integer no greater than t.

Part (a) of Conjecture 5 consists of properties (3.19) and (3.20). To prove properties (3.19) and (3.20) it suffices to show that for any subsequence $\mathcal{K}_1 \subseteq \mathcal{K}$, there exists another subsequence $\mathcal{K}_2 \subseteq \mathcal{K}_1$ such that these properties hold when $k \to \infty$ along $\mathcal{K}_2$. To do that we must construct all processes (for all $k \in \mathcal{K}$) on the same probability space and choose a subsequence $\mathcal{K}_2$ in a way such that the desired properties hold with probability 1 as $k \to \infty$ along $\mathcal{K}_2$.

According to the Skorohod representation theorem (see e.g., [72, p. 70]) for each route r the sequence of input processes and a standard Brownian motion B_r can be constructed on a probability space such that, as $k \to \infty$ along $\mathcal{K}$, the convergence to Brownian motions, given by (3.25), holds uoc with probability 1:

$$\left(\widetilde{a}_r^{(k)}(t) - \lambda_r^{(k)} kt,\, t \geq 0\right) \stackrel{\text{uoc}}{\to} (s_r\, B_r(t),\, t \geq 0)\,, \tag{3.49}$$

where $\stackrel{\text{uoc}}{\to}$ denotes uniform on compact sets convergence of elements of $D([0,\infty),\infty)$ the standard Skorohod space of cadlag functions defined on $[0,\infty)$ taking real numbers. The sequences of processes $\{H_{i,j}^{(k)}, G^{(k)}\}$, with distributions governed by the Markov chains $M^{(k)}$, and standard Brown motions $B_{i,j}$ can be constructed on probability spaces such that the convergences (3.26) and (3.33) also hold uniformly on compact intervals (uoc) with probability 1:

$$\left(\widetilde{h}_{1,j}^{(k)}(t) - \mu_{1,j}^{\star} kt,\, t \geq 0\right) \stackrel{\text{uoc}}{\to} (\varrho_{1,j}\, B_{1,j}(t),\, t \geq 0)\,, \quad j = 1, \ldots, N^{\max}\,, \tag{3.50}$$

$$\left(\widetilde{h}_{i-1,j}^{(k)}(t) - \widetilde{h}_{i,j}^{(k)}(t),\, t \geq 0\right) \stackrel{\text{uoc}}{\to} (\widehat{\varrho}_{i,j}\, B_{i,j}(t),\, t \geq 0)\,, \quad j = 2, \ldots, N^{\max},\ \ i = 2, \ldots, j\,. \tag{3.51}$$

We assume that the underlying probability space $(\Omega, \mathcal{F}, \mathbb{P})$ is the direct product of the probability spaces specified above, and (without loss of generality) assume that this probability space is complete. We will denote elements of Ω by ω. For the remainder of this section statements are made for $j = 1, \ldots, N^{\max}$ and $i = 1, \ldots, j$.

Recall that

$$\widetilde{z}_{i,j}^{(k)}(t) = \widetilde{w}_{i,j}^{(k)}(t) + \widetilde{y}_{i,j}^{(k)}(t)\,, \quad t \geq 0\,. \tag{3.52}$$

We now have the probability 1 convergence version of $\widetilde{w}_{i,j}$ (as defined in (3.12) and (3.15) in Section 3.6):

$$\left(\widetilde{w}_{i,j}^{(k)}(t),\ t \geq 0\right) \stackrel{\text{uoc}}{\to} \left(\widetilde{w}_{i,j}(t),\ t \geq 0\right). \tag{3.53}$$

Note that by definition of a Brownian motion, the sample paths of $\widetilde{w}_{i,j}$ are continuous.

For the rest of this section we define $\Omega_2 \subset \Omega$ as the (measurable, probability 1) collection $\omega \in \Omega$ such that (3.49)–(3.53) hold along the subsequence $\mathcal{K}_2$.

For any $k \in \mathcal{K}$ a sample path of $\widetilde{y}_{i,j}^{(k)}$, as defined in (3.29), is a non-negative and non-decreasing cadlag function (since departing workload cannot exceed available workload service and both of these functions are assumed 0 at $t = 0$). Therefore, for any fixed $\omega \in \Omega$, from any subsequence $\mathcal{K}_3(\omega) \subseteq \mathcal{K}_2$, possibly depending on ω, we can choose a further subsequence $\mathcal{K}_4(\omega) \subset \mathcal{K}_3(\omega)$ along which

$$\widetilde{y}_{i,j}^{(k)} \Rightarrow \widetilde{y}_{i,j}^{\circ}\,, \tag{3.54}$$

where $\widetilde{y}_{i,j}^{\circ}$ is some non-negative and non-decreasing function in $D([0,\infty), \overline{\mathbb{R}})$. The notation $\Rightarrow$ means convergence at every point of continuity of the limit function. For this function let $\widetilde{z}_{i,j}^{\circ} := \widetilde{w}_{i,j} + \widetilde{y}_{i,j}^{\circ}$.

Therefore, to prove the convergences (3.19) and (3.20) (Conjecture 5(a)) it will suffice to prove that as $k \to \infty$ along $\mathcal{K}_2$, for any $\omega \in \Omega_2$ (and, therefore with probability 1), we have

$$\left(\widetilde{y}_{i,j}^{(k)}(t),\ t \geq 0\right) \overset{\text{uoc}}{\to} \left(\widetilde{y}_{i,j}^{\circ}(t),\ t \geq 0\right), \tag{3.55}$$

$$\left(\widetilde{\boldsymbol{x}}_{i,j}^{(k)}(t),\ t \geq 0\right) \overset{\text{uoc}}{\to} \left(\widetilde{\boldsymbol{x}}_{i,j}^{\circ}(t),\ t \geq 0\right), \tag{3.56}$$

where the equivalance

$$\widetilde{y}_{i,j}^{\circ} \equiv \widetilde{y}_{i,j} \tag{3.57}$$

holds, as defined in (3.14) and (3.16), i.e.,

$$\widetilde{y}_{i,j}(t) := -\left[0 \wedge \inf_{0 \leq u \leq t} \widetilde{w}_{i,j}(u)\right], \quad t \geq 0\,,$$

and the equivalence

$$\widetilde{\boldsymbol{x}}_{i,j}^{\circ} \equiv \widetilde{\boldsymbol{x}}_{i,j} = \widetilde{z}_{i,j}\,\boldsymbol{\zeta}_{i,j} \tag{3.58}$$

holds, as similarly defined in Section 3.7, and where these are then utilised in $\widetilde{z} = \widetilde{w} + \widetilde{y}$ as defined in (3.18).

3.9.1 Attraction property of fluid sample paths

To prove the equivalences (3.57) and (3.58), as just outlined, we will first show that the queue size process is proportional to the vector $\boldsymbol{\zeta}$, ensuring that workload service is not wasted. Our expressions for the diffusion limits of the queue size and workload process then follow from verification of the conditions of a Skorohod problem (see Appendix A.1). The next two propositions provide the key information needed to show these conditions.

In addition to the Lyapunov function V (3.59) we now need the Lyapunov function

$$H(\boldsymbol{x}(t)) := \frac{V(\boldsymbol{x}(t))}{V(\boldsymbol{x}^{\star}(t))} - 1\,, \quad t \geq 0\,, \tag{3.59}$$

where

$$\boldsymbol{x}^{\star}(t) := \arg\min_{\boldsymbol{y} \in \mathcal{V}} ||\boldsymbol{y} - \boldsymbol{x}(t)||$$

is the element of the invariant manifold $\mathcal{V}$ closest to $\boldsymbol{x}$. We use the conventions: (i) if $\boldsymbol{x}(t) \neq 0$ and $\boldsymbol{x}^\star = 0$, then we put $H(\boldsymbol{x}(t)) = \infty$, and (ii) if $\boldsymbol{x}(t) = 0$, then we put $H(\boldsymbol{x}(t)) = 0$.

The next proposition (which we do not prove) shows the attraction property of fluid sample paths to the invariant manifold. The drift conditions in the first part of the proposition are used to ensure that the fluid sample paths eventually go to the invariant manifold, and that once they are there they do not leave it (this is summarised more formally in the final part of the proposition). The second part of the proposition provides a guarantee that so long as the workload of the system has a positive initial condition, then in heavy traffic the fluid sample paths remain bounded away from zero. This guarantees that when the queue length process enters the invariant manifold it does so at a point that is not the origin — which is important for our diffusion result since it also ensures workload service is not wasted.

Proposition 10. *Assume* $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}^\star$.

(i) *For some* $\delta > 0$, *if* $\boldsymbol{x}(t) \notin \mathcal{V}$, *then*

$$\frac{D^+}{\mathrm{d}t^+} V(\boldsymbol{x}(t)) < -\delta\,.$$

If $\boldsymbol{x}(t) \in \mathcal{V}$, *then*

$$\frac{D^+}{\mathrm{d}t^+} V(\boldsymbol{x}(t)) = 0\,.$$

(ii) *If* $x_{l,r}(0) > 0$, *then* $x_{l,r}(t) > 0$ *for all* $t \geq 0$.

(iii) *If* $\boldsymbol{x}(0) \in \mathcal{V}$, *then* $H(\boldsymbol{x}(t)) \equiv 0$. *If* $\boldsymbol{x}(0) \notin \mathcal{V}$, *then there exists* $T > 0$ *such that* $H(\boldsymbol{x}(t)) \in \mathcal{V}$ *for all* $t \geq T$.

We currently know very little about the limiting function $\widetilde{y}^\circ_{i,j}$ from (3.54). The following proposition gives us the tools we will need to provide a proof of Conjecture 5. Later, in order to meet the conditions of Proposition 41 (Skorohod problem) and prove Conjecture 5 we need to ensure $y^\circ_{i,j}$ has some properties to show it converges to $\widetilde{y}_{i,j}$ as defined in (3.14) and (3.16). Namely, we would like to show that $\widetilde{y}^\circ_{i,j}$ does not increase when $\widetilde{z}^\circ_{i,j} = \widetilde{w}_{i,j} + \widetilde{y}^\circ_{i,j}$ is positive. To do that we need to show $\widetilde{y}^\circ_{i,j}$ is continuous for all $t \geq 0$, and to do that we need to show it is finite for all $t \geq 0$. The purpose of the first part of the next proposition is to give precisely the property of $\widetilde{y}^\circ_{i,j}$ we will need later: that it does not increase for a small amount of time when $\widetilde{z}^{(k)}_{i,j}(t) > 0$. Later, to use the first part on all points of continuity, we first use it on points of discontinuity combined with the second part of the proposition to show that there are in fact no points of discontinuity. We will then show workload service is not wasted when the queue length process is on the invariant manifold, a property given to us by (c), (d), and (e), which enables us to use (a).

The *oscillation* of function h over a subset $A \subseteq \mathbb{R}_0$ is defined as

$$\mathrm{Osc}(h;\, A) := \sup_{\zeta_1, \zeta_2 \in A} |h(\zeta_1) - h(\zeta_2)|\,.$$

Let $\lambda^+ := \max_{(l,r)} \lambda_r / a^{N_r - n_l}$.

Proposition 11. **(i)** *Suppose the scheduling discipline is WQPRA. Suppose* $\omega \in \Omega_2$ *and a subsequence* $\mathcal{K}_4(\omega) \subset \mathcal{K}_2$ *are fixed such that along this subsequence* $\widetilde{y}_{i,j}^{(k)} \Rightarrow \widetilde{y}_{i,j}^{\circ}$ (3.54) *holds. Suppose a sequence* $\{\widetilde{t}^{(k)},\, k \in \mathcal{K}_4(\omega)\}$ *is fixed such that for* $1 \le j \le N^{\max}$ *and* $1 \le i \le j$, *for* $0 < C_{i,j} < \infty$,

$$\widetilde{t}^{(k)} \to t' \ge 0\,, \tag{3.60}$$

$$\widetilde{z}_{i,j}^{(k)}(\widetilde{t}^{(k)}) \to C_{i,j} > 0\,. \tag{3.61}$$

Let δ *be such that*

$$\varepsilon = Osc(\widetilde{w}_{i,j};\, [t' - 3\delta, t' + 3\delta] \cap \mathbb{R}_0) < C_{i,j}/2\,.$$

Then,

(a) $\widetilde{y}_{i,j}^{\circ}$ *does not increase in* $(t', t' + \delta]$, *that is,* $\widetilde{y}_{i,j}^{\circ}(t' + \delta) - \widetilde{y}_{i,j}^{\circ}(t') = 0$,

(b) $C_{i,j} - 2\varepsilon < \widetilde{z}_{i,j}^{\circ}(t) \le C_{i,j}^{1} + 2\varepsilon$, *where* $C_{i,j}^{1} := C_{i,j}\lambda^{+} S\, V(\overline{\boldsymbol{x}}^{k,0}(0)) \max\{\lambda_r\}$ *for all* $t \in [t', t' + \delta]$, *and*

(c) for any $0 < \delta' < \delta$, $\left(\widetilde{\boldsymbol{x}}_{i,j}^{(k)}(t) - \widetilde{z}_{i,j}^{(k)}(t)\boldsymbol{\zeta}_{i,j},\ t \in [t' + \delta', t' + \delta]\right) \overset{\text{uoc}}{\to} 0$.

(ii) *Suppose the conditions of (i) hold and, in addition,* $\widetilde{t}^{(k)} = t'$ *for all* k *and* $\widetilde{\boldsymbol{x}}_{i,j}^{(k)}(t') \to C_{i,j}\boldsymbol{\zeta}_{i,j}$. *Then,*

(d) $\widetilde{z}_{i,j}^{\circ}(t') = C_{i,j}$, *and*

(e) $\left(\widetilde{\boldsymbol{x}}_{i,j}^{(k)}(t) - \widetilde{z}_{i,j}^{(k)}(t)\boldsymbol{\zeta}_{i,j},\ t \in [t', t' + \delta]\right) \overset{\text{uoc}}{\to} 0$.

In order to prove Proposition 11 we first need to build some more machinery. This is provided by Lemma 12 below. Firstly, for any scalar function $h = (h(t), t \in \mathbb{R}_0)$, we define the shift operator θ_τ, $d \in \mathbb{R}_0$, in the standard way:

$$(\theta_\tau h)(t) = h(\tau + t), \quad t \in \mathbb{R}_0\,,$$

which is applied component-wise for vector-valued functions.

Now, recall $\boldsymbol{\mathcal{Z}}^{(k)}$ as defined at the beginning of Section 3.7 which contains detail on most aspects of how the system we are studying evolves. Let $\Theta(\tau)\boldsymbol{\mathcal{Z}}^{(k)}$ be this process restarted at an arbitrary time $\tau \ge 0$. For the reader's convenience, in Table 3.2 we illustrate how this shift transformation affects $\boldsymbol{\mathcal{Z}}^{(k)}$ for some of its components. The key difference in effect for different components is that for process components which are cumulative the value of the process at the point of restart needs to be subtracted, and otherwise it does not. Furthermore, the fluid scaled version $\Theta(\tau)\boldsymbol{z}^{(k)}$ is defined analogously to $\Theta(\tau)\boldsymbol{\mathcal{Z}}^{(k)}$.

Using these definitions, to establish the result for the diffusion scaled paths in the interval $[\widetilde{t}^{(k)},\, \widetilde{t}^{(k)} + \delta]$, $\delta > 0$, we study the fluid-scaled paths $\boldsymbol{z}^{(k)}$ in the corresponding interval $[k\widetilde{t}^{(k)},\, k\widetilde{t}^{(k)} + k\delta]$. That is, we consider the following family of fluid-scaled paths restarted at times which are a constant $T > 0$ apart from each other. For each k and each integer $\kappa \in [0,\, 2\delta k/T - 1]$ consider the path

$$\overline{\boldsymbol{z}}^{k,\kappa} := \Theta(k\widetilde{t}^{k} + T\kappa)\boldsymbol{z}^{(k)}\,, \tag{3.62}$$

Table 3.2: Effect of shift transformation on $\boldsymbol{\mathcal{Z}}^{(k)}$.

Original process:	$\boldsymbol{X}^{(k)}$	$Z^{(k)}$	$Y^{(k)}$	$\overline{A}_r^{(k)}$	$\overline{\boldsymbol{D}}_i^{(k)}$
Transformed process:	$\theta_\tau \boldsymbol{X}^{(k)}$	$\theta_\tau Z^{(k)}$	$\theta_\tau Y^{(k)} - Y^{(k)}(\tau)$	$\theta_\tau \overline{A}_r^{(k)} - \overline{A}_r^{(k)}(\tau)$	$\theta_\tau \overline{\boldsymbol{D}}_i^{(k)} - \overline{\boldsymbol{D}}_i^{(k)}(\tau)$

with similar notation for corresponding components of $\overline{\boldsymbol{z}}^{k,\kappa}$, for example $\overline{\boldsymbol{x}}^{k,\kappa}$, $\overline{z}^{k,\kappa}$, $\overline{y}^{k,\kappa}$, $\overline{\overline{a}}_r^{k,\kappa}$, and $\overline{\overline{\boldsymbol{d}}}_i^{k,\kappa}$.

Lemma 12. **(i)** *Suppose the conditions of Proposition 11(i) hold.*

Let $C^1_{i,j} = \lambda^+ SV(\overline{\boldsymbol{x}}^{k,0}(0)) \max\{\lambda_r\}$. *Then, for all sufficiently small* $\varepsilon_2 > 0$, *there exists* $T > 0$ *such that, for all sufficiently large* k, *properties* (3.63) *and* (3.64) *hold for all integer* κ *in the interval* $[1, 2\delta k/T - 1]$ *and the property* (3.65) *holds for all integer* κ *in the interval* $[0, 2\delta k/T - 1]$ *(including 0). For all* $u \in [0, T]$*:*

$$H(\overline{\boldsymbol{x}}^{k,\kappa}(u)) < \varepsilon_2\,, \tag{3.63}$$

$$\overline{y}^{k,\kappa}_{i,j}(u) \equiv \overline{y}^{k,\kappa}_{i,j}(u) - \overline{y}^{k,\kappa}_{i,j}(0) = 0\,, \tag{3.64}$$

$$C_{i,j} - 2\varepsilon < \overline{z}^{k,\kappa}_{i,j}(u) < C^1_{i,j}\,. \tag{3.65}$$

(ii) *Suppose the conditions of Proposition 11(ii) hold. Then, there exists* $T > 0$ *such that, for all sufficiently large* k *properties* (3.63)–(3.64) *hold for all integer* κ *in the interval* $[0, 2\delta k/T - 1]$ *(including 0).*

Proof. (i) We choose $T > 0$ according to Proposition 10(iii), the time that the fluid sample paths must belong to the invariant manifold. This also gives us (3.63) for $\kappa \geq 1$.

By Proposition 10(i) we have that $V(\boldsymbol{x})$ is continuous non-increasing, so for $\kappa = 0$ we have

$$\limsup_{k\to\infty} \sup_{u\in[0,T]} ||\widetilde{z}_{i,j}(u)|| < C_{i,j}\lambda^+ SV(\overline{\boldsymbol{x}}^{k,0}(0)) \max\{\lambda_r\}\,,$$

which gives the upper bound in (3.65) for $\kappa = 0$. The lower bound of 0 in (3.65) is obtained similarly for $\kappa = 0$ by Proposition 10(ii).

We now consider (3.63) to (3.65) for $\kappa \geq 1$. Suppose these properties do not hold. This means that there exists a subsequence $\mathcal{K}_5 \subset \mathcal{K}_4(\omega)$ such that at least one of the properties (3.63) to (3.65) does not hold for some $\kappa' \geq 1$ but do hold for $0 < \kappa \leq \kappa' - 1$ when they should. This is possible since we already showed (3.65) holds for $\kappa = 0$ and properties (3.63) and (3.64) need not hold for $\kappa = 0$, so $\kappa' = 1$ is possible.

Proposition 10(iii) easily gives that (3.63) must hold for κ' and the lower bound in (3.65) follows from Proposition 10(ii). We then also obtain (3.64) for κ' by observing that $\boldsymbol{x}_{i,j}$ is therefore proportional to $\boldsymbol{\zeta}_{i,j}$ and bounded away from 0. Given that (3.63) and

(3.64) hold, we know

$$\begin{aligned}\overline{z}_{i,j}^{k,\kappa'}(u) &= \overline{z}_{i,j}^{k,0}(T) + \sum_{\kappa=1}^{\kappa'-1}\Big[\overline{z}_{i,j}^{k,\kappa}(T) - \overline{z}_{i,j}^{k,\kappa}(0)\Big] - \overline{z}_{i,j}^{k,\kappa'}(u)\overline{z}_{i,j}^{k,\kappa'}(0) \\ &= \overline{z}_{i,j}^{k,0}(T) + \sum_{\kappa=1}^{\kappa'-1}\Big[\overline{w}_{i,j}^{k,\kappa}(T) - \overline{w}_{i,j}^{k,\kappa}(0)\Big] + \overline{w}_{i,j}^{k,\kappa'}(u) - \overline{w}_{i,j}^{k,\kappa'}(0) \\ &= \overline{z}_{i,j}^{k,0}(T) + \widetilde{w}_{i,j}^{k}\big(\widetilde{t}^{(k)} + \kappa' T/k + u/k\big) + \widetilde{w}_{i,j}^{k}\big(\widetilde{t}^{(k)} + T/k\big)\,,\end{aligned} \tag{3.66}$$

where

$$|\widetilde{w}_{i,j}^{k}\big(\widetilde{t}^{(k)} + \kappa' T/k + u/k\big) + \widetilde{w}_{i,j}^{k}\big(\widetilde{t}^{(k)} + T/k\big)| < 2\varepsilon$$

for all large k by our choice of ε and the uoc convergence $\widetilde{w}_{i,j}^{(k)} \to \widetilde{w}_{i,j}$ — which gives the upper bound in (3.65).

(ii) This is a modified version of the proof of (i), where the fact that $\boldsymbol{x}_{i,j}$ is an element of the invariant manifold at time t' allows us to use the first part of Proposition 10(iii) immediately on H, and consequently the fact that no workload service is wasted for $\kappa = 0$ holds immediately as well. □

Proof of Proposition 11. Choose a small $\varepsilon_2 > 0$ and take T as in Lemma 12(i). If we recall that time $u \in [0,T]$ for $\overline{\boldsymbol{z}}^{k,\kappa}$ corresponds to the time $\widetilde{t}^{(k)} + \kappa T/k + u/k$ on the diffusion time scale, we see that Lemma 12 implies that for all large k we have (3.67) for $t \in [\widetilde{t}^{(k)}, \widetilde{t}^{(k)} + (3/2)\delta]$ and (3.68) and (3.69) for $t \in [\widetilde{t}^{(k)} + T/k, \widetilde{t}^{(k)} + (3/2)\delta]$:

$$C_{i,j} - 2\varepsilon < \widetilde{z}_{i,j}^{(k)}(t) < C_{i,j}^{1}\,, \tag{3.67}$$

$$\widetilde{y}_{i,j}^{(k)}(t) - \widetilde{y}_{i,j}^{(k)}(\widetilde{t}^{(k)} + T/k) = 0\,, \tag{3.68}$$

$$H(\widetilde{x}_{i,j}^{(k)}(t)) < \varepsilon_2\,. \tag{3.69}$$

Since $\widetilde{y}_{i,j}^{(k)} \Rightarrow \widetilde{y}_{i,j}^{\circ}$, $\widetilde{z}_{i,j}^{(k)} \Rightarrow \widetilde{z}_{i,j}^{\circ}$, and both $\widetilde{y}_{i,j}$ and $\widetilde{z}_{i,j}$ are cadlag, properties (3.67) and (3.68) imply statements (a) and (b), and (c) follows from (3.69). To obtain (d) and (e), apply Lemma 12(ii) similarly. □

3.9.2 Diffusion limit of workload process and state space collapse

In this subsection we sketch the proof of the uoc convergence of $\widetilde{y}_{i,j}^{(k)}$ and $\widetilde{x}_{i,j}^{(k)}$ in $(\Omega_2, \mathcal{F}, \mathbb{P})$, as stated in (3.55) and (3.56), which directly completes the sketch of proof of part (a) of Conjecture 5. We consider arbitrary fixed $\omega \in \Omega_2$. As explained earlier, for an arbitrary subsequence $\mathcal{K}_3(\omega) \subset \mathcal{K}_2$, there exists another subsequence $\mathcal{K}_4(\omega) \subset \mathcal{K}_3(\omega)$ such that the convergence (3.54), namely $\widetilde{y}_{i,j}^{(k)} \Rightarrow \widetilde{y}_{i,j}^{\circ}$, holds along $\mathcal{K}_4(\omega)$.

Based on Proposition 41 (Skorohod problem), given in Appendix A.1, the proof of (3.55) and (3.56) will follow directly from the following proposition (which we do not prove).

Proposition 13. *For any $\omega \in \Omega_2$ (defined immediately after (3.53)), with $k \to \infty$ along $\mathcal{K}_4(\omega)$:*

(i) The functions $\widetilde{y}_{i,j}^{\circ}$ are continuous everywhere they are finite and $\widetilde{y}_{i,j}(0) = 0$;

(ii) *The limit functions* $\widetilde{y}^{\circ}_{i,j}$ *are finite everywhere in* $[0,\infty)$;

(iii) *If* $\widetilde{z}^{\circ}_{i,j}(t) > 0$, *then* t *is not a point of increase of* $\widetilde{y}^{\circ}_{i,j}$.

(iv) $\big(\widetilde{y}^{(k)}_{i,j}(t),\, t \geq 0\big) \overset{\text{uoc}}{\to} \big(\widetilde{y}_{i,j}(t),\, t \geq 0\big)$.

(v) $\big(\widetilde{\boldsymbol{x}}^{(k)}_{i,j}(t),\, t \geq 0\big) \overset{\text{uoc}}{\to} \big(\widetilde{z}_{i,j}(t)\boldsymbol{\zeta}_{i,j},\, t \geq 0\big)$

3.9.3 Workload minimisation property

In this subsection we sketch the proof of part (b) of Conjecture 5, which states that for any scheduling discipline Φ with corresponding diffusion scaled workload process $\widetilde{z}^{(k)}_{\Phi}$, in heavy traffic, along $\mathcal{K}$ we have

$$\liminf_{k\to\infty} \mathbb{P}(\widetilde{z}^{k}_{\Phi}(t) > u) \geq \mathbb{P}(\widetilde{z} > u)\,.$$

This follows in a similar manner to the proof presented in the previous subsection, now utilising the second part of Proposition 41 (Skorohod problem). Consider the same probability space $(\Omega, \mathcal{F}, \mathbb{P})$ as constructed earlier for the proof of part (a) of Conjecture 5. For arbitrary fixed $\omega \in \Omega$, look at paths of $\widetilde{z}^{(k)}_{\Phi}$, $\widetilde{y}^{(k)}_{\Phi}$, and $\widetilde{w}\Phi^{(k)}$, which correspond to discipline Φ. Since we are considering the same probability space, $\widetilde{w}^{(k)}_{\Phi} = \widetilde{w}^{(k)}$ and therefore $\widetilde{w}^{(k)}_{\Phi} \overset{\text{uoc}}{\to} \widetilde{w}_{\Phi}$, as in (3.53) where $\widetilde{w}^{(k)} \overset{\text{uoc}}{\to} \widetilde{w}$.

As in the previous subsections, for any subsequence $\mathcal{K}_3(\omega) \subseteq \mathcal{K}_2$ we can choose a further subsequence $\mathcal{K}_4(\omega) \subseteq \mathcal{K}_3$ such that $\widetilde{y}^{(k)}_{\Phi} \to \widetilde{y}_{\Phi}$, where $\widetilde{y}_{\Phi}$ is some non-decreasing and non-negative function in $D([0,\infty), \overline{\mathbb{R}})$. Therefore, for any $t > 0$ where $\widetilde{y}_{\Phi}$ is continuous, as $k \to \infty$ along $\mathcal{K}_4(\omega)$,

$$\lim \widetilde{z}^{(k)}_{\Phi}(t) = \widetilde{w}(t) + \widetilde{y}_{\Phi}(t), \quad t \geq 0\,,$$

and $\widetilde{w}(t) + \widetilde{y}_{\Phi}(t) \geq 0$ since $\widetilde{z}_{\Phi}(t) \geq 0$ by definition. Then, by right-continuity this also holds at $t = 0$. Therefore, by Proposition 41, $\widetilde{y}_{\Phi}(t) \geq \widetilde{y}(t)$ for all $t \geq 0$. This and the continuity of $\widetilde{y}$ implies that for any $t \geq 0$, $\liminf_{k\to\infty} \widetilde{x}^{(k)}_{\Phi}(t) \geq \widetilde{z}(t)$ holds along $\mathcal{K}_4$, and therefore along $\mathcal{K}$ since the subsequence $\mathcal{K}_3(\omega)$ is arbitrary. This will complete the proof of Conjecture 5(b).

3.10 Other supporting lemmas

We first prove Lemma 6. To do this we need to define some additional stochastic processes. Let $\overline{A}_{l,r}(t) := \sum_{\tau=0}^{t} A_{l,r}(\tau)$ be the total arrivals to queue (l,r) and $\overline{R}_l(t) := \sum_{\tau=0}^{t-1} R_l(\tau)$ be the total amount of service available to station l from time slot 0 to $t-1$. Let $\overline{D}_{l,r}(t) := \sum_{\tau=0}^{t-1} D_{l,r}(\tau)$, $D_l(t) := \sum_{l:l\in r} D_{l,r}(t)$, and $\overline{D}_l(t) := \sum_{\tau=0}^{t-1} D_l(\tau)$. Further, let $\overline{A}_{l,r}(0) = \overline{R}_l(0) = \overline{D}_l(0) = \overline{D}_{l,r}(0) = 0$. As noted before, we interpolate these newly defined processes linearly between $\lfloor t \rfloor$ and $\lfloor t \rfloor + 1$, where $\lfloor t \rfloor$ denotes the largest integer no greater than t.

Then the evolution of the queue length can be rewritten as

$$X_{l,r}(t) = X_{l,r}(0) + \overline{A}_{l,r}(t) + \overline{D}_{l^{(r)}_{-},r}(t) - \overline{D}_{l,r}(t) \tag{3.70}$$

for all (l,r). Note that $\overline{D}_{l^-(r),r}(t) = 0$ if $n_l = 1$.

Recall that we usually denote fluid scaled processes and their limits with lower case letters (or Greek characters), for example, for $\overline{A}_{l,r}^{(k)}$, $\overline{R}_l^{(k)}$, and $\overline{D}_{l,r}^{(k)}$:

$$\overline{a}_{l,r}^{(k)}(t) := k^{-1}\overline{A}_{l,r}^{(k)}(k^2 t)\,, \quad \overline{r}_l^{(k)}(t) := k^{-1}\overline{R}_l^{(k)}(k^2 t), \quad \overline{d}_l^{(k)}(t) := k^{-1}\overline{D}_l^{(k)}(k^2 t)\,, \quad t \geq 0\,. \tag{3.71}$$

Lemma 14. *Under the WQPRA algorithm, with probability 1, for any positive sequence $\mathcal{K}_{f_1}$ such that $k \to \infty$ along $\mathcal{K}_{f_1}$ there exists a subsequence $\mathcal{K}_f \subset \mathcal{K}_{f_1}$ such that as $k \to \infty$ along $\mathcal{K}_f$ the following convergence holds uniformly over compact intervals of time t:*

$$\overline{a}_{l,r}^{(k)}(t) \to \lambda_{l,r}\, t \qquad \forall\,(l,r) \in \mathcal{S}\,, \quad t \geq 0\,, \tag{3.72}$$

$$g_m^{(k)}(t) \to \pi_m\, t \qquad \forall\, l \in \mathcal{L}\,, \quad t \geq 0\,, \tag{3.73}$$

$$\overline{r}_l^{(k)}(t) \to \overline{\sigma}_l(t) \qquad \forall\, l \in \mathcal{L}\,, \quad t \geq 0\,, \tag{3.74}$$

$$\overline{d}_{l,r}^{(k)}(t) \to \overline{\mu}_{l,r}(t) \qquad \forall\,(l,r) \in \mathcal{S}\,, \quad t \geq 0\,, \tag{3.75}$$

$$\overline{d}_l^{(k)}(t) \to \overline{d}_l(t) \qquad \forall\, l \in \mathcal{L}\,, \quad t \geq 0\,, \tag{3.76}$$

$$x_{l,r}^{(k)}(t) \to x_{l,r}(t) \qquad \forall\,(l,r) \in \mathcal{S}\,, \quad t \geq 0\,, \tag{3.77}$$

$$q_l^{(k)}(t) \to \overline{q}_l(t) \qquad \forall\,(l,r) \in \mathcal{S}\,, \quad t \geq 0\,, \tag{3.78}$$

where the limiting functions are Lipschitz continuous in $[0,\infty)$, which implies that these limiting functions are differentiable for almost all t. Let $\mathcal{T}$ be the set of time instants where these functions are differentiable. Additionally, the following equations hold for all $t \in \mathcal{T}$:

$$\frac{\mathrm{d}}{\mathrm{d}t}\overline{\sigma}_l(t) = \sigma_l(t) \qquad \forall\, l \in \mathcal{L}\,, \tag{3.79}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\overline{\mu}_{l,r}(t) = \mu_{l,r}(t) \qquad \forall\,(l,r) \in \mathcal{S}\,, \tag{3.80}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\overline{d}_l(t) = \sigma_l(t) \qquad \forall\,(l,r) \in \mathcal{S}\,, \quad \textit{whenever } q_l(t) > 0\,, \tag{3.81}$$

$$\overline{d}_l(t) = \sum_{r:l\in r} \overline{\mu}_{l,r}(t) \qquad \forall\, l \in \mathcal{L}\,, \tag{3.82}$$

$$q_l(t) = \sum_{r:l\in r} x_{l,r}(t) \qquad (l,r) \in \mathcal{S}\,, \tag{3.83}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}x_{l,r}(t) = \lambda_{l,r} + \mu_{l^-(r),r}(t) - \mu_{l,r}(t) \qquad \forall\,(l,r) \in \mathcal{S}\,, \tag{3.84}$$

where $\mu_{l^-(r),r} = 0$ if $n_{l(r)} = 1$ and $\lambda_{l,r} = 0$ if $n_{l(r)} > 1$. Here: (i) $\boldsymbol{\mu}(t)$ satisfies (3.42)*, and (ii) $\sigma(t)$ lies on the boundary of the capacity region $\overline{\boldsymbol{\Lambda}}^\star$ and satisfies* (3.43)*.*

Proof of Lemma 6. Equations (3.36) and (3.41)–(3.43) follow directly from Lemma 14. Then, the remaining equations in Lemma 6 follow from elementary combinations of these. □

Hence, to prove Lemma 6 we now prove Lemma 14.

Proof of Lemma 14. Equations (3.72) and (3.73) follow directly from the functional strong law of large numbers.

Note that for any $0 \le t_1 \le t_2$ we have

$$0 \le \frac{1}{k}\overline{R}_l(kt_2) - \frac{1}{k}\overline{R}_l(kt_1) \le t_2 - t_1\,, \tag{3.85}$$

where we use the fact that each station l can transfer at most a single packet in any time slot. Thus the sequence of functions $\{\frac{1}{k}\overline{R}_l(kt)\}$ is uniformly equicontinuous, and since $\overline{R}_l(0) = 0$, the sequence is uniformly bounded. Similarly the sequence $\{\frac{1}{k}\overline{D}_{l,r}(kt)\}$ is uniformly bounded and uniformly equicontinuous. Consequently, according to the Arzela–Ascoli theorem (see e.g., [72, Theorem 7.2]), there must exist a subsequence $\mathcal{K}_f$ for which (3.74) and (3.75) hold. Since (3.76)–(3.78) are elementary functions of (3.72)–(3.75) we also trivially obtain these.

The Lipschitz continuity of the limits in (3.72)–(3.78), follows from the Lipschitz continuity of their pre-limit forms. Hence, these limiting functions are differentiable for almost all t. In the rest of the proof we consider all $t \in \mathcal{T}$ (recall this is the set of time instances where these functions are differentiable).

Next we prove (3.79). Note that $\sigma_l(q(t))$ is continuous with respect to q when $q_l > 0$. Therefore, for any $\varepsilon > 0$ there exists a $u > 0$ such that for all $s \in [t, t+u]$ we have

$$|\sigma_l(q(s)) - \sigma_l(q(t))| \le \varepsilon\,. \tag{3.86}$$

Since $\frac{1}{k}Q(\lfloor k\,s\rfloor) \to q(s)$ uniformly over compact intervals of time, and $\sigma_l(aQ) = \sigma(Q)$ for any $a > 0$, we have $\sigma_l(Q(\lfloor ks\rfloor)) \to \sigma_l(q(s))$ with probability 1. Thus, there exists an integer $K > 0$ such that for all $k > K$ and $s \in [t, t+u]$,

$$\sigma_l(q(s)) - \varepsilon \le \sigma_l(Q(\lfloor ks\rfloor)) \le \sigma_l(q(s)) + \varepsilon\,. \tag{3.87}$$

Combining with (3.86), we have

$$\sigma_l(q(t)) - 2\varepsilon \le \sigma_l(Q(\lfloor ks\rfloor) \le \sigma_l(q(t)) + 2\varepsilon\,. \tag{3.88}$$

By the definition of the limit in (3.79), for any $s \in [t, t+u]$ we have

$$\overline{\sigma}_l(s) - \overline{\sigma}_l(t) = \lim_{k\to\infty} \frac{1}{k} \sum_{j=\lfloor kt\rfloor}^{\lfloor ks\rfloor} \overline{R}_l(j)\,.$$

Define the filtration $\mathcal{F}_j$, $j = 1, 2, \ldots$, where $\mathcal{F}_j$ is the σ-algebra generated by the random variables $\overline{A}_{l,r}(\lfloor kt\rfloor + j')$, $\overline{R}_l(\lfloor kt\rfloor + j')$, $\overline{D}_{l,r}(\lfloor kt\rfloor + j')$, and $X_{l,r}(\lfloor kt\rfloor + j')$ for all $(l,r) \in \mathcal{S}$ and for $j' = 0, 1, \ldots, j-1$. Let

$$E_j := \overline{R}_l(\lfloor kt\rfloor + j) - \mathbb{E}\big[\overline{R}_l(\lfloor kt\rfloor + j)\,|\,Q(\lfloor kt\rfloor + j)\big]\,.$$

Therefore $\sum_{i=0}^{j-1} E_i$, $j = 1, 2, \ldots$ is a martingale with respect to $\mathcal{F}_j$, $j = 1, 2, \ldots$. Further $\mathbb{E}[E_j^2]$ is bounded for all j. Hence, using a strong law of large numbers for martingales [73] we have

$$\lim_{j\to\infty} \frac{1}{j} \sum_{i=0}^{j-1} E_i = 0\,,$$

with probability 1. Combining the above we have

$$\begin{aligned}\sigma_l(q(s)) - \varepsilon &= \lim_{k\to\infty} \frac{1}{k} \sum_{j=\lfloor kt \rfloor}^{\lfloor ks \rfloor} \overline{R}_l(j) \\ &= \lim_{k\to\infty} \frac{1}{k} \sum_{j=\lfloor kt \rfloor}^{\lfloor ks \rfloor} \mathbb{E}[\overline{R}_l(j) \,|\, Q(k)] \\ &= \lim_{k\to\infty} \frac{1}{k} \sum_{j=\lfloor kt \rfloor}^{\lfloor ks \rfloor} \sigma_l(Q(k)) \,.\end{aligned}$$

Using (3.88) we have

$$(s-t)(\sigma_l(q(t)) - 2\varepsilon) \le \overline{\sigma}_l(s) - \overline{\sigma}_l(t) \le (s-t)(\sigma_l(q(t)) + 2\varepsilon)$$

for $s \in [t, t+u]$. Since we assume that $\overline{\sigma}_l$ is differentiable at t, we have

$$\sigma_l(q(t)) - 2\varepsilon \le \frac{\mathrm{d}}{\mathrm{d}t}\overline{\sigma}_l(t) \le \sigma_l(q(t)) + 2\varepsilon \,.$$

Since this is true for any $\varepsilon > 0$ we therefore have (3.79). The proof of (3.80) is similar and so we omit it here for brevity.

Next we consider (3.81). If $q_l(t) > 0$, then there exists a positive u such that for all $s \in [t, t+u]$, $q_l(s) > 0$ also. This implies that for all sufficiently large k, the backlog $Q_l(\lfloor ks \rfloor)$ at station l is larger than 1 for all $s \in [t, t+u]$. Therefore, the available service for station l will be fully utilised between $\lfloor kt \rfloor$ and $\lfloor kt + u \rfloor$. We thus have

$$\overline{R}_l(\lfloor ks \rfloor) - \overline{R}_l(\lfloor kt \rfloor) = \overline{D}_l(\lfloor ks \rfloor) - \overline{D}_l(\lfloor kt \rfloor)$$

for all $t \le s \le t+u$. Dividing both sides by k and taking $k \to \infty$ we have

$$\overline{\sigma}_l(s) - \overline{\sigma}_l(t) = \overline{d}_l(s) - \overline{d}_l(t) \,,$$

for all $t \le s \le t+u$, which implies $\frac{\mathrm{d}}{\mathrm{d}t}\overline{d}_l(t) = \frac{\mathrm{d}}{\mathrm{d}t}\overline{\sigma}_l(t)$. By combining with (3.79) we therefore have (3.81).

Equations (3.82) and (3.83) follow from the equations $\overline{D}_l(t) = \sum_{r:l\in r} \overline{D}_{l,r}(t)$ and $Q_l(t) = \sum_{r:l\in r} X_{l,r}(t)$ by taking the limit $k \to \infty$ for each respectively.

Finally, by using the queue-evolution equation (3.70) and taking limits as $k \to \infty$ we have

$$\frac{\mathrm{d}}{\mathrm{d}t}x_{l,r}(t) = \lambda_{l,r} + \frac{\mathrm{d}}{\mathrm{d}t}\overline{\mu}_{l_-^{(r)},r}(t) - \frac{\mathrm{d}}{\mathrm{d}t}\overline{\mu}_{l,r}(t) \,.$$

Using (3.80) we have (3.84). □

The following publication has been incorporated as Chapter 4.
1. [74] M. Mandjes, **B. Patch**, and N. S. Walton. Detecting Markov chain instability: A Monte Carlo approach, *Stochastic Systems*, 7.2 (2017). pp. 48–62.

Contributor	Statement of contribution	%
Michel Mandjes	writing of text	33
	proof-reading	33
	supervision, guidance	50
	theoretical derivations	33
	preparation of figures	33
	initial concept	33
Brendan Patch	writing of text	33
	proof-reading	33
	theoretical derivations	33
	preparation of figures	33
	initial concept	33
Neil Walton	writing of text	33
	proof-reading	33
	supervision, guidance	50
	theoretical derivations	33
	preparation of figures	33
	initial concept	33

CHAPTER 4

Detecting Markov chain instability: a Monte Carlo approach

4.1 Introduction

The stability of a Markov chain is arguably among its most important properties. For example, in queueing applications it offers the guarantee that service has been sufficiently provisioned to cope with the load imposed on the network in the long run. For this reason the assessment of the stability of Markov chains has long been an area of intense research. The objective is often to determine the set of parameter values for which the system's state does not diverge, referred to as the *stability region*, of a Markov chain. For many relatively standard Markov chains the stability region is easily expressed in terms of quantities related to the transition probabilities. However, despite a substantial and growing literature, for a large class of systems determining the stability region has appeared a subtle and highly non-trivial task. Importantly, various (at first sight) counterintuitive results have been found; in particular, for specific queueing models 'naïvely conjectured' conditions turn out to be insufficient to ensure stability.

More specifically, initial results, for example those by Jackson [75], Baskett *et al.* [76], and Kelly [77], suggested that the stability of queueing networks would be determined by the network's *subcritical region* (i.e., the set of parameters for which the nominal load at each queue is less than 1). This conjecture was later proven incorrect by a series of counterexamples that showed instability can occur with subcritical parameters when seemingly benign work conserving rules are applied. Early examples include those of Lu and Kumar [18], Rybko and Stolyar [17], and Kumar and Seidman [78]. In these examples the instability is typically essentially caused by the priority rules that apply between the customer classes. Similar effects can, however, be constructed in first-in first-out queueing networks with customer classes that have strongly differing mean service requirements, see for example Bramson [79]. It was thus realized that, at first sight counterintuitively, *decreasing* the mean service requirement of certain job classes can in fact induce instability. As a consequence the stability region need not be monotone (nor convex) in its parameters. For example, Bordenave *et al.* provide an instance of a non-convex stability region in [80]. There are various other examples of queueing networks with unusually shaped stability

regions. In [81] MacPhee *et al.* provide an example with a 'thick null recurrent set', in [82] Baccelli and Bonald show that for certain TCP models the stability region is of a fractal nature, and in [83] Nazarathy *et al.* investigate a case where the stability region is conjectured to consist of disjoint parts.

To avoid determining the stability regions of queueing networks on a case-by-case basis, various general approaches have been proposed. Perhaps the most straightforward among these amounts to determining the invariant measure of the number of customers; when this allows a normalization, then a stationary distribution exists. This approach works for a set of classical models, relying on concepts such as product form and (quasi-)reversibility [2], but unfortunately not for many (sometimes just slightly more complex) other systems.

An arguably more robust approach to determining stability is to construct an appropriate Lyapunov function, and then apply the Foster-Lyapunov theorem. Along these lines Tassiulas and Ephremides, for example, use a quadratic Lyapunov function to find a series of policies which are stable in a wide variety of settings [13]. Constructing an appropriate Lyapunov function is often specific to the application at hand, but the approach can be simplified by studying the fluid model associated with the queueing network. Such a fluid approach was first described by Rybko and Stolyar [17] and was developed in a general form by Dai [21]; a textbook treatment is presented in Bramson [84]. Importantly, for specific models this approach can help determine the conditions under which there is stability, but it does not instantly provide a stability condition for a given network at hand. Thus far, no general framework has been developed that is capable of deciding whether a given Markov chain is stable or not.

The objective of this chapter is to develop a general simulation based approach to determining when a given Markov chain should be classified as unstable. A first paper that considers this approach is [85]. Then, concurrently to our work, [86] proposes a simulation based method for determining the stability region of a multiclass queueing network with respect to its arrival rate, when it is possible to verify that the stability region satisfies particular stochastic monotonicity properties. Given the variety of models and counter-examples discussed above, we place importance on the generality of settings to which our algorithm is suitable. To apply our algorithm we do not place structural assumptions on the stability region, we do not restrict parameters of interest, and we do not restrict the mechanism from which the simulations are derived. We focus on providing theoretical guarantees on the performance of our method for a broad class of models where simulations can exhibit either positive or negative drift.

In particular, rather than specific parameter choices, we are interested in the stability classification of parameter *sets*. The distinguishing features are: (i) that the methodology can be easily used for a relatively broad class of systems, and (ii) that the technique is based on Monte Carlo simulation. Clearly, it is straightforward to develop a simulation-based method that can speculatively test stability for a single parameter setting. It is nevertheless far from obvious how an algorithm should be set up that can identify whether a system is unstable for any of the parameter values within a given set.

This chapter resolves this issue by proposing a *simulated annealing* [87] based algorithm that systematically searches the parameter set, and determines whether it contains a subset of positive measure consisting only of unstable parameter values. Instead of having to perform a series of simulations to answer the stability identification question, our algorithm performs a single simulation run of a process that encompasses both the queueing network and the parameter set, which is provably capable of finding positive measurable subsets of unstable parameters. That is, the output of our algorithm is a

statistical statement that provides explicit asymptotic performance guarantees. We view our work as a substantive pioneering study on the *simulation based* computation of the stability region of Markov chains.

The framework we propose has the major advantages over existing ones of being broadly applicable and relying only on mild modelling assumptions. Our method provably provides the correct outcome if the Markov chain has bounded increments. Another significant advantage of the approach is that the annealing algorithm can essentially be performed separately from the simulation of the queueing network; as a consequence, the program can be organized with an inner loop (simulating the queueing network with given parameter values using a rather complex simulator) and an outer loop (simulating the annealing step). It thus enables us to computationally determine the stability region for (i) relatively straightforward models with non standard features for which this has not been identified in closed form, but also for (ii) larger, realistic models capturing application-specific details.

We now proceed by providing an informal description of the setting we consider as well as our algorithm. The key object in this chapter is the collection of Markov chains

$$\big((X_k^{(\lambda)})_{k\geq 0} : \lambda \in \mathcal{L}\big),$$

each of them evolving on the state space $\mathcal{X}$, where $\mathcal{L} \subset \mathbb{R}^I$ is a compact set of parameter values. For instance, $\lambda \in \mathcal{L}$ could parametrize the arrival rates of a queueing network consisting of I queues. Our algorithm detects if there is a subset $\overline{\mathcal{L}} \subset \mathcal{L}$ of positive measure for which the Markov chains $(X^{(\lambda)} : \lambda \in \overline{\mathcal{L}})$ are unstable. Importantly, for reasons that will become clear, we use a definition of 'stability' that differs slightly from those in common use. We essentially define stability of an individual chain through a Lyapunov drift condition imposed on a function f. For example, $f(x)$ could give the total number of jobs in the queueing network when it is in state x. Informally speaking, if for f the process $f(X^{(\lambda)})$ has negative drift above some finite threshold, then we call $X^{(\lambda)}$ f-stable. Alternatively, if $f(X^{(\lambda)})$ has positive drift above some finite threshold, then we call $X^{(\lambda)}$ f-unstable. If there exists a $\overline{\mathcal{L}} \subset \mathcal{L}$ of positive Lebesgue measure such that $X^{(\lambda)}$ is f-unstable for all $\lambda \in \overline{\mathcal{L}}$, then we call $\mathcal{L}$ f-unstable, and otherwise we call $\mathcal{L}$ stable.

The main idea behind the algorithm is that it generates a discrete time process with state space $(\mathcal{X}, \mathcal{L})$. Given an initial state (x, λ), a new parameter proposal γ is chosen uniformly from $\mathcal{L}$. The Markov chain $X^{(\gamma)}$ then evolves for $\tau(x)$ time units starting from initial state x. In our implementation $\tau(x)$ is chosen to be proportional to $f(x)$. Denoting $X^{(\gamma)}_{\tau(x)} =: y$, the next state of the bivariate process is subsequently chosen by comparing the proposed state (y, γ) with the current state (x, λ) according to the Metropolis rule:

$$(x', \lambda') = \begin{cases} (y, \gamma) & \text{with probability} \quad \exp(\eta\,[f(y) - f(x)]_-), \\ (x, \lambda) & \text{otherwise.} \end{cases} \tag{4.1}$$

Here $[z]_- := \min\{0, z\}$ and η is a positive tuning parameter for the algorithm.

The above iteration is motivated by the simulated annealing algorithm initially proposed by Kirkpatrick, Gelatt, and Vecchi [87]. The main advantage of this type of update is the relative generality of optimization problems that it can provably handle, while still being superior to exhaustive search methods. A key distinction between our method and the typical implementation of simulated annealing is that our cooling schedule is achieved using a combination of the fixed parameter η and the parameter $\tau(x)$, that varies with

the state of the Markov chain.

In addition to the global search algorithm just described, we will also study a local search version. This version is different in two respects. Firstly, the new parameter proposal γ is sampled uniformly from the neighborhood of the current parameter, a set we denote by $\mathcal{B}_\lambda$. Secondly, we allow $X^{(\lambda)}$ and $X^{(\gamma)}$ to evolve for $\tau(x)$ time units starting from initial state x, let $X^{(\lambda)}_{\tau(x)} := y'$ and then apply the above Metropolis rule with x replaced by y'. Our key theorems apply to both versions of the algorithm and we explore differences in performance of the two methods through examples.

After having pointed out how the algorithm works, we now provide results that separate its sample paths into either the stable or unstable regimes. Let $S_k = (Y_k, \Lambda_k)$ be the state of the bivariate process achieved after k iterations of the above rule (4.1) and let T_k be the total amount of time that the algorithm has run for by the kth iteration. The first main theoretical contribution of this chapter, later stated formally in Theorem 6, shows under mild conditions on X, that if there does *not* exist a subset of unstable parameters $\overline{\mathcal{L}} \subset \mathcal{L}$ with positive Lebesgue measure, then almost surely

$$\lim_{k\to\infty} \frac{f(Y_k)}{T_k} = 0\,. \tag{4.2}$$

Importantly, the second main theoretical contribution of this chapter, later stated formally in Theorem 7, shows that there exists a true 'dichotomy' since if there does exist an unstable set of parameters $\overline{\mathcal{L}} \subset \mathcal{L}$ with positive Lebesgue measure, then the process $(Y_k : k \in \mathbb{N}_0)$ diverges, in the sense that, almost surely

$$\liminf_{k\to\infty} \frac{f(Y_k)}{T_k} > 0\,. \tag{4.3}$$

The bound (4.3) relies on a martingale argument in combination with an application of the Azuma–Hoeffding inequality. The bound (4.2) is proven by a coupling argument: as it turns out, in the stable situation the process $f(Y)$ can be majorized by a Markov chain $(W_k : k \in \mathbb{N}_0)$ that has an asymptotic drift of zero. An advantage of the coupling approach used to prove (4.2) is that the Markov process W is easily simulated, and can therefore be used to provide probabilistic bounds on the likelihood of instability. We use this approach to perform rigorous statistical tests for instability of the underlying set $\mathcal{L}$. There are many potential approaches to the adaptation of our theoretical results to a practical test for instability. The approach we suggest is to compare $f(Y_k)$ with the quantiles of W_k. Specifically, we let the process $f(Y)$ evolve according to the rule given in (4.1) until the total number of steps in $\mathcal{X}$ taken between steps of Y exceeds some predetermined level k^*, at which point we compare $f(Y_k)$ with the $1-\alpha$ quantile of W_k, with α being the desired confidence level. If $f(Y_k)$ exceeds this quantile then we obtain a strong rigorous statistical statement of instability, whereas otherwise we fail to reject the 'null hypothesis' of stability.

We provide five example applications of our algorithm. We apply it to a system consisting of a set of parallel queues studied by Tassiulas and Ephremides in [88], a tandem queueing system, the celebrated Rybko–Stolyar network [17], a network of input queued switches studied by Andrews and Zhang in [89], and a broken diamond random access network (RAN) recently studied by Ghaderi et al. in [90].

Since the stability region is well known for the parallel and tandem systems, these are ideal examples on which to verify that the algorithm performs as desired. We use

the tandem system to show that although our results are in a discrete time setting, we are still able to effectively study continuous time systems using a jump chain associated with the process. Additionally, this network also highlights that we are able to test multidimensional parameter sets for instability, and suggests that we are able to relax the Markov assumption. The Rybko–Stolyar network is a popular example of a system with oscillating queue sizes. Not only does our analysis confirm existing theoretical results that give sufficient conditions for stability of this system, it also provides a statistically rigorous statement that these conditions are also necessary. The network of input queued switches allows us to show that our algorithm provides interesting results for systems with high dimensional state spaces and complex dynamics. In addition, we are able to use our methodology to show that this is an example of a system where the longest queue first policy is not maximally stable. Our final example, the RAN of Ghaderi et al., is currently a hot topic of research in the applied probability community. This system exhibits oscillatory queue size sample path behavior reminiscent of the Rybko–Stolyar network, but in a higher dimensional setting. We are able to expand on the theoretical results of [90] by providing more specific (statistical) information about which parameter sets are unstable. Throughout this section results are given in terms of both the global and local versions of the algorithm. For some of the models (parallel, tandem, Rybko–Stolyar), the local algorithm appears to perform better, while for others (switches, RAN) the global algorithm appears to be superior.

The remainder of this chapter is structured as follows. In Section 4.2 we give a formal description of our framework and the assumptions imposed. Section 4.3 presents the algorithm and states our main results, i.e., Theorem 7 and Theorem 6. In Section 4.4 detailed proofs are given (of our main results, propositions, and lemmas). We then provide a range of case studies in Section 4.5 that demonstrate the algorithm's potential. Section 4.7 presents concluding remarks as well as an outlook on future research.

4.2 Framework

In this section we present the set-up considered in the chapter. The object of study is the irreducible Markov chain $X^{(\lambda)}$ that is parametrized by $\lambda \in \mathcal{L}$; these parameters can, for example, be thought of as the arrival or service rates in a queueing network. It is assumed throughout that $\mathcal{L}$ is a compact subset of $\mathbb{R}_0^I$, for some $I \in \mathbb{N}$, with finite positive Lebesgue measure (which is denoted $|\mathcal{L}|$). The Markov chain, which may represent the evolution of the population of a queueing network, attains values in $\mathcal{X} := \mathbb{N}_0^J := \{0, 1, \dots\}^J$, for some $J \in \mathbb{N}$.

As pointed out in the introduction, the main goal of this chapter is to devise a procedure that identifies if a parameter set contains any unstable parameters. Put more precisely, the algorithm verifies whether or not there is a subset $\overline{\mathcal{L}}$ of $\mathcal{L}$ such that for all $\lambda \in \overline{\mathcal{L}}$ the associated Markov chain is *unstable*.

Further, for each $\lambda \in \mathcal{L}$, we let $\mathcal{B}_\lambda$ be a neighborhood of λ. As is commonly assumed for local search algorithms, we assume that $\lambda \in \mathcal{B}_\lambda$ and for any $\lambda_1, \lambda_n \in \mathcal{L}$ there is a sequence of neighbourhoods with $\lambda_{k+1} \in \mathcal{B}_k$ for $k = 1, ..., n-1$.

We will work extensively with a Lyapunov function that maps the state of the Markov chain to a non-negative real number, that is a continuous, positive definite function $f : \mathcal{X} \to [0, \infty)$ such that $f(x) > 0$ for all $x \neq 0$, and has continuous first-order partial derivates at every point of $\mathcal{X}$. For queueing applications, f usually has strictly positive first-order partial derivates. For example, $f(x)$ could represent the *sum* of the queue sizes

within a network (that is, the total network population). We assume that f is unbounded in the sense that

$$\liminf_{||x||\to\infty} f(x) = \infty\,.$$

It is assumed throughout that for all λ the process $f(X^{(\lambda)})$ has bounded increments, implying there exists a constant $\phi_f > 0$, independent of x, such that

$$\left| f(X_{k+1}^{(\lambda)}) - f(X_k^{(\lambda)}) \right| \le \phi_f\,. \tag{4.4}$$

We now provide the formal definitions of stability and instability, as used in this chapter.

Definition 15. *Given f, we say that the set of parameters $\mathcal{L}$ is* f-stable *if there exists $\delta > 0$, $\sigma > 0$, and $\kappa > 0$ such that*

$$\mathbb{E}\left[f(X_k^{(\lambda)}) - f(X_0^{(\lambda)}) \,\middle|\, X_0^{(\lambda)} = x \right] \le -\delta\,\sigma \tag{4.5}$$

for all x such that $|x| \ge \kappa$, for all $\lambda \in \mathcal{L}$, and all $k \ge \sigma$.

Similarly, the set of parameters $\mathcal{L}$ is f-unstable *if there exists a set $\overline{\mathcal{L}} \subset \mathcal{L}$ of positive measure, $\delta > 0$, $\sigma > 0$, and $\kappa > 0$ such that*

$$\mathbb{E}\left[f(X_k^{(\lambda)}) - f(X_0^{(\lambda)}) \,\middle|\, X_0^{(\lambda)} = x \right] \ge \delta\,\sigma \tag{4.6}$$

for all x such that $|x| \ge \kappa$, for all $\lambda \in \overline{\mathcal{L}}$, and all $k \ge \sigma$.

For a given value of λ, the conditions (4.5) and (4.6) are Lyapunov conditions for which one can obtain positive recurrence or transience of the Markov chain $X^{(\lambda)}$, respectively (see for instance [91]). We remark that a countable state space Markov chain is positive recurrent if and only if there exists a Lyapunov function f for which it is f-stable, see Meyn and Tweedie [92, Theorem 11.0.1]. In our definition of f-stable we consider a *set* of Markov chains for which the same choice of f positive recurrence holds. In this sense, the definitions of 'stable' and 'unstable' then ask whether or not the Markov chains $X^{(\lambda)}$ are positive recurrent for parameters λ in $\mathcal{L}$. For our simulations we use the L^1 norm, i.e. the sum of queue sizes, though of course other functions might be considered.

We further remark that if a fluid limit, $\overline{f}(\overline{X}^{(\lambda)})$, exists for each (rescaled) process,

$$\left(\frac{f(X_{\lfloor kt \rfloor}^{(\lambda)})}{k} : t \ge 0 \right), \qquad \lambda \in \mathcal{L}\,,$$

then the above conditions (4.5) and (4.6) imply, respectively, that

$$\frac{\mathrm{d}\overline{f}(\overline{X}^{(\lambda)}(t))}{\mathrm{d}t} \le -\delta \qquad \text{and} \qquad \frac{\mathrm{d}\overline{f}(\overline{X}^{(\lambda)}(t))}{\mathrm{d}t} \ge \delta$$

for $\overline{X}^{(\lambda)}(t) > 0$. In other words, (4.5) and (4.6) respectively imply fluid stability and fluid instability (see for instance [79]). In general, fluid stability and instability are not equivalent to the positive recurrence and transience of an underlying Markov process. Nevertheless, the Lyapunov analysis of fluid models remains one of the most widely de-

ployed and established devices used to determine the positive recurrence and transience of Markov processes. Similarly, our work provides a broadly applicable technique that may be used to determine the positive recurrence and transience of families of Markov processes.

Now that we have introduced our framework, the next section describes our algorithm, as well as the main results upon which the algorithm is based.

4.3 Implementation and main results

In this section we explicitly give our algorithm and provide a detailed discussion of the choices underlying it. We then give in Theorem 6 and Theorem 7 our main theoretical contribution. We follow this up with a suggested method of using our results to implement actual tests for instability.

4.3.1 Algorithm

We now describe our algorithm for identifying whether a parameter set is unstable. Our approach is based on the principle of searching the relevant parameter set for a parameter choice that maximizes the drift of the Markov process under consideration. As such, well known optimization algorithms provide an ideal source of inspiration for potential methods. As previously mentioned, the approach taken in this chapter is based on the well known simulated annealing optimization algorithm. While many other optimization techniques have found acceptance through testing against well known 'hard' problems, the simulated annealing algorithm has shown itself to be amenable to rigorous results on performance guarantees.

In this section we provide a detailed description of our algorithm, and through the use of two theorems provide guarantees on its asymptotic performance. A key advantage of this strong theoretical grounding is that the machinery used to provide these theoretical guarantees also allows us to develop a hypothesis test that outputs a statistical statement of whether or not a Markov chain is stable given a particular parameter set. In this section we also include an illustration of the algorithm and its output in the context of a single server discrete time queueing system. In later sections we demonstrate the algorithm's potential through a series of experiments concerning more complex systems.

We assume that $\tau(x) = c\,f(x) + d$, where c, $d \in (0, \infty)$ are chosen by the algorithm's user. Note that this implies $\tau(x) \to \infty$ as $|x| \to \infty$ and that τ has bounded increments. Finally, let T_k give the time that our chain has been running for at the k-th step, that is,

$$T_k = \sum_{i=0}^{k-1} \tau(Y_i)\,.$$

We now have all of the machinery needed to give both versions of our algorithm.

Algorithm 16. *Global search algorithm:*
Initialize: Set $k = 1$, $T_0 = 0$, *choose* Y_0 *from* $\mathcal{X}$, *and* Λ_0 *from* $\mathcal{L}$.

(i) For $x = Y_{k-1}$ *and* $\tau = \tau(Y_{k-1})$, *set* $T_k = T_{k-1} + \tau$ *and sample* $\gamma \sim$ *Uniform*$(\mathcal{L})$.

(ii) Sample $y = X_\tau^{(\gamma)}$ *conditional on* $X_0^{(\gamma)} = x$.

(iii) For $\lambda = \Lambda_{k-1}$*, set*

$$(Y_k\,, \Lambda_k) = \begin{cases} (y,\, \gamma) & \textit{with probability} \quad \mathrm{e}^{\eta\,[f(y)-f(x)]_-}, \\ (x,\, \lambda) & \textit{otherwise.} \end{cases} \tag{4.7}$$

(iv) If stopping condition *is met, then stop, else set* $k = k{+}1$ *and return to (i).*

As outlined in the introduction, each step of the global search algorithm compares x, as sampled in the previous step, with a new value y, sampled using a uniformly at random selected parameter γ from $\mathcal{L}$ with runtime $\tau(x)$ and initial state x. The state is then updated according to the Metropolis rule (4.7).

Recalling from Section 4.2 that $\mathcal{B}_\lambda$ is a neighbourhood of λ in $\mathcal{L}$, the local search version operates as follows.

Algorithm 17. *Local search algorithm:*
Initialize: Set $k = 1$*,* $T_0 = 0$*, choose* Y_0 *from* $\mathcal{X}$*, and* Λ_0 *from* $\mathcal{L}$*.*

(i) For $x = Y_{k-1}$*,* $\lambda = \Lambda_{k-1}$ *and* $\tau = \tau(Y_{k-1})$*, set* $T_k = T_{k-1} + \tau$ *and sample* $\gamma \sim$ *Uniform*$(\mathcal{B}_\lambda)$*.*

(ii) Sample $x' = X_\tau^{(\lambda)}$ *conditional on* $X_0^{(\lambda)} = x$*.*

(iii) Sample $y = X_\tau^{(\gamma)}$ *conditional on* $X_0^{(\gamma)} = x$*.*

(iv) Set

$$(Y_k\,, \Lambda_k) = \begin{cases} (y,\, \gamma) & \textit{with probability} \quad \mathrm{e}^{\eta\,[f(y)-f(x')]_-}, \\ (x',\, \lambda) & \textit{otherwise.} \end{cases} \tag{4.8}$$

(v) If stopping condition *is met, then stop, else set* $k = k{+}1$ *and return to (i).*

The local search algorithm compares states $(x',\, \lambda)$ and $(y,\, \gamma)$ where x' is sampled by running the current parameter λ for a further τ steps and $(y,\, \gamma)$ is sampled by running a neighbouring parameter $\gamma \in \mathcal{B}_\lambda$ for the same number of steps. These states are then compared according to the Metropolis rule (4.8). We discuss choices for the stopping condition further below.

As we will discuss in more detail, the relative performance of the global search and local search differs depending on the model and setting to which they are applied. Under general modelling assumptions both algorithms converge to a behaviour that only accepts unstable parameters in $\mathcal{L}$. The Global Search Algorithm proposes parameters uniformly at random and thus asymptotically will only accept parameters uniformly at random in the unstable set $\overline{\mathcal{L}}$. This is useful if one wants to identify the region of instability, in addition to determining if instability occurs. The Local Search Algorithm proposes two neighbouring parameters and compares them simultaneously. In this way the Local Search Algorithm applies a hill-climbing heuristic. In this sense it is more aggressive in approaching regions of instability, but will not identify the entire unstable region.

When analyzing Algorithm 16, we assume that $\mathcal{L}$ is a general measurable set and that $\overline{\mathcal{L}}$ is a set with positive Lebesgue measure, while for the local search Algorithm 17 we place some restrictions. We assume

Assumption 18. *When analyzing the local search algorithm we assume that $\mathcal{L}$ is a finite countable set where for each $\mathcal{L}' \subset \mathcal{L}$ either $\mathcal{L}'$ is unstable or $\mathcal{L}'$ is stable, according to Definition 15. Further, we assume that there exists a state x_0 where*

$$\mathbb{P}(X_1^{(\lambda)} = x_0 \,|\, X_0^{(\lambda)} = x_0) > 0\,. \tag{4.9}$$

In a queueing setting x_0 may, for example, correspond to a state where all queues are idle.

An important feature of our work is that we do not place structural conditions on $\mathcal{L}$ such as convexity or monotonicity. Since examples of Markov processes violating these conditions frequently occur in both theory and practice, by avoiding such conditions our work is widely applicable. Another key feature is that we do not assume knowledge of the process generating each sample path is available, we only require samples of the state description in response to parameter choices. This further extends the set of models that may be analysed using our method, since practical simulators (although Markovian) are often not generated from a simple closed form Markovian descriptor (transition matrix or infinitesimal generator), but rather come in the form of a 'black box' that provides outputs in response to parameter inputs.

Note that we have not provided an explicit stopping condition for the algorithm yet. Since our results are asymptotic in the number of steps k, it may be sensible to run the algorithm until some large k, chosen based on CPU time limitations. An alternative may be to dictate a particular total budget of time that the algorithm may evolve in $\mathcal{X}$. To do this, choose a k^* and run the algorithm until $T_k > k^*$. In either case it may not be obvious whether the sample path belongs to the stable or unstable regimes, an issue that we address with a test for instability in Section 4.3.3.

We now briefly address some of the choices we made in the design of the algorithm. Firstly, note that the τ function we introduced has replaced the cooling schedule from the traditional simulated annealing algorithm. The functional form of τ ensures that when Y_k is large the subsequent Λ_{k+1} proposal is given an increased opportunity to demonstrate that it has higher drift. Since a large Y_k hints that an unstable parameter choice has been recently chosen, this helps to ensure that CPU budget is expended comparing parameter choices which appear to be unstable.

The conditioning in step (ii) of the algorithm on $X_0^{(\gamma)} = x$, rather than starting each new sample from $X_0^{(\gamma)}$ equal to zero, is intentionally designed to allow the system to build up to a size where instability properties become evident. That is, as per Definition 4.6, the drift properties we are seeking only become evident after $|x| > \kappa$ has occurred. Forcing the system to reach $|x| > \kappa$ in a single $X_{\tau(x)}^{(\gamma)}$ sample may result in the algorithm inefficiently repeating 'burn in' time.

4.3.2 Main results

Our main theoretical contributions are Theorem 7 and Theorem 6 below. These demonstrate that the stability of a set $\mathcal{L}$ can be summarised in terms of the asymptotic sample path behaviour of the process $f(Y)/T$. In essence the stability of a parametrised family of Markov processes can be summarized by the stability of a single Markov process, as generated by Algorithm 16 or Algorithm 17.

The main results of this chapter are as follows:

Theorem 6. *If the set $\mathcal{L}$ is stable then, almost surely,*

$$\lim_{k\to\infty} \frac{f(Y_k)}{T_k} = 0\,. \tag{4.10}$$

Theorem 6 shows that when $\mathcal{L}$ is stable, the sample path of $f(Y)/T$ converges to 0.

Theorem 7. *If the set $\mathcal{L}$ is unstable then, almost surely,*

$$\liminf_{k\to\infty} \frac{f(Y_k)}{T_k} > 0\,. \tag{4.11}$$

Theorem 7 shows that when $\mathcal{L}$ is unstable, the sample path of $f(Y)/T$ eventually never returns to 0. In practical use it is $f(Y_k)/T_k$, for some large k, that is observed, rather than its limiting value. It is therefore not possible to directly apply the theorems. Instead, when $f(Y)/T$ appears to converge to 0, the contrapositive of Theorem 7 provides evidence that the parameter set is not unstable. Conversely, when $f(Y)/T$ appears to diverge, converge to a positive constant, or fluctuate within a set that does not contain 0, the contrapositive of Theorem 6 provides evidence that the parameter set is not stable.

We now include a short example to illustrate these theorems. Consider a simple discrete time queueing system where an arrival occurs at the beginning of each time slot with probability $p \in [0,\, 1]$, and then subsequently, if the queue is non-empty, a service occurs with probability 0.5. Clearly, so long as the queue is non-empty the expected change in queue size between time periods is $p - 0.5$. Hence, for $p < 0.5$ the system is L^1-stable with $\kappa = \sigma = 1$ and $\delta = 0.5 - p$. Figure 4.1 illustrates Theorem 7 and Theorem 6 using the sample path behaviour of $f(Y)/T$ for this simple system. We have taken $\eta = 1$ and $\mathcal{B}_\lambda$ to be the intersection of a ball of radius 0.01 around λ with $\mathcal{L}$. The sample path corresponding to p sampled from $\mathcal{L} = [0,\, 0.4]$ appears to converge towards 0, providing evidence that this set is not unstable. Similarly, the sample path corresponding to p sampled from $\mathcal{L} = [0,\, 0.6]$ appears to remain constant at approximately 10^{-2} in the global case and appears to diverge in the local case, providing evidence that this set is not stable.

We remark that the behaviour of the unstable sample path substantially differs between the global and local versions of the algorithm. In the global case the unstable sample path quickly separates from the stable sample path and appears to tend towards some constant value. For the local algorithm, however, the stable and unstable sample paths appear highly similar until suddenly the unstable sample path rapidly increases. Since the time that this divergence occurs is random and likely to depend strongly on the initial condition, this suggests that if n is not large enough, the local algorithm may perform very poorly, however for n large it may perform vastly better.

It is not necessarily clear in finite time if a sample path of $f(Y)/T$ belongs to the regime of Theorem 6 or Theorem 7. In the next subsection we address this issue by presenting a test for instability.

4.3.3 A test for instability

We now provide a method to test, statistically, whether or not a parameter set is unstable. Here one could consider a null-hypothesis which states that the parameter set is stable for some given δ, cf. (4.5). Given this and the simulated model, we can construct a closed form family of random variables $Z(w)$, $w \geq 0$ (given by Lemma 21 in Section 4.4.1) such

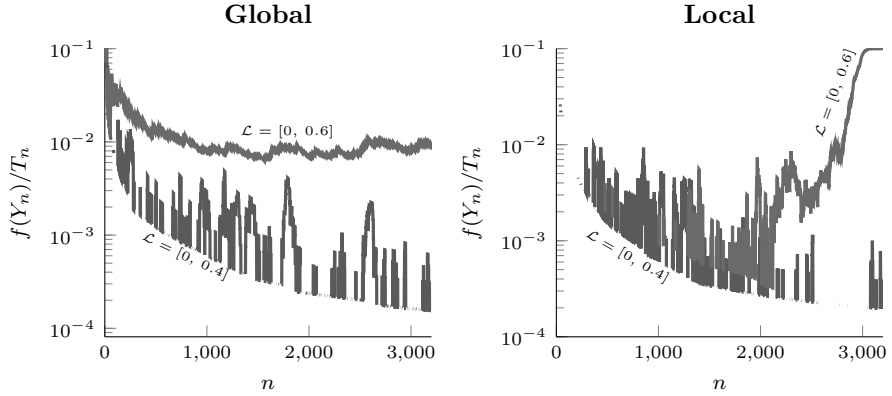


Figure 4.1: Comparison of $f(Y)/T$ sample paths for stable ($\mathcal{L} = [0,\, 0.4]$) and unstable ($\mathcal{L} = [0,\, 0.6]$) parameter sets when the global and local versions of the algorithm are applied to a simple queue.

that $Z(w)$ stochastically majorises the increments of $f(Y_k)$. With this choice of $Z(w)$, we can then define a Markov chain $(W_k : k \in \mathbb{N}_0)$ according to the recursion

$$W_k = W_{k-1} + Z(W_{k-1})\,. \tag{4.12}$$

The following proposition will be proven to show that there is a coupling where the Markov chain $(W_k : k \in \mathbb{N}_0)$ stochastically dominates $(f(Y_k) : k \in \mathbb{N}_0)$.

Proposition 19. *For stable $\mathcal{L}$, when $f(Y_0) \leq W_0$, there exists a coupling between $(Y_k : k \in \mathbb{N}_0)$ and $(W_k : k \in \mathbb{N}_0)$ such that*

$$f(Y_k) \leq W_k, \qquad \textit{for all } k.$$

Since Theorem 7 says that $f(Y)$ will diverge in the unstable case, we suggest comparing $f(Y_k)$, as outputted by Algorithm 16, with the quantiles of W_k. In particular, let $q_k^{(\alpha)}$ be such that $\mathbb{P}(W_k < q_k^{(\alpha)}) = 1 - \alpha$. Note that, given a problem instance chosen according to Definition 15 and (4.4) (that is, particular values of ϕ_f, δ, σ, and κ), the quantiles of $q_k^{(\alpha)}$ can be estimated quickly and easily through Monte Carlo simulations of the W process. If $f(Y_k) > q_k^{(\alpha)}$ then we suggest concluding that the parameter set is f-unstable for that problem instance. Otherwise we suggest that there is not enough evidence to make a conclusion either way.

To illustrate this approach we return to the simple example introduced in the previous section. In Figure 4.2 estimated $q^{(0.05)}$ curves with $\delta = 0.05$ and $\delta = 0.01$, $\tau(x) = 0.5\,x + 1$, and $\sigma = \kappa = Y_0 = 1$ are compared with estimated mean curves for the $f(Y)$ process with $\mathcal{L} = [0, \ell]$ for $\ell = 0.6,\ 0.55,\ 0.5,\ 0.45$, and 0.4. As expected the $q^{(0.05)}$ curves bound the mean curves of $f(Y)$ for $\ell < 0.5$, while for $\ell > 0.5$ the mean curves of $f(Y)$ appear to eventually exceed the $q^{(0.05)}$ curve. With reference to Definition 15, δ is the downward drift that a process must exhibit in order to be stable. As can be seen here, it is to be expected that $q^{(0.05)}$ curves generated using a particular δ value bound those generated using higher values of δ. Since we test for a stabilizing drift up to δ, it is desirable to use a δ which is as low as possible. In Section 4.5 we investigate further the trade-off between

simulation run-time and δ that users of our algorithm must keep in mind. Note that the global and local $q^{(0.05)}$ curves are nearly indistinguishable from each other here.

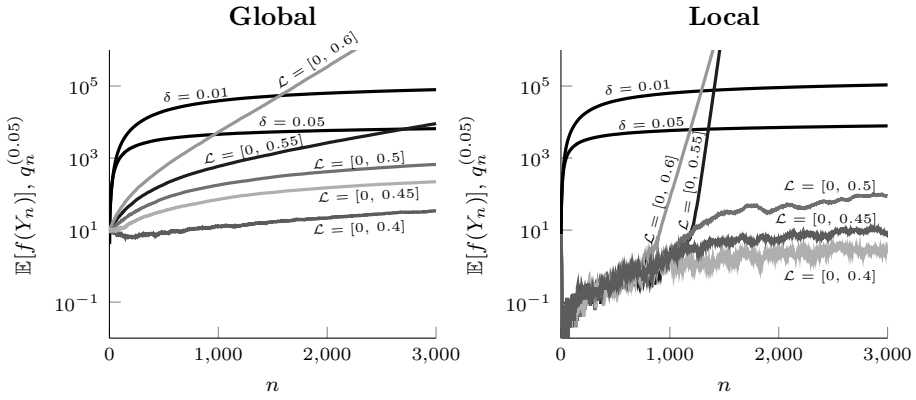


Figure 4.2: Estimated mean curves of $f(Y)$ when $\mathcal{L} = [0, 0.4]$, $[0, 0.45]$, $[0, 0.5]$, $[0, 0.55]$, or $[0, 0.6]$ for a simple queue compared to estimated $q^{(0.05)}$ curves with $\delta = 0.01$ or $\delta = 0.05$.

In some instances, obtaining long sample paths of $f(Y)$ may be a computationally intensive task. We now describe an approach to managing the user's simulation budget, but various other approaches could be taken. In order to achieve a significance level of *at least* α, we propose choosing a simulation budget k^*, to take the first sample point $f(Y_k)$ such that $T_{k+1} > k^*$ and to then compare this $f(Y_k)$ with an estimate of $q_k^{(\alpha)}$. If $f(Y_k)$ exceeds $q_k^{(\alpha)}$ then we suggest rejecting the 'null hypothesis' of stability, and otherwise we suggest concluding that there is not enough evidence to make a conclusion. This is the approach that we take in Section 4.5. Note that this does not involve comparing the 'test statistic' $f(Y_k)$ with its distribution, but rather we compare it with a distribution which is stochastically dominant. Assuming that the $1-\alpha$ quantile estimate for W_k is accurate, asymptotically in k^* the significance level will in fact tend to 0 for all $\alpha > 0$, and never exceed α.

We summarize the above discussion in Algorithm 20, below. Note that this algorithm is just one of many potential extensions of our basic Algorithm 16, for which we give specific theoretical results, and an input to this algorithm is an appropriate $(q^{(\alpha)} : k \in \mathbb{N}_0)$ estimate.

Algorithm 20. *Stability test algorithm:*
Initialize: Set $k = 1$, $T_0 = 0$, *choose* Y_0 *from* $\mathcal{X}$, *and* Λ_0 *from* $\mathcal{L}$.

(i) For $x = Y_{k-1}$ *and* $\tau = \tau(Y_{k-1})$, *set* $T_k = T_{k-1} + \tau$ *and sample* $\gamma \sim$ *Uniform*$(\mathcal{L})$.

(ii) Sample $y = X_\tau^{(\gamma)}$ *conditional on* $X_0^{(\gamma)} = x$.

(iii) For $\lambda = \Lambda_{k-1}$, *set*

$$(Y_k, \Lambda_k) = \begin{cases} (y, \gamma) & \textit{with probability } \mathrm{e}^{\eta[f(y)-f(x)]_-}, \\ (x, \lambda) & \textit{otherwise.} \end{cases} \tag{4.13}$$

(iv) If $T_k + \tau(Y_k) > k^*$, *then proceed to (v), else set* $k = k + 1$ *and return to (i).*

(v) If $f(Y_k) > q_k^{(\alpha)}$*, then conclude* $\mathcal{L}$ *is* f*-unstable.*

In Section 4.4 we prove the results presented above, and then in the Section 4.5 we will demonstrate the algorithm's potential on some more complex systems.

4.4 Proofs

We first prove Theorem 6 in Section 4.4.1, which applies to the stable regime, in the context of the global search algorithm and provide a remark on the minor modifications to this proof that would be needed to show the local search case. We then prove Theorem 7, which applies to the unstable case, for the global search and local search algorithms in Section 4.4.2 and Section 4.4.3 respectively.

4.4.1 Stable parameter set

In this subsection we prove Theorem 6. In what follows, we first give a formal definition of the random variables $Z(w)$. Then, to prove Theorem 6, we require Proposition 19, given in Section 4.3, and Proposition 22, given below. The first of these propositions shows the existence of a process that majorises any Y process generated from a stable parameter set. The second proposition then shows that this majorising process has an asymptotic drift of zero, which leads to the result of the theorem. In order to obtain Proposition 19 we require Lemma 21, given next, and Lemma 27, which is a simple technical lemma that can be found in Section 4.6. Lemma 21 explicitly provides a level dependent random variable that bounds the jumps of the $f(Y)$ process and Lemma 27 gives a useful monotonicity property for these jumps. Proposition 22 is proven by contradiction and depends on Lemma 28 which states that the sequence of random variables defined in Lemma 21 are square integrable and tend to an expectation of zero as the level diverges.

In this section all of the proofs are performed in the context of the global search algorithm, however at the end of the section we remark on the minor modification required to adapt the proof to the local search algorithm context.

We develop a process W that stochastically majorises any $f(Y)$ process generated from a stable parameter set. Recall ϕ_f, δ and σ from Definition 15. We define the function n such that $n(w)$ is the smallest integer such that $\sigma\, n(w) \geq \tau(x)$ when the underlying process is in a state $x = f^{-1}(w)$. We bound the jumps of $f(X^{(\lambda)})$, for a given *stable* λ. The following lemma provides this bound.

Lemma 21. *If* λ *is* f*-stable, then there exists random variables* $(Z(w) : w \geq 0)$ *and a constant* w^* *such that, for all* x *with* $f(x) \geq w^*$*,*

$$\mathbb{P}\left(f(X^{(\lambda)}_{\tau(x)}) - f(x) \geq z \,\middle|\, X_0^{(\lambda)} = x\right) \leq \mathbb{P}(Z(f(x)) \geq z)\,,$$

where, for σ*,* κ *and* δ *as given in* (4.5)*,* $Z(w)$ *is a random variable with distribution*

$$\mathbb{P}(Z(w) \geq z) = \begin{cases} 1 \wedge \Big[\exp\Big(-\dfrac{(z-\alpha_1(w))^2}{2\alpha_2(w)}\Big) + n(w)\exp\Big(-\dfrac{(z-\alpha_3(w))^2}{2\alpha_4(w)}\Big)\Big], & \text{if } z > 0, \\ 1\,, & \text{otherwise.} \end{cases} \tag{4.14}$$

where

$$\begin{aligned} \alpha_1(w) &= \sigma\phi - \sigma n(w)\delta\,, \qquad \alpha_2(w) = (\phi_f + \delta)^2\sigma^2 n(w)\,, \\ \alpha_3(w) &= \sigma\phi - w + \kappa\,, \qquad \alpha_4(w) = \phi_f^2\,\sigma^2 n(w)\,. \end{aligned}$$

The proof of this lemma is straightforward, yet, somewhat technical; a proof is given in Section 4.6. The form of the expression for $\mathbb{P}(Z(w) \geq z)$ given above can be understood as follows. By (4.5) the stable downward drift condition only applies when the chain has run for at least σ steps, so we consider the process on steps of size σ and ensure that we take enough of these steps, $n(w)$, to exceed $\tau(x)$. The maximum is a result of the trivial upper bound on probabilities, and the $1 + n(w)$ exponential terms correspond to a union bound using an equivalent number of applications of the Azuma–Hoeffding inequality.

The downward drift condition requires $|x| > \kappa$, and so we apply Azuma–Hoeffding to different martingales depending on whether the sample path of interest enters the states $\{x : |x| < \kappa\}$ or not. The first exponential term corresponds to sample paths that never enter $|x| < \kappa$, and so the martingale we use does not include κ and has steps which are bounded by $(\phi_f + \delta)\,\sigma$. The remaining exponential terms correspond to sample paths that hit the level κ. We associate with these sample paths a martingale which reflects the fact that for such sample paths there be must be an excursion from κ to $z + f(x)$ that can be stopped just before this excursion occurs. As such these remaining exponentials do not rely on δ.

From Lemma 21 we can prove Proposition 19.

Proof of Proposition 19. The inequality in Lemma 21 bounds the upward movement of the Markov process X_λ. For $w_0 = f(x_0)$, we see that

$$\mathbb{P}(f(Y_1) - f(Y_0) \geq z \,\big|\, Y_0 = x_0) \leq \mathbb{P}(Z(w_0) \geq z), \qquad \forall z \geq 0. \tag{4.15}$$

Namely, if $f(X^{(\lambda)}_{\tau(x_0)}) - f(X^{(\lambda)}_0) > 0$, then $f(Y_1) - f(Y_0) = f(X^{(\lambda)}_{\tau(x_0)}) - f(X^{(\lambda)}_0)$. Therefore the bound (4.15) holds by Lemma 21 for $z > 0$. Further, for $z \leq 0$, the right-hand side of (4.15) is equal to 1, so the bound trivially holds.

Lemma 27, stated and proved in Section 4.6, assists with the coupling of W and Y by providing a monotonicity property for the transitions of W. Specifically, for constants v, w with $w^* \leq v \leq w$, we have that

$$\mathbb{P}(W_1 \geq z \,|\, W_0 = v) \leq \mathbb{P}(W_1 \geq z \,|\, W_0 = w)\,. \tag{4.16}$$

Combining together (4.15) and (4.16), we have that

$$\mathbb{P}(f(Y_1) \geq z \,|\, Y_0 = x_0) \leq \mathbb{P}(W_1 \geq z \,|\, W_0 = w_0) \tag{4.17}$$

whenever $f(x_0) \leq w_0$.

A direct consequence of this inequality is that there is a coupling of $f(Y_k)$ and W_k where, provided $f(Y_0) \leq W_0$, then $f(Y_k) \leq W_k$ for all k. This short, but standard, argument is presented in the next paragraph.

Let

$$\begin{aligned} F_{Y,x_0}(z) &= \mathbb{P}(f(Y_1) \geq z \,|\, Y_0 = x_0)\,, \\ F_{Z,w_0}(z) &= \mathbb{P}(W_1 \geq z \,|\, W_0 = w_0)\,, \end{aligned}$$

and U be an independent uniform $[0,1]$ random variable. The distribution of $F^{-1}_{Y,x_0}(U)$ and $F^{-1}_{Z,w_0}(U)$ are respectively versions of $f(Y_1)$ and W_1 for initial values $Y_0 = x_0$ and $W_0 = w_0$ (see e.g. [93, Section 3.12]). Thus we set $f(Y_1) = F^{-1}_{Y,x_0}(U)$ and $W_1 = F^{-1}_{Z,w_0}(U)$.

Notice that once $f(Y_1)$ is determined, we can extend the coupling to determine Y_1. To do this, we take an independent random variable with distribution

$$\mathbb{P}(Y_1 = y \,|\, f(Y_1), Y_0)\,.$$

Now, from inequality (4.17) it is clear that $F^{-1}_{Y,x_0}(u) \le F^{-1}_{Z,w_0}(u)$ for all values of u. Thus under this coupling

$$f(Y_1) = F^{-1}_{Y,x_0}(U) \le F^{-1}_{Z,w_0}(U) = W_1\,.$$

Continuing inductively, we have $f(Y_k) \le W_k$ for all k, as claimed. □

We now analyze the chain W_k. As the following lemma states, we find that its asymptotic drift is zero, whenever the parameter set $\mathcal{L}$ is stable.

Proposition 22.

$$\limsup_{k\to\infty} \frac{W_k}{k} = 0\,.$$

Proof. It is a straightforward calculation to show that $Z(w)$ is L^2 bounded in w and that $\mathbb{E}Z(w) \to 0$ as $w \to \infty$. This is shown in Lemma 28 in Section 4.6. We analyse the martingale

$$\begin{aligned} M_k &= \sum_{n=1}^{k} \big(Z(W_{n-1}) - \mathbb{E}\big[Z(W_{n-1}) \,|\, W_{n-1}\big]\big) \\ &= W_k - W_0 - \sum_{n=1}^{k} \mathbb{E}\big[Z(W_{n-1}) \,|\, W_{n-1}\big]\,. \end{aligned} \tag{4.18}$$

Since $Z(w)$ is L^2 bounded, M_k is an L^2 martingale (with unbounded variation). Further, such L^2 martingales obey the strong law of large numbers, that is

$$\lim_{k\to\infty} \frac{M_k}{k} = 0\,. \tag{4.19}$$

For instance, see [93, Section 12.14] for a proof.

We therefore have

$$\begin{aligned} \limsup_{k\to\infty} \frac{W_k}{k} &= \limsup_{k\to\infty} \left(\frac{W_0 + M_k}{k} + \frac{1}{k} \sum_{n=1}^{k} \mathbb{E}\big[Z(W_{n-1}) \,|\, W_{n-1}\big] \right) \\ &\le \limsup_{k\to\infty} \frac{W_0 + M_k}{k} + \limsup_{k\to\infty} \frac{1}{k} \sum_{n=1}^{k} \mathbb{E}\big[Z(W_{n-1}) \,|\, W_{n-1}\big] \\ &= \limsup_{k\to\infty} \frac{1}{k} \sum_{n=1}^{k} \mathbb{E}\big[Z(W_{n-1}) \,|\, W_{n-1}\big]\,, \end{aligned} \tag{4.20}$$

where the first equality holds due to (4.18) and the final equality holds due to (4.19).

We now note that the inequality (4.20) can only hold when $\lim_{k\to\infty} W_k/k = 0$. To see this, note that if $\limsup_k W_k/k$ were positive then W_k must diverge. However, as was shown in Lemma 28, we also have that $\mathbb{E}\big[Z(W_{n-1}) \,\big|\, W_{n-1}\big] \to 0$ as $W_{n-1} \to \infty$. Thus the average of these terms must be zero, that is

$$0 = \limsup_{k\to\infty} \frac{1}{k} \sum_{n=1}^{k} \mathbb{E}\big[Z(W_{n-1}) \,\big|\, W_{n-1}\big] \geq \limsup_{k\to\infty} \frac{W_k}{k} > 0\,,$$

which is a contradiction. Thus, $\limsup_{k\to\infty} W_k/k = 0$, as required. □

The proof of Theorem 6 is now an application of Proposition 19 and Proposition 22.

Proof of Theorem 6. For $W_0 = f(Y_0)$ Proposition 19 provides

$$f(Y_k) \leq W_k\,, \qquad \text{for all } k\,.$$

The time increment $\tau(x)$ is bounded below, so $T_k \geq \gamma k$ for some positive constant γ. Hence, Proposition 22 implies

$$\limsup_{k\to\infty} \frac{f(Y_k)}{T_k} \leq \limsup_{k\to\infty} \frac{W_k}{\gamma k} = 0\,,$$

as required. □

We briefly remark one way in which the above argument can be adapted to the local-search case. Firstly, we note that local search (in the worse case) will choose the maximum of two independent simulation runs. Given that both parameters γ and λ are stable and given Lemma 21, we then have that the local search update can be bounded as follows

$$\begin{aligned}\mathbb{P}(f(Y_1) \geq z \,|\, f(Y_0) = x, \lambda, \gamma) &\leq \mathbb{P}(\Big\{f(X_\tau^{(\lambda)}) - f(x) \geq z\Big\} \cup \Big\{f(X_\tau^{(\lambda)}) - f(x) \geq z\Big\}) \\ &\leq \mathbb{P}(Z(f(x)) + Z'(f(x)) \geq 2z)\,.\end{aligned}$$

In the second inequality above, we apply Lemma 21 to obtain two i.i.d. copies of $Z(f(x))$. From this one can see that the result of the proof of Theorem 6 follows by replacing (4.12) with

$$W_k = W_{k-1} + Z(W_{k-1}) + Z'(W_{k-1})$$

for two iid versions of Z. This gives one straightforward way of adapting the proof of Theorem 6. Other methods with tighter bounds are also possible.

4.4.2 Proof of Theorem 7 for the global search algorithm

In this subsection we prove Theorem 7 for the global search algorithm. The proof relies on two lemmas, Lemma 23, and Lemma 24. In Lemma 23 we bound the drift of $f(Y)$ and show that $f(Y)$ can be used to construct a submartingale. This follows from the fact that unstable parameter choices significantly increase the drift, while stable parameter choices do not significantly decrease it. In Lemma 24 we bound the moments of this submartingale. Then, using standard martingale arguments we show that every time

the submartingale exceeds some level, with positive probability it stays above this level forever. Since $f(Y)$ is an irreducible Markov chain our divergence result then follows.

In the following we use the notation $[x]_+ := \max\{x, 0\}$ and $\Delta f(x, y) := f(y) - f(x)$.

Lemma 23. *If $\mathcal{L}$ is unstable, then there exist constants $\kappa \geq 0$ and $a > 0$ such that for all x with $|x| \geq \kappa$*

$$\mathbb{E}\left[f(Y_{k+1}) - f(Y_k) \,|\, Y_k = x\right] \geq a\,\tau(Y_k) > 0\,. \tag{4.21}$$

Proof. Aside from the simulation in $\mathcal{X}$ between Y samples, our algorithm consists of two random steps: (i) the selection of Λ_{k+1}, and (ii) the random state update rule (4.7). Upon conditioning on these two steps the expected change in f can be calculated as follows

$$\begin{aligned}
&\mathbb{E}\left[\Delta(Y_k, Y_{k+1}) \,|\, Y_k = x\right] \\
&= \frac{1}{|\mathcal{L}|}\int_{\mathcal{L}} \mathbb{E}\left[\Delta f(Y_k, Y_{k+1}) \,|\, Y_k = x, \Lambda_{k+1} = \mu\right] d\mu \\
&= \frac{1}{|\mathcal{L}|}\int_{\mathcal{L}} \mathbb{E}\left[\Delta f(X_0^{(\mu)}, X_{\tau(x)}^{(\mu)}) \exp\left(-\eta\left[-\Delta f(X_0^{(\mu)}, X_{\tau(x)}^{(\mu)})\right]_+\right)\right] \mathrm{d}\mu\,.
\end{aligned} \tag{4.22}$$

Denote $p := |\overline{\mathcal{L}}|/|\mathcal{L}| \in (0, 1]$, where $\overline{\mathcal{L}}$ is the set for which X_λ is unstable (cf. (4.6)). Now split the above integral by distinguishing between: (i) $\mu \in \overline{\mathcal{L}}$, and (ii) $\mu \in \mathcal{L} \setminus \overline{\mathcal{L}}$. It is readily verified that for all $z \in \mathbb{R}$ the function $z \mapsto z\exp(-\eta\,[-z]_+)$ satisfies

$$z\exp(-\eta[-z]_+) \geq \max\left\{z, -(\mathrm{e}\,\eta)^{-1}\right\}\,. \tag{4.23}$$

The stated bound is trivial for $z \geq 0$, and for $z < 0$ simply note that $z\exp(-\eta[-z]_+)$ is minimized at $z = -\eta^{-1}$.

For $\mu \in \overline{\mathcal{L}}$ we use the lower bound of z in (4.23),

$$\begin{aligned}
&\frac{1}{|\mathcal{L}|}\int_{\overline{\mathcal{L}}} \mathbb{E}\left[\Delta f\left(X_0^{(\mu)}, X_{\tau(x)}^{(\mu)}\right) \exp\left(-\eta\left[-\Delta f\left(X_0^{(\mu)}, X_{\tau(x)}^{(\mu)}\right)\right]_+\right)\right] \mathrm{d}\mu \\
\geq &\frac{1}{|\mathcal{L}|}\int_{\overline{\mathcal{L}}} \mathbb{E}\left[\Delta f\left(X_0^{(\mu)}, X_{\tau(x)}^{(\mu)}\right)\right] \mathrm{d}\mu \geq p\,\delta\,\tau(x)\,.
\end{aligned} \tag{4.24}$$

In the second inequality above, we apply the assumption that our Markov chain is unstable on $\mathcal{L}$ (cf. (4.6)).

For $\mu \in \mathcal{L} \setminus \overline{\mathcal{L}}$, we use the lower bound of $-(\mathrm{e}\,\mu)^{-1}$ in (4.23). This yields

$$\begin{aligned}
&\frac{1}{|\mathcal{L}|}\int_{\mathcal{L}\setminus\overline{\mathcal{L}}} \mathbb{E}\left[\Delta f\left(X_0^{(\mu)}, X_{\tau(x)}^{(\mu)}\right) \exp\left(-\eta\left[-\Delta f\left(X_0^{(\mu)}, X_{\tau(x)}^{(\mu)}\right)\right]_+\right)\right] \mathrm{d}\mu \\
&\geq -\,(1-p)\,(\mathrm{e}\,\mu)^{-1}\,.
\end{aligned} \tag{4.25}$$

Combining Equation (4.22) with Inequalities (4.24) and (4.25) yields

$$\mathbb{E}\left[\Delta f(Y_k, Y_{k+1}) \,|\, Y_k = x\right] \geq p\,\delta\,\tau(x) - (1-p)\,(\mathrm{e}\,\mu)^{-1}\,. \tag{4.26}$$

Since $\tau(x) \to \infty$ as $|x| \to \infty$. There exists $\kappa > 0$ such that $p\,\delta\,\tau(x) - (1-p)\,(\mathrm{e}\,\mu)^{-1} \geq p\,\delta\tau(x)/2$ for all $|x| > \kappa$. Letting $a = p\,\delta/2$, we have the result. □

Let k_κ be the hitting time for $f(Y)$ on the states $\{x : |x| \leq \kappa\}$. An immediate

consequence of the above proof is that the process

$$F_k := f(Y_{k\wedge k_\kappa}) - a \sum_{i=0}^{(k\wedge k_\kappa)-1} \tau(Y_i)$$

forms a submartingale. This in itself is not sufficient to prove $f(Y_k)$ diverges in the sense of Theorem 7. However, this is possible when we bound the moments of F_k as follows. In the following let $S_k = (Y_k, \Lambda_k)$.

Lemma 24. *If $\mathcal{L}$ is unstable, then there exist an $r > 0$ such that for $|Y_{k-1}| \geq \kappa$*

$$\mathbb{E}\big[\exp\big((-r\,(F_k - F_{k-1})\big)\,\big|\,S_{k-1}\big] < 1\,. \tag{4.27}$$

Proof. The change in the process from F_{k-1} to F_k is achieved by a process with bounded increments. Upon applying the Azuma–Hoeffding inequality, we have that

$$\mathbb{P}\left(F_k - F_{k-1} - \mathbb{E}[F_k - F_{k-1}] \leq -y\,\middle|S_{k-1}\right) \leq \exp\left(-\frac{2\,y^2}{\tau_k \phi_f^2}\right)\,. \tag{4.28}$$

Now consider the sequence of inequalities:

$$\begin{aligned}
&\mathbb{E}\left[\exp\big(-r\,(F_k - F_{k-1})\big)\,\middle|\,S_{k-1}\right] \\
&= \int_0^\infty \mathbb{P}\left(\exp(-r\,(F_k - F_{k-1})) \geq z\middle|S_{k-1}\right)\mathrm{d}z \\
&= \int_0^\infty \mathbb{P}\left(F_k - F_{k-1} \leq -\frac{1}{r}\log z\middle|S_{k-1}\right)\mathrm{d}z \\
&\leq \int_0^\infty \mathbb{P}\left(F_k - F_{k-1} - \mathbb{E}[F_k - F_{k-1}] \leq -a\tau_k - \frac{1}{r}\log z\middle|S_{k-1}\right)\mathrm{d}z \\
&\leq \int_{\exp(-ra\tau_k)}^\infty \mathbb{P}\left(F_k - F_{k-1} - \mathbb{E}[F_k - F_{k-1}] \leq -a\tau_k - \frac{1}{r}\log z\middle|S_{k-1}\right)\mathrm{d}z + \mathrm{e}^{-ra\tau_k} \\
&\leq \int_{\exp(-r\,a\,\tau_k)}^\infty \exp\Big(-\frac{2}{\tau_k\,\phi_f^2}(a\,\tau_k + r^{-1}\log z)^2\Big)\mathrm{d}z + \exp(-r\,a\,\tau_k)\,.
\end{aligned}$$

In the first inequality above, we apply the bound that $\mathbb{E}[F_k - F_{k-1}] \geq a\,\tau_k$ from Lemma 23. In the second inequality, for values of z such that $-a\,\tau_k - r^{-1}\log z \geq 0$ we bound the integrand from above by 1, which results in the $\exp(-r\,a\,\tau_k)$ term appearing. In the final inequality we apply (4.28).

We now show that the right hand side of the expression above is strictly less than 1 for a suitable choice of r, and τ_k suitably large:

$$\begin{aligned}
&\int_{\exp(-r\,a\tau_k)}^\infty \mathrm{e}^{-\frac{2}{\tau_k\,\phi_f^2}(a\,\tau_k + r^{-1}\log z)^2}\mathrm{d}z = \frac{1}{r}\int_0^\infty \mathrm{e}^{-\frac{2\,y^2}{\tau_k\,\phi_f^2}}\cdot\mathrm{e}^{r\,y}\cdot\mathrm{e}^{-r\,a\,\tau_k}\mathrm{d}y \\
&= \frac{1}{r}\int_0^\infty \exp\left(-\frac{2}{\tau_k\,\phi_f^2}(y - r\,\tau_k\phi_f^2/4)^2\right)\cdot\exp\left(r^2\,\tau_k/4\right)\cdot\exp\left(-r\,a\,\tau_k\right)\mathrm{d}y \\
&\leq r^{-1}\sqrt{\tau_k\,\phi_f^2\,\pi/2}\,\cdot\exp(r^2\tau_k/4 - r\,a\,\tau_k)\,.
\end{aligned}$$

The final inequality follows by integrating over $\mathbb{R}$ rather than $\mathbb{R}_0$ and by noting that the integral of $\exp(-y^2)$ over $\mathbb{R}$ is equal to $\sqrt{\pi}$.

Observe that there exists $r > 0$ such that $r^2/4 - r\,a < 0$. Thus for this choice of r, for all τ_k suitably large, we have that, as desired,

$$\int_{\exp(-r\,a\,\tau_k)}^{\infty} \exp\Big(-\frac{1}{\tau_k}(a\,\tau_k + r^{-1}\log z)^2\Big)\mathrm{d}z + \exp(-r\,a\,\tau_k) < 1\,.$$

□

We can now prove Theorem 7 using well known martingale arguments.

Proof of Theorem 7. We first apply standard stopping arguments to (4.27) to show that if Y_0 is such that $|Y_0| > \kappa$, then there is positive probability that F_k will not go negative, namely,

$$\mathbb{P}\Big(\inf_{k\geq 0} F_k \geq 0\Big) \geq 1 - \exp(-rK) > 0\,, \tag{4.29}$$

for some $K > 0$. We do so by investigating the probability of its complement.

Let T be the first time when $F_k < 0$ occurs for $k \geq 0$, which is a stopping time. Using Lemma 24, recalling that $r > 0$,

$$\begin{aligned}
\mathbb{P}\Big(\inf_{k\geq 0} F_k < 0\Big) &= \mathbb{P}(F_T < 0) \\
&= \mathbb{P}\left(\mathrm{e}^{-rF_T} > 1\right) \\
&\leq \mathbb{E}\,\exp(-rF_T) \\
&= \mathbb{E}\left[\liminf_{n\to\infty} \exp(-rF_{T\wedge n})\right] \\
&\leq \liminf_{n\to\infty} \mathbb{E}\left[\exp(-rF_{T\wedge n})\right] \\
&\leq \liminf_{n\to\infty} \mathbb{E}\,\exp(-rF_0) = \mathbb{E}\,\exp(-rF_0) \leq \exp(-rK)
\end{aligned}$$

where $K := \min_{y:|y|>\kappa}\{f(y)\}$ is a positive constant since f is positive and $f(x) \to \infty$ as $|x| \to \infty$. The first two equalities above apply our stopping time definition and an exponential change of variable. The first inequality above applies Markov's inequality, the second applies Fatou's lemma and the third is the optional stopping theorem (see e.g. [93, Section 10.10]) applied to our supermartingale.

The next step is to show using the Strong Markov Property, that at every time ℓ when $|Y_\ell| > \kappa$ holds, there is a positive probability that the process F_k remains positive for all remaining time. Due to irreducibility $|Y_k| > \kappa$ occurs infinitely often, and so eventually it will be that $F_k > 0$ for all time. We now argue this point more formally. Let ℓ_0 be the first time that $|Y_k| > \kappa$ holds. For $n \geq 1$, let

$$F_k^{(n)} = f(Y_k) - a\sum_{i=\ell_{n-1}}^{k-1} \tau(Y_i)\,,$$

which is the process F started from time ℓ_{n-1}. Let σ_n be the first time after ℓ_{n-1} when $F_k^{(n)} < 0$ holds, and let ℓ_n be the first time after σ_n that $|Y_k| > \kappa$ holds. Since our Markov

chain is irreducible it must be that if σ_n is finite, then ℓ_{n+1} is finite. By this and (4.29) we have

$$\mathbb{P}(\sigma_n < \infty \,|\, \sigma_{n-1} < \infty) = \mathbb{P}(\sigma_n < \infty \,|\, \ell_n < \infty) < \mathrm{e}^{-rK}.$$

Thus, upon noting that σ_n cannot possibly be finite if σ_{n-1} is not, we have

$$\mathbb{P}(\sigma_n < \infty) \leq \exp(-rK)\,\mathbb{P}(\sigma_{n-1} < \infty) < \ldots < \exp(-nrK)\,.$$

Now, note that

$$\sum_{n=0}^{\infty} \mathbb{P}(\sigma_n < \infty) = \sum_{n=0}^{\infty} \exp(-nrK) < \infty\,,$$

so by Borel–Cantelli (see e.g. [93, Section 2.7])

$$\mathbb{P}(F_k^{(n)} < 0, \text{ infinitely often }) = 0\,.$$

Thus, there exists a k' such that for all $k \geq k'$, we have that

$$f(Y_k) - a\sum_{i=k'}^{k-1} \tau(Y_i) \geq 0$$

which, after rearranging, implies

$$\liminf_{k\to\infty} \frac{f(Y_k)}{\sum_{i=0}^{k-1}\tau(Y_i)} \geq \liminf_{k\to\infty} \frac{f(Y_k)}{\sum_{i=k'}^{k-1}\tau(Y_i)} \cdot \liminf_{k\to\infty} \frac{\sum_{i=k'}^{k-1}\tau(Y_i)}{\sum_{i=0}^{k-1}\tau(Y_i)} \geq a\,,$$

as required.

□

4.4.3 Proof of Theorem 7 for the local search algorithm

We now prove Theorem 7 under the premise that the local search algorithm, Algorithm 17, is applied. First, we consider the situation where the local search algorithm must compare an unstable parameter λ with a stable parameter γ. The following lemma will be used to show that the probability of the Metropolis rule, (4.8), selecting γ will be a low probability event.

Lemma 25. *For the events*

$$A = \left\{ f\big(X_{\tau(x)}^{(\lambda)}\big) - f\big(X_0^{(\lambda)}\big) \leq \frac{3\delta}{4} f(X_0^{(\lambda)}) \right\}$$

and

$$B = \left\{ f\big(X_{\tau(x)}^{(\gamma)}\big) - f\big(X_0^{(\gamma)}\big) \geq \frac{\delta}{2} f(X_0^{(\gamma)}) \right\},$$

with $\lambda \in \overline{\mathcal{L}}$ and $\gamma \notin \overline{\mathcal{L}}$ there exists positive constants β_1 and β_2 such that

$$\mathbb{P}(A \,|\, X_0^{(\lambda)} = x) \leq \beta_1 \mathrm{e}^{-\beta_2 \tau(x)}\,, \tag{4.30}$$

$$\mathbb{P}(B \,|\, X_0^{(\gamma)} = x) \leq \beta_1 \mathrm{e}^{-\beta_2 \tau(x)}\,. \tag{4.31}$$

Proof. The bound (4.30) is a consequence of the Azuma-Hoeffding Inequality. In particular,

$$\left\{f\big(X^{(\lambda)}_{\tau(x)}\big) - f\big(X^{(\lambda)}_0\big) \leq \frac{3\delta}{4} f(X^{(\lambda)}_0)\right\} \subset \left\{f\big(X^{(\lambda)}_{\tau^*}\big) - f\big(X^{(\lambda)}_0\big) \leq \frac{3\delta}{4} f(X^{(\lambda)}_0)\right\}$$

where $\tau^* = \min\{t \leq \tau(x) : |X^{(\lambda)}_t| \leq \kappa\}$ for suitably large values of $|x|$. Since λ is unstable, $f(X^{(\lambda)}_{t\wedge\tau^*})$ is a sub-martingale with bounded increments and drift δ. Thus we can directly apply the Azuma-Hoeffding Inequality to obtain (4.30).

The bound (4.31) is a direct consequence of Lemma 21. In particular, taking $w = f(x)$ and $z = \frac{\delta}{2} f(x)$, the terms in the exponential in statement (4.14) of Lemma 21 are such that

$$\frac{(z - \alpha_1(w))^2}{2\alpha_2(w)} \sim \left[\frac{(\frac{\delta}{2c} + 1)^2}{2(\phi + \delta)^2 \sigma}\right] \tau(x)\,,$$
$$\frac{(z - \alpha_3(w))^2}{2\alpha_4(w)} \sim \left[\frac{(1 + \frac{\delta}{2})^2}{2c^2 \phi \sigma}\right] \tau(x)\,.$$

Further, $n(f(x)) = \mathrm{O}(\tau(x))$. This in turn implies that there are constants β_1 and β_2 such that (4.31) holds. □

We let $(Y, \Lambda) = (x, \lambda)$ be the initial state of Algorithm 17, we let $\gamma \notin \overline{\mathcal{L}}$ be the parameter selected in Step (i) of Algorithm 17, and we let (Y', Λ') the state of Algorithm 17 after its first iteration. Given this notation, the following lemma, which is a consequence of the above result, shows that with high probability $\Lambda' = \lambda$ and that over this step $\tau(x)$ is increased by a positive fraction.

Lemma 26. *There exists positive constants ε, β_3, and β_4 such that*

$$\mathbb{P}(\tau(Y') \geq \tau(x)(1 + \varepsilon), \Lambda' = \lambda) \geq 1 - \beta_3 \mathrm{e}^{-\beta_4 \tau(x)}\,.$$

Proof. Let A and B be the events specified in Lemma 25, above. Given the event A^c, for $\Lambda' = \lambda$ we have that $f(Y') \geq (1 + 3\delta/4) f(x)$. Since $f(x) = \Theta(\tau(x))$, for an appropriate choice of $\varepsilon > 0$ (dependent only on δ), we have that

$$\tau(Y') \geq (1 + \varepsilon)\, \tau(x).$$

Now given this choice of ε the following equalities hold,

$$\begin{aligned}
&\mathbb{P}(\tau(Y') \geq \tau(x)(1 + \varepsilon),\ \Lambda' = \lambda) \\
\geq &\mathbb{P}(\tau(Y') \geq \tau(x)(1 + \varepsilon),\ \Lambda' = \lambda \mid A^c, B^c)\, \mathbb{P}(A^c \cap B^c) \\
\geq &\Big(1 - \mathrm{e}^{-\frac{1}{4}\delta f(x)}\Big)\Big(1 - 2\beta_1 \mathrm{e}^{-\beta_2 \tau(x)}\Big)
\end{aligned}$$

The second inequality follows from definition of the Metropolis rule, (4.8), and from Lemma 25. From this it is clear there are appropriate constants β_3 and β_4, as required. □

Proof of Theorem 7 for local-search algorithm. We see that under Assumption 18, the local search algorithm is such that the process Λ_k will eventually visit a state in $\overline{\mathcal{L}}$. To see this note that, from any state $(Y_k, \Lambda_k) = (x, \lambda)$ with $\lambda \notin \overline{\Lambda}$, by irreducibility and positive

recurrence of $X^{(\lambda)}$ and the fact $\lambda \in \mathcal{B}_\lambda$, there is a positive probability of reaching state (x_0, λ). Further, by (4.9) there is a positive probability of reaching a state (x_0, μ) for any $\mu \in \overline{\mathcal{L}}$. From that state, again by the irreducibility of $X^{(\mu)}$, there is a positive probability of reaching a state x' with $\tau(x') > \tau$ for any specified value of τ. Once such a state is reached we now show that there is a positive probability of Λ_k remaining in $\overline{\Lambda}$ indefinitely.

Let E_k be the event

$$E_k := \{\Lambda_k \in \overline{\mathcal{L}},\ \tau_k \geq \tau_{k-1}\,(1+\varepsilon)\}\,.$$

Then, by Lemma 26,

$$\mathbb{P}\Big(\bigcap_k E_k\Big) \geq 1 - \sum_k \mathbb{P}\left(E_k^{\mathrm{c}} \,\Big|\, \bigcap_{k'<k} E_{k'}\right) \geq 1 - \sum_k 2\beta_1 \mathrm{e}^{-\beta_2 \tau_0\,(1+\varepsilon)^k}\,. \tag{4.32}$$

Thus for suitably large initial values of τ_0 we have that

$$\mathbb{P}(\Lambda_k \in \overline{\mathcal{L}},\ \tau_k \geq \tau_{k-1}\,(1+\varepsilon)\ \forall\ k) > 0\,.$$

Hence, eventually it must occur that the algorithm evolves only according to unstable parameter choices. □

4.5 Examples

This section presents five example applications of the algorithm, where each example is designed to highlight aspects of the algorithm's implementation and use. More specifically, we subsequently consider a network of parallel queues, a tandem queueing system, the Rybko–Stolyar network, a network of input queued switches, and a random access network (RAN).

Before proceeding with the examples we briefly provide some details on algorithm parameter choices used in this section. The η parameter scales the effect of the difference between outcomes sampled from consecutive draws from Λ_n samples. Taking $\eta \to \infty$ is akin to adopting a greedy hill climbing approach, while $\eta \to 0$ is equivalent to moving at random. Hence the choice of η allows for a trade off between moving towards more unstable parameter choices and becoming trapped in local (stable) optimizers. Throughout this section we have chosen $\eta = 1$ as a balance between these two extremes. For the local search algorithm, associated with each $\lambda \in \mathcal{L}$ is a neighbourhood $\mathcal{B}_\lambda$ from which the next parameter candidate will be selected. Choosing these neighbourhoods to be large will explore $\mathcal{L}$ more aggressively, while smaller neighbourhoods will increase the effect of local gradient information. Throughout our illustrations we take $\mathcal{B}_\lambda$ to be a ball centred at λ with radius $\varepsilon = 0.01$ that intersects with $\mathcal{L}$. Finally, throughout the section we use $\mathcal{U}(A)$ as an indicator variable for the algorithm declaring the set A unstable.

4.5.1 Parallel queues with randomly varying connectivity

For our first example we extend the illustrative example used in Section 4.3. Consider a system where N parallel queues compete for the service of a single server. Time is slotted, and in each time slot $t \in \mathbb{N}_0$ queue $i \in \{1, \ldots, N\}$ is connected to the server with probability 0.8. Similarly, at the beginning of each time slot an arrival occurs at each queue with probability $p \in [0,\, 1]$, so that there are at most N arrivals to the system in any

particular time slot. After the arrivals have occurred and connectivity is determined, the longest non-zero queue that is connected to the server is reduced by one with probability 0.8 — a policy called *longest queue first* (LQF). The system is therefore a discrete time Markov chain $X^{(p)}$ taking values in $\mathbb{N}_0^N$. We illustrate this system in Figure 4.3.

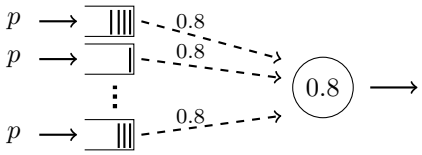


Figure 4.3: A parallel queueing system with randomly varying connectivity.

The stability region for this irreducible Markov chain is known. In particular, from Corollary 1 in [88] we have that for any $p \leq \ell^*$, where

$$\ell^* = \frac{4}{5}\left(\frac{1-(1/5)^N}{N}\right),$$

the limiting distribution of $X^{(p)}$ exists, and otherwise does not.

Therefore any $\mathcal{L} \subset [0,\infty)$ that shares an intersection with $[\ell^*,\infty)$ of positive measure, is unstable under our Definition 15. Taking $N = 4$ and $\mathcal{L}$ to be of the form $[0,\ell)$, we therefore have instability for approximately those instances when $\ell > 0.2$, that is $\ell^* \approx 0.1997$. Furthermore, Theorem 1 in [88] shows that the system is stable under the LQF policy for the network's subcritical region — a property known as *maximal stability*. This property is well known to hold for single-hop networks under LQF and its generalization the Max Weight-α algorithm (see e.g. [94, 13]).

In Figure 4.4 we give the proportion of simulation runs out of 1000 where the parameter set [0, 0.3] is declared unstable by the local and global algorithms as k^* is increased. Recall that k^* is the total number of steps the algorithm is permitted to take in $\mathcal{X}$ before a value of $f(Y_k)$ is compared to $q_k^{(\alpha)}$. Now, the greatest change in f occurs when there are no services and all queues experience an arrival, so that $\phi = 4$. We assume $\kappa = 4$ and $\sigma = 1$. It can be seen that longer simulation runs are more likely to declare the system unstable, with an apparent almost sure declaration of instability in the limit. In this case, the local algorithm approaches this limit far more rapidly than the global algorithm.

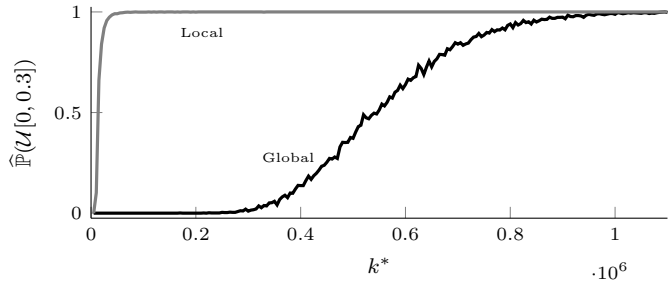


Figure 4.4: Parallel system L_1-stability tests with $\alpha = 0.05$ for p sampled from the set $\mathcal{L} = [0, 0.3]$ for $k^* \in (0, 10^6]$ with $\tau(x) = 0.5|x| + 1$, $\delta = 0.01$, $\sigma = 1$, $\kappa = 4$, $\phi = 4$ and $\varepsilon = 0.01$.

Figure 4.5 explores the effect of the chosen δ on an unstable declaration for an unstable parameter set. The figure gives the proportion of simulation runs out of 100 where the parameter set [0, 0.21] is declared unstable by the local and global algorithms. Recall that the definition of stability we use compares the drift of the process under consideration with a linear function that depends on δ. As discussed in Section 4.3, with reference to Figure 4.2, if a parameter is unstable for a particular δ, then this implies instability for all higher values of δ. This is because a W process parameterized by a particular δ will stochastically dominate all W processes parametrised by higher choices of δ. Figure 4.5 demonstrates that this occurs for both the global and local search algorithms. Again we see that the local algorithm appears to perform better — in this example it has detected lower values of downward drift when $k^* = 10^5, 10^6$.

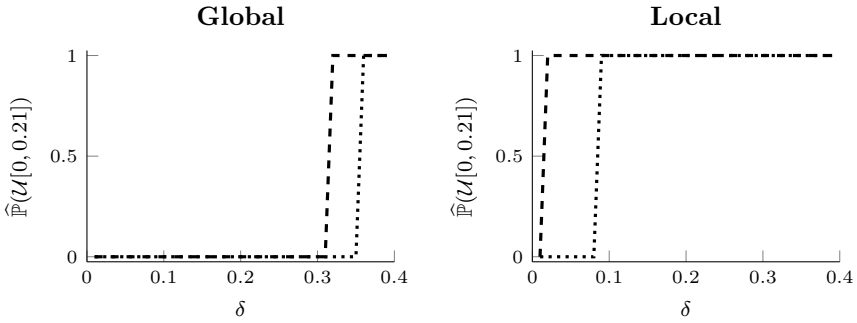


Figure 4.5: Parallel system L_1-stability tests for p sampled from the set $\mathcal{L} = [0, 0.21]$ for $\delta \in [0.01, 0.4]$ with $\tau(x) = 0.5|x| + 1$, $\sigma = 1$, $\kappa = 4$, $\varepsilon = 0.01$ and $k^* = 10^5$ (dotted), 10^6 (dashed).

Figure 4.6 explores the effect of the chosen δ on an unstable declaration for a stable parameter set. The figure gives the proportion of simulation runs out of 100 where the parameter set [0, 0.19] is declared unstable by the global algorithm. The algorithm rejects the null hypothesis of stability once δ reaches approximately 0.28. This indicates that $f(Y_{10^5})$ exceeds the (estimated) 95-th percentile of W_{10^5} parametrised with $\delta = 0.3$, but is bounded by the 95-th percentile of a W_{10^5} parametrised by $\delta = 0.25$. For $\mathcal{L} = [0, 0.19]$ rejecting the null hypothesis of downward drift greater than 0.28 is not unexpected. Over the range of δ considered the local algorithm did not make a declaration of instability; hence the local algorithm nonetheless appears to perform better.

In Figure 4.7 we give the proportion of simulation runs out of 100 where the parameter set $[0, \ell]$ is declared unstable for a range of ℓ. It can be seen that longer simulation runs declare the system unstable for a larger proportion of the ℓ values that give an unstable $\mathcal{L}$. The figure provides evidence that in the discrete time case the algorithm is performing as it is intended to, in the next section we move to a continuous time example.

4.5.2 Tandem queues

Our next example is the tandem queueing system. We will contrast the results for a Markov system consisting of two single server queues in tandem with a system that has renewal arrivals and iid service times at both nodes (which is not Markov). In the former system jobs arrive to a server according to a Poisson process with rate one, they are then processed one at a time, first come first served (FCFS), with $\mathsf{Exp}(\mu_1)$ service times,

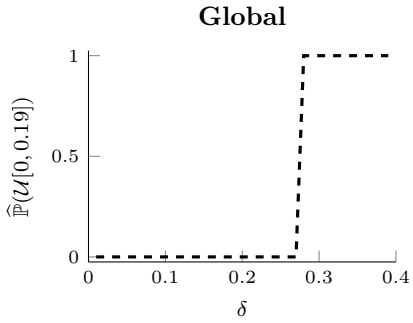


Figure 4.6: Parallel system L_1-stability tests with $\alpha = 0.05$ for p sampled from the set $\mathcal{L} = [0, 0.19]$ for $\delta \in [0.01, 0.4]$ with $\tau(x) = 0.5|x| + 1$, $\sigma = 1$, $\kappa = 4$, $\varepsilon = 0.01$ and $k^* = 10^5$.

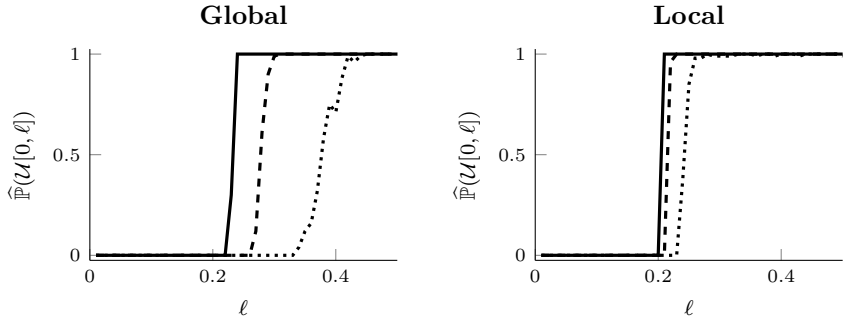


Figure 4.7: Parallel system L_1-stability tests with $\alpha = 0.05$ for p sampled from sets of the form $\mathcal{L} = [0, \ell]$ with $\tau(x) = 0.5|x| + 1$, $\delta = 0.05$, $\sigma = 1$, $\kappa = 4$, $\varepsilon = 0.01$ and $k^* = 10^5$ (dotted), 10^6 (dashed), 10^7 (solid).

before being sent to a subsequent server where they are again processed one at a time, FCFS, with service time $\mathsf{Exp}(\mu_2)$. It is well known that the output from the first server to the second corresponds to a Poisson process with rate $\min\{1, \mu_1\}$. Consequently, the system is L_1-stable for $(\mu_1^{-1}, \mu_2^{-1}) \in [0, 1]^2$, and L_1-unstable otherwise. In the latter system we assume the times between arrivals to the first server are Erlang distributed with rate parameter $1/2$ and shape parameter 2. Jobs are also served FCFS and must pass through the first server before being sent to the second. In this case the service times are Weibull distributed with shape parameter 2, so that they have distribution function $(1 - \exp(-(x/\mu)^k)$ for $x \geq 0$, with $k = 2$ and scale parameters $\mu = \mu_1^{-1}$ and $\mu = \mu_2^{-1}$ for the first and second server, respectively. Note that in both cases the mean time between arrivals is 1, that the mean service times are μ_1^{-1} and μ_2^{-1} for the former case, and are $\Gamma(1.5)\mu_1^{-1} \approx 0.8862\,\mu_1^{-1}$ and $\Gamma(1.5)\mu_2^{-1} \approx 0.8862\,\mu_2^{-1}$ in the latter case.

To apply our discrete time framework to these continuous time systems, we have used the embedded process corresponding to the sequence of states recorded immediately after each jump (which is Markovian for the $M/M/1$ system, and non Markovian for the system with renewal arrivals and i.i.d. service times). In Figure 4.8 and Figure 4.9 we are testing parameter sets of the form $(\mu_1^{-1}, \mu_2^{-1}) \in \mathcal{L} = [0, \ell]^2$, and as such sets with $\ell > 1$ are L_1-unstable in the Markov case and with approximately $\ell > 1.1284 = (0.8862)^{-1}$ in the

non Markov case. In both the global and local cases it is clear that the test converges to an accurate declaration of instability over $\ell \in (0.5, 1.5)$ as $k^* \to \infty$. The figures provide evidence that it is possible to relax the discrete time and Markov assumptions we made in the theoretical development of our algorithm.

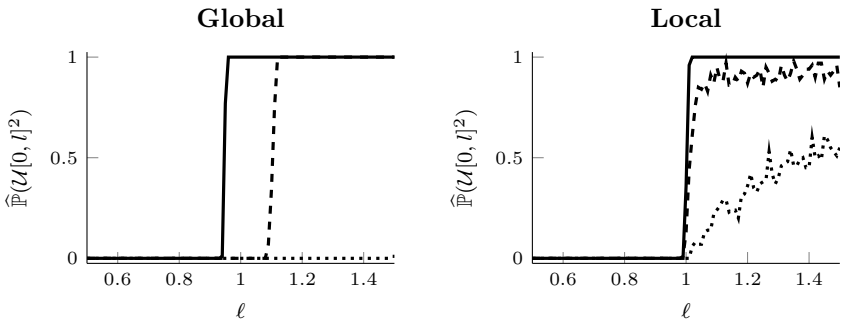


Figure 4.8: Tandem Markovian system L_1-stability tests with $\alpha = 0.05$ for (μ_1^{-1}, μ_2^{-1}) sampled from sets of the form $\mathcal{L} = [0, \ell]^2$ with $\tau(x) = 0.5|x| + 1$, $\delta = 0.05$, $\sigma = 1$, $\kappa = 1$, $\varepsilon = 0.01$ and $k^* = 10^5$ (dotted), 10^6 (dashed), 10^7 (solid).

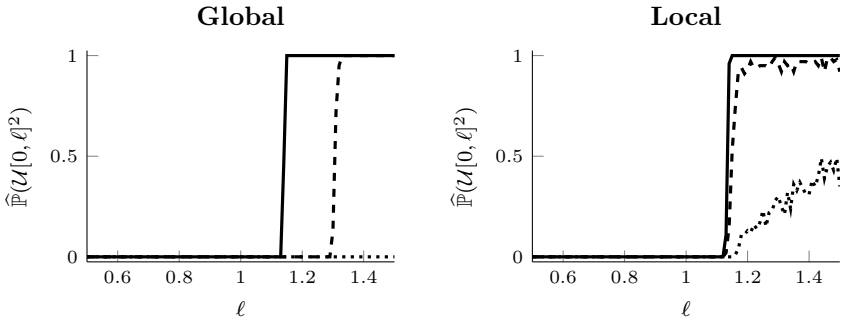


Figure 4.9: Non-Markovian tandem system L_1-stability tests with $\alpha = 0.05$ for (μ_1^{-1}, μ_2^{-1}) sampled from sets of the form $\mathcal{L} = [0, \ell]^2$ with $\tau(x) = 0.5|x| + 1$, $\delta = 0.05$, $\sigma = 1$, $\kappa = 1$, $\varepsilon = 0.01$ and $k^* = 10^5$ (dotted), 10^6 (dashed), 10^7 (solid).

Further, we are stretching the original modelling framework since there is no fixed σ after which the systems exhibit unstable behaviour. The required number of steps before an upward drift is expected to occur depends on the system state. Consequently, over short time periods, unstable parameter choices may appear stable, e.g., in the Markov system, when the second server has a very large queue but a parameter selection with $\mu_1^{-1} > 1$ and $\mu_2^{-1} < 2 - \mu_1^{-1}$ is made. Nonetheless, asymptotically both systems are expected to become infinitely large due to the first queue being unstable, and through the τ function our algorithm is able to maintain accurate prediction. Due to this, in systems of this kind the choice of c in the τ function may have an important impact on the algorithm's performance.

In Figure 4.10 we perform instability tests on $[0, 1.2]$ for a range of c. For the global algorithm the choice of c can have a substantial impact on performance, for $k^* = 10^6$ a high value of c is required to obtain a high level of accuracy. For the local algorithm,

however, the choice of c does not appear to have as much of an effect as the choice of k^*. This suggests that if k^* is limited by computational resources, then it is preferable to use the global algorithm with a high c — particularly if the system is suspected of exhibiting oscillatory behaviour.

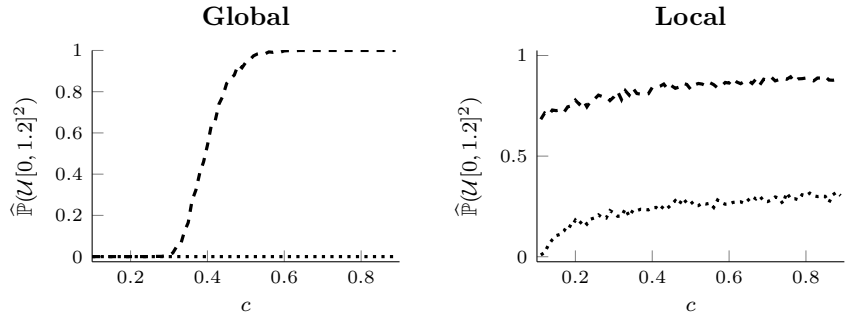


Figure 4.10: Tandem Markovian system L_1-stability tests with $\alpha = 0.05$ for (μ_1^{-1}, μ_2^{-1}) sampled from $\mathcal{L} = [0, 1.2]^2$ with $\tau(x) = c|x| + 1$, $\delta = 0.05$, $\sigma = 1$, $\kappa = 1$, $\varepsilon = 0.01$, $k^* = 10^5$ (dotted), 10^6 (dashed), and $c \in (0.1, 0.6)$.

4.5.3 Rybko–Stolyar queueing network

The Rybko–Stolyar queueing network, displayed in Figure 4.11, was introduced in [17] as an example of a work-conserving queueing network that can be unstable for sub-critical parameter choices. To the best of our knowledge, matching necessary and sufficient conditions for instability are not known.

This queueing network consists of two stations, each with a single server, which we call the left and right stations. All customers served at the left station require $\mathsf{Exp}(\mu_l)$ service time and all customers served at the right station require $\mathsf{Exp}(\mu_r)$ service time. There are two classes of customers. The first class enters the network according to a Poisson process at rate λ where it is served at the left station before proceeding to the right station to be served, and from here it departs the network. Jobs from the second class also enter the network at rate λ, are served at the right station, proceed to be served at the left station, and then depart from the network. Within each customer class the customers are served on a FCFS basis. Between the customer classes, however, there is priority: jobs being served at their second station (bold in Figure 4.11) have priority over jobs being served at their first station.

In [17] it is shown that for λ equal to one and $\mu_r > 0$, a sufficient condition for instability is $\mu_l < 2$. In Figure 4.12 we consider the situation where μ_l is sampled from sets of the form $(\ell, \ell + 1)$ for $\ell \in (1, 3)$, with $\lambda = 1$ and $\mu_r = 4$. Due to the result from [17] we expect that $\ell \in (1, 2)$ will be returned as unstable by the algorithm. This occurs for k^* equal to 10^7. Interestingly, for $\ell > 2$ we never reject the null hypothesis of stability, suggesting that $\mu_l < 2$ is also a necessary condition for instability with $\lambda = 1$ and $\mu_r > 0$. In this case the local algorithm appears to outperform the global algorithm. The estimates for the local algorithm do, however, exhibit a large amount of variance (over the 100 sample paths used to generate the figure).

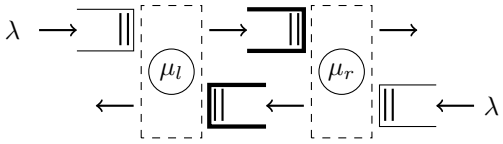


Figure 4.11: The Rybko–Stolyar network.

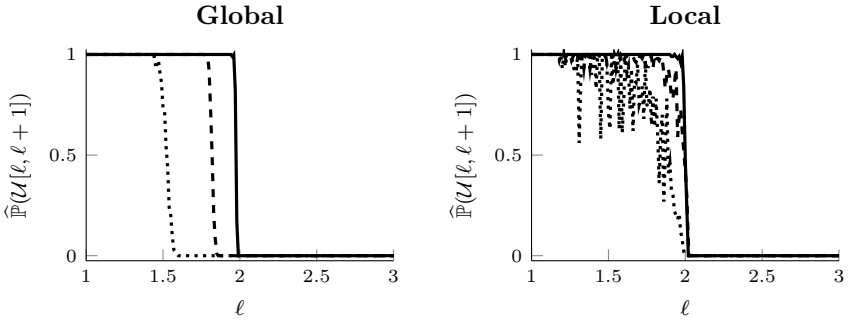


Figure 4.12: Rybko–Stolyar system L_1-stability tests with $\alpha = 0.05$ for μ_l sampled from sets of the form $\mathcal{L} = [\ell,\, \ell+1]$ for $k^* = 10^5$ (dotted), 10^6 (dashed), 10^7 (solid) with $\lambda = 1$, $\mu_r = 4$, $\tau(x) = 0.5|x| + 1$, $\delta = 0.05$, $\phi = \kappa = \sigma = 1$ and $\varepsilon = 0.01$.

4.5.4 A switch network

Our next example is a network of input-queued switches which was investigated by Andrews and Zhang in [89]. This discrete time model provides an example where the LQF policy is not maximally stable. In this simulation study, we are able to demonstrate the use of our algorithm on a model which exhibits complex queueing dynamics on a 52 dimensional state space. Again, unlike the parallel queue or tandem models considered earlier, the explicit form of the stability region of this model is unknown.

The model we are considering is illustrated in Figure 4.13. It has four main switches with labels A, B, C, and D and four auxiliary switches with labels A', B', C', and D'. Each of the main switches has ten external input queues to which a packet arrival occurs instantaneously at the beginning of each time slot independently and with probability $r/30$.

Packets are given a type according to the switch at which they first arrive, for example packets starting at A are of type 1; packets are routed through the network according to their type. After these arrivals the longest of the 12 queues at each main switch and of the three queues at each auxiliary switch sends a single packet to the corresponding input queue of another switch or are removed from the system (as designated by Figure 4.13). Packets sent in a time slot arrive at their destination at the beginning of the next time slot.

In Figure 4.14 we test for L_1-instability in r on parameter sets of the form $\mathcal{L}_s = [0.5, \ell]$. Due to the large size of the system we have chosen $\delta = 5$. We set $\phi = 40$, $\tau(x) = 0.5\,|x|+1$, and $\kappa = \sigma = 1$. Although the stability region for this model is not yet known, this figure provides strong (statistical) evidence that the set [0, 0.95] is unstable. We have thus demonstrated that our algorithm can be used to provide statistical evidence that the LQF policy is not necessarily maximally stable in multi-hop settings. In this case the

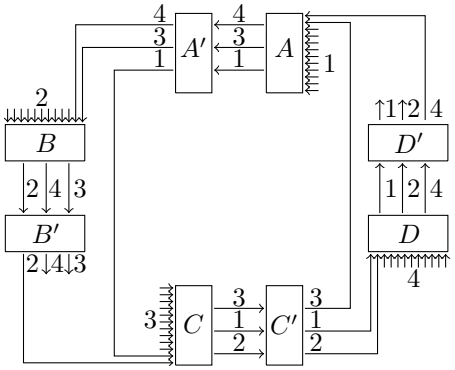


Figure 4.13: A network of input queued switches.

global algorithm appears to perform much better, suggesting that $k^* = 10^7$ is not great enough for the local algorithm to start performing well.

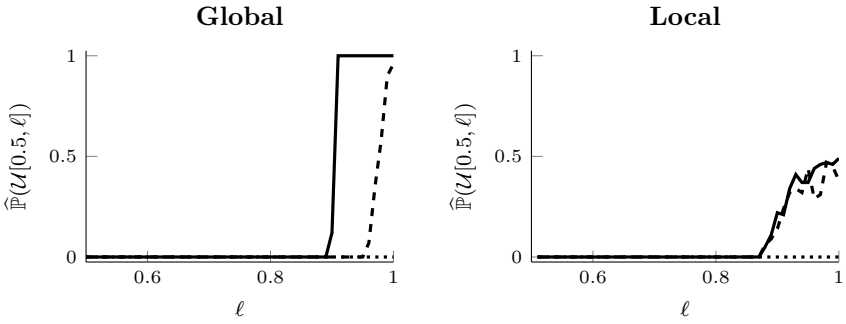


Figure 4.14: Network of input queued switches L_1-stability tests with $\alpha = 0.05$ for r sampled from sets of the form $\mathcal{L} = [0, \ell]$ or $\mathcal{L} = [0.5, \ell]$ for $k^* = 10^5$ (dotted), 10^6 (dashed), 10^7 (solid), $\tau(x) = 0.5|x| + 1$, $\delta = 5$, $\phi = 40$, $\kappa = \sigma = 1$, and $\varepsilon = 0.01$.

It may be the case the ratio $|\overline{\mathcal{L}}|/\mathcal{L}$ has a substantial impact on performance in finite time. Figure 4.15 explores this relationship by testing for stability of $[\ell, 0.95]$ over a variety of (k^*, ℓ) combinations. Intuitively, this ratio should have a greater impact on performance of the local algorithm than the global algorithm. Instead, the figure indicates highly similar (poor) performance over the varying combinations of (k^*, ℓ), with some degradation of accuracy for very low k^*. While for the global algorithm fixing either k^* or ℓ and then increasing the other leads to substantial increases in accuracy.

4.5.5 A broken diamond random access network

So far we have presented classical examples that facilitated the assessment of the algorithm's performance. In our final example we address a contemporary area of research initiated by [90], exploring the stability properties of a wireless network with a queue-based random-access algorithm. We focus on a network consisting of nodes $\{1, 2, \ldots, 6\}$, some of which are connected by edges, as depicted in Figure 4.16 (where it is remarked that [90]

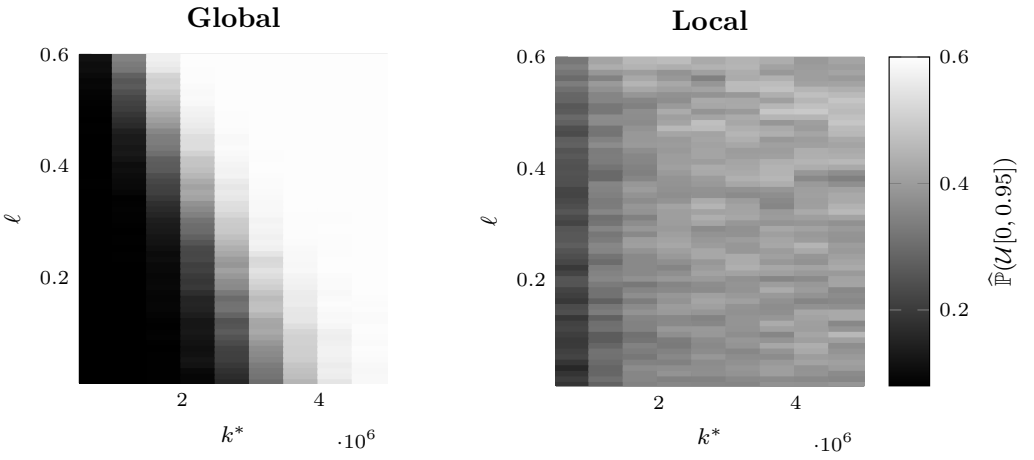


Figure 4.15: Network of input queued switches L_1-stability tests for r sampled from sets of the form $\mathcal{L} = [\ell, 0.95]$ for $k^* \in (0, 7 \cdot 10^6]$, $\tau(x) = 0.5|x| + 1$, $\delta = 5$, $\phi = 40$, $\kappa = \sigma = 1$, and $\varepsilon = 0.01$.

is set in a more general context). In our model we assume that nodes which are connected by an edge interfere with each other, that is, they cannot transmit simultaneously.

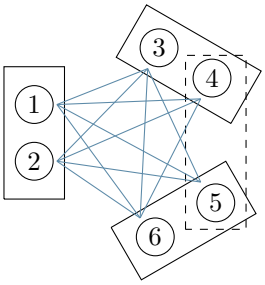


Figure 4.16: A broken diamond random access network.

In this continuous time model, packets arrive to node i according to a Poisson process with rate λ_i and take $\mathsf{Exp}(\mu_i)$ time to transmit, so that the traffic intensity at node i is $\varrho_i = \lambda_i/\mu_i$. Let $U(t) \in \{0,1\}^6$ be a vector of indicator variables representing which nodes are active at time t and $X(t) \in \{0, 1, \dots\}^6$ be a vector representing the number of packets at each node at time t.

In order to fully describe the evolution of this process, we must specify how nodes decide when to attempt transmission of packets. Whenever a node is not being interfered with it will wait an $\mathsf{Exp}(\nu_i)$ amount of *back-off period*. At this point, it will then begin transmitting with probability $\phi_i(X_i(t))$, where $\phi_i(0) = 0$, and otherwise it will begin another back-off period with the same distribution. After each successful transmission, node i will release the medium and begin a back-off period with probability $\psi_i(X_i(t^-))$, with $\psi_i(1) = 1$ for all i, and otherwise begin another transmission.

It is easy to see that (X, U) is a Markov process evolving according to the rates given in Table 4.1. Note that here $\overline{u}_i = 0$ indicates that none of the neighbours of i is transmitting.

Consider the network in Figure 4.16, and suppose that $\phi_i(x) \equiv 1$, $x > 0$, and $\psi_i(x) =$

Table 4.1: Transition rates of the random access network network in Figure 4.16.

Transition	Rate	States
$(x,\, u) \to (x + e_i,\, u)$	λ_i	All
$(x,\, u) \to (x,\, u + e_i)$	$\nu_i\, \phi_i(x_i)$	$x_i > 0,\, u_i = 0,\, \overline{u}_i = 0$
$(x,\, u) \to (x - e_i,\, u)$	$\mu_i\, (1 - \psi_i(x_i))$	$x_i \geq 1,\, u_i = 1$
$(x,\, u) \to (x - e_i,\, u - e_i)$	$\mu_i\, \psi_i(x_i)$	$x_i \geq 1,\, u_i = 1$

$\mathrm{o}(x^{-\gamma})$, with $\gamma > 1$. Let

$$(\varrho_1, \varrho_2, \varrho_3, \varrho_4, \varrho_5, \varrho_6) = \varrho\, (\kappa_1, \kappa_2, \kappa_3, \kappa_3 - \widetilde{\alpha}, \kappa_6 - \widetilde{\alpha}, \kappa_6),$$

with $(\kappa_1 \vee \kappa_2) + \kappa_3 + \kappa_6 = 1$, and $0 < \widetilde{\alpha} < (\kappa_3 \wedge \kappa_6)$. Then the main result of Ghaderi et al. in [90] implies that there exists a constant $\varrho^*(\kappa, \widetilde{\alpha}) < 1$, such that for all $\varrho \in (\varrho^*(\kappa, \widetilde{\alpha}),\, 1]$ the Markov process is transient under the given parameter conditions.

We now consider the example network from the simulation section of [90]. The relative traffic intensities are taken to satisfy $\kappa_1 = \kappa_2 = \kappa_3 = 0.4$ and $\kappa_6 = 0.2$ with $\widetilde{\alpha} = 0$. Further, $\phi_i(x) \equiv 1$, $x \geq 1$, and $\psi_i(x) = (1 + x)^{-2}$. The authors note that it is 'difficult to make any conclusive statements concerning stability/instability based on simulation results alone'. They do, however, remark that for these parameter choices and $\varrho = 0.97$, their simulated sample paths appear to demonstrate strong signs of instability.

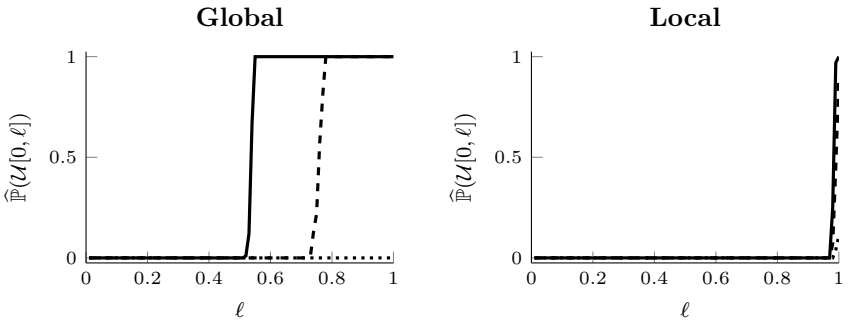


Figure 4.17: Broken diamond random access network L_1-stability tests with $\alpha = 0.05$ for sets of the form $\mathcal{L} = [0,\, \ell]$ for $k^* = 10^5$ (dotted), 10^6 (dashed), 10^7 (solid), $\tau(x) = 0.5|x| + 1$, $\delta = 0.05$, $\phi = \kappa = \sigma = 1$, and $\varepsilon = 0.01$.

In order to perform our stability test, we assume $\kappa = \phi = \sigma = 1$. In Figure 4.17 we test for L_1 instability with $\delta = 0.05$. Looking at Figure 4.17, which uses our simulation based stability test, we are able to say, with a strong statistically firm footing, that there exists a constant $\varrho^*(\kappa, \widetilde{\alpha}) < 1$, such that for all $\varrho \in (\varrho^*(\kappa, \widetilde{\alpha}),\, \ell]$ the network is unstable for a range of ℓ in approximately [0.6, 1]. This statement expands on the statement of the theorem (for a particular choice of parameters) by allowing for more information to be gained about what values are likely to be possible for $\varrho^*(\kappa, \widetilde{\alpha})$. Of course, our statement does not rule out perverse behaviour such as the network suddenly exploding after 10^7 jumps of the process. It can however be very quickly and easily applied to similar or even

vastly more complex networks. We note that the global algorithm is in this case enabling us to make this strong statement, while the local algorithm algorithm only allows us to make the statement for a substantially reduced set of ℓ.

4.6 Supporting lemmas

In this section we prove some supporting results that were used earlier in the chapter to prove our main results.

Lemma 27. *There exists a positive constant w^* such that*

$$\mathbb{P}(W_1 \geq z \,|\, W_0 = v) \leq \mathbb{P}(W_1 \geq z \,|\, W_0 = w)\,.$$

for v and w such that $z \leq w^* \leq v \leq w$.

Proof. Noting the equality

$$\mathbb{P}(W_1 \geq z \,|\, W_0 = w) = \mathbb{P}(Z(w) \geq z - w),$$

we claim it is sufficient to prove that the function

$$g(z, w) = \exp\left(-\frac{(z - w + n(w))^2}{bw}\right) \tag{4.33}$$

is nondecreasing in w for values of z with $z \geq w$. This is because the second term in (4.14) evaluated at a point $z - w$, given our assumptions on the function τ, will tend to zero as $z \to \infty$, and the first term is of the form (4.33). Also note that for g nondecreasing the function $1 \wedge g$ is also nondecreasing.

After taking logs and rearranging $g(v,\, z) \leq g(w,\, z)$, we see that it is sufficient to show that

$$\frac{z + n(v) - v}{\sqrt{n(v)}} \geq \frac{z + n(w) - w}{\sqrt{n(w)}} \tag{4.34}$$

for $z \leq w^* \leq v \leq w$. It is finally noted that (4.34) holds as long as for $w \geq w^*$ we have $n(w) \leq w$. □

Lemma 28. *The random variables $Z(w)$ are L^2 bounded and*

$$\mathbb{E}Z(w) \to 0 \qquad \textit{as} \qquad w \to 0\,.$$

Proof. In particular, we bound the mean of $Z(w)$ for large values of w as follows:

$$\begin{aligned}\mathbb{E}Z(w) &= \int_0^\infty \mathbb{P}(Z(w) \geq z)\mathrm{d}z \\ &\leq \int_0^\infty \left[\exp\left(-\frac{(z + a_1 n(w))^2}{b_1 n(w)}\right) + n(w)\,\exp\left(-\frac{(z + a_2 w)^2}{b_2 n(w)}\right)\right]\mathrm{d}z.\end{aligned}$$

We start by bounding the first of these terms:

$$\begin{aligned}
&\int_0^\infty \exp\left(-\frac{(z+a_1 n(w))^2}{b_1 n(w)}\right)\mathrm{d}z \\
&= \int_0^\infty -\frac{b_1 n(w)}{2(z+a_1 n(w))}\mathrm{d}\left(\exp\left(-\frac{(z+a_1 n(w))^2}{b_1 n(w)}\right)\right)\mathrm{d}z \\
&= \frac{b_1}{2\,a_1}\exp\left(-\frac{a_1^2 n(w)}{b_1}\right) - \int_0^\infty \frac{b_1 n(w)}{2(z+a_1 n(w))^2}\exp\left(-\frac{(z+a_1 n(w))^2}{b_1 n(w)}\right)\mathrm{d}z \\
&\leq \frac{b_1}{a_1}\exp\left(-\frac{a_1^2 n(w)}{b_1}\right).
\end{aligned}$$

Upon applying a similar sequence of steps to the second term we find that

$$\mathbb{E}Z(w) \leq \frac{b_1}{a_1}\exp\left(-\frac{a_1^2 n(w)}{b_1}\right) + \frac{b_2 n(w)}{2a_2 w}\exp\left(-\frac{a_2^2 w^2}{b_2 n(w)}\right).$$

We conclude that $\mathbb{E}Z(w) \to 0$ as $w \to \infty$, as required.

We now analyze the second moment of $Z(w)$ to establish that these random variables are L^2 bounded. Observe that

$$\begin{aligned}
\mathbb{E}\Big[Z(w)^2\Big] &= \int_0^\infty 2z\,\mathbb{P}\Big(Z(w)\geq z\Big)\mathrm{d}z \\
&= \int_0^\infty 2z\left[\exp\left(-\frac{(z+a_1 n(w))^2}{b_1 n(w)}\right) + n(w)\exp\left(-\frac{(z+a_2 w)^2}{b_2 n(w)}\right)\right]\mathrm{d}z
\end{aligned}$$

We bound the first of these terms as follows:

$$\begin{aligned}
&\int_0^\infty 2z\,\exp\Big(-\frac{(z+a_1 n(w))^2}{b_1 n(w)}\Big)\mathrm{d}z \\
&= \int_0^\infty b_1 n(w)\frac{z}{(z+a_1 n(w))}\mathrm{d}\left(-\exp\left(\frac{-(z+a_1 n(w))^2}{b_1 n(w)}\right)\right) \\
&\leq \int_0^\infty b_1 n(w)\mathrm{d}\left(-\exp\left(\frac{-(z+a_1 n(w))^2}{b_1 n(w)}\right)\right) \\
&= b_1 n(w)\exp\left(-\frac{a_1^2 n(w)}{b_1}\right).
\end{aligned}$$

We then apply similar steps to the second term, which yields

$$\mathbb{E}\Big[Z(w)^2\Big] \leq b_1 n(w)\exp\left(-\frac{a_1^2 n(w)}{b_1}\right) + b_2 n(w)^2\exp\left(-\frac{a_2^2 n(w)^2}{b_2 w}\right).$$

The right-hand terms are uniformly bounded in w, as required. □

Proof of Lemma 21. The set $\mathcal{L}$ is assumed to be stable. That is there exists $\delta > 0$, $\sigma > 0$ and $\kappa > 0$ such that

$$\mathbb{E}\left[f(X_k^{(\lambda)}) - f(X_0^{(\lambda)})\,|\,X_0^{(\lambda)} = x\right] \leq -\delta\sigma \tag{4.35}$$

for all $k > \sigma$, all x such that $|x| > \kappa$, and $\lambda \in \mathcal{L}$.

Since (4.35) only occurs after σ time units have occurred, we consider our process on steps of size σ. That is, we consider the process $(X^{(\lambda)}_{\sigma n} : n \in \mathbb{Z}_+)$. Assuming we start with $x_0 > \kappa$, let $n(x_0)$ be the smallest integer for which $\sigma n(x_0) \geq \tau(x_0)$ holds.

Since the increments of $f(X^{(\lambda)})$ are bounded by ϕ_f we have that

$$\mathbb{P}_{x_0}\Big(f(X^{(\lambda)}_{\tau(x_0)}) - f(x_0) \geq z\Big) \leq \mathbb{P}_{x_0}\Big(f(X^{(\lambda)}_{\sigma n(x_0)}) - f(x_0) \geq z - \sigma\phi_f\Big). \tag{4.36}$$

Let n_κ be the hitting time for $(X^{(\lambda)}_{n\sigma} : n \in \mathbb{N}_0)$ on the states $\{x : |x| \leq \kappa\}$.

By splitting the right-hand expression of (4.36) into terms depending on whether the event $n_\kappa \leq n(x_0)$ occurs or not, we obtain two terms

$$\begin{aligned}
&\mathbb{P}_{x_0}\Big(f(X^{(\lambda)}_{\sigma n(x_0)}) - f(x_0) \geq z - \sigma\phi_f\Big) \\
&= \mathbb{P}_{x_0}\Big(f(X^{(\lambda)}_{\sigma n(x_0)}) - f(x_0) \geq z - \sigma\phi_f, n_\kappa > n(x_0)\Big) &(4.37)\\
&+ \mathbb{P}_{x_0}\Big(f(X^{(\lambda)}_{\sigma n(x_0)}) - f(x_0) \geq z - \sigma\phi_f, n_\kappa \leq n(x_0)\Big) &(4.38)
\end{aligned}$$

We deal with these two terms, (4.37) and (4.38), separately.

First, we bound the term (4.37). We consider the process

$$M_n = f(X^{(\lambda)}_{\sigma(n\wedge n_\kappa)}) - f(x_0) - \delta\sigma(n \wedge n_\kappa),$$

which, for $X^{(\lambda)}_0 > \kappa$, is a supermartingale by the stability assumption (4.35). Due to our bounded increments assumption we can apply the Azuma–Hoeffding inequality to this process to obtain

$$\begin{aligned}
&\mathbb{P}_{x_0}\Big(f(X^{(\lambda)}_{\sigma n(x_0)}) - f(x_0) \geq z - \sigma\phi_f, n_\kappa > n(x_0)\Big) \\
&\leq \mathbb{P}_{x_0}\Big(M_{n(x_0)} \geq z - \sigma\phi_f + \delta\sigma n(x_0)\Big) \leq \exp\left(-\frac{(z - \sigma\phi_f + \delta\sigma n(x_0))^2}{2(\phi_f + \delta)^2\sigma^2 n(x_0)}\right). &(4.39)
\end{aligned}$$

This provides a bound on our first term (4.37).

We now bound the second term (4.38). For this term the process $f(X^{(\lambda)}_{\sigma n})$ has hit below level κ, so there must be an excursion from level κ to level $z + f(x_0)$. There are at most $n(x_0)$ such excursions that can occur from below $z + f(x_0)$. We can apply the Azuma–Hoeffding inequality to each excursion. A simple union bound on these excursions then gives an upper bound that is, for our purposes, sufficiently tight.

We let n_0 be a time for which $\kappa < f(X^{(\lambda)}_{\sigma n_0}) \leq \kappa + \sigma\phi_f$. We remark that this condition is satisfied immediately after the process leaves the set of states $\{x : |x| \leq \kappa\}$. Again let n_κ be the first time after n_0 for which $\{x : |x| \leq \kappa\}$ holds.

Now consider the process

$$\widehat{M}_n = f(X^{(\lambda)}_{\sigma(n\wedge n_\kappa)}) - f(X^{(\lambda)}_{\sigma n_0}), \quad n \geq 0,$$

which, again by (4.35), is a supermartingale.

The process $\widehat{M}$ follows an excursion of $f(X^{(\lambda)}_{\sigma n})$ from when it hits above κ to when it hits below again. Further, let $\widehat{M}^*_n$ be the maximum achieved by the process $\widehat{M}$ by time

n, that is

$$\widehat{M}_n^* = \max_{k \le n} \{\widehat{M}_k\} .$$

Notice that for the event in (4.38) to hold there must be an excursion of $\widehat{M}^*$ from just above κ to above $z - \sigma\phi_f + f(x_0)$. We can bound this probability using the Azuma–Hoeffding inequality as follows:

$$\mathbb{P}_{x_0}\left(\widehat{M}^*_{n(x_0)} \ge z - \sigma\phi_f + f(x_0) - \kappa\right) \le \exp\left(-\frac{(z - \sigma\phi_f + f(x_0) - \kappa)^2}{2\phi_f^2 \sigma^2 n(x_0)}\right) .$$

Further, there are at most $n(x_0)$ possible excursions of this type. Thus we arrive at the bound

$$\begin{aligned}
&\mathbb{P}_{x_0}\left(f(X^{(\lambda)}_{\sigma n(x_0)}) - f(x_0) \ge z - \kappa - \sigma\phi_f, n_\kappa > n(x_0)\right) \\
&\le n(x_0)\, \mathbb{P}_{x_0}\left(\widehat{M}^*_{n(x_0)} \ge z - \sigma\phi_f + f(x_0) - \kappa\right) \\
&\le n(x_0)\, \exp\left(-\frac{(z - \sigma\phi_f + f(x_0) - \kappa)^2}{2\phi_f^2 \sigma^2 n(x_0)}\right) . \qquad (4.40)
\end{aligned}$$

Combining the bounds (4.39) and (4.40), we find the claimed inequality. □

4.7 Concluding remarks

The main contribution of this chapter (published as [74]) concerned the development of an automated procedure that determines if, for a specified set of parameter values, a given Markov chain is unstable. A distinctive feature of our work is that our method is simulation based, and in addition broadly applicable and straightforward to implement. It provides statistical statements on the stability of the parameter set, but, notably, we have succeeded in providing explicit performance guarantees. Some of our experiments show that our technique provides us with useful insights for models for which the stability set has not been characterized so far.

This chapter can be considered as a pioneering study on this topic, and various extensions and improvements are envisaged. An important first branch of research could relate to relaxing the assumptions imposed, such as the fact that we restrict ourselves to the class of Markov processes and the bounded step size assumption. Experiments that we performed for non Markovian tandem queues indicated that the approach still provides us with the correct result, if we perform our algorithm as if the underlying system is Markovian. In order to remove the bounded step size assumption it would be necessary to use a concentration inequality that is stronger than Azuma–Hoeffding. Additionally, our experiments contrasted global and local search versions of the algorithm. We obtained mixed results on performance, and were unable to declare either version superior to the other. Determining conditions that point towards which of these versions should be used in different circumstances remains to be a challenge.

The objective of a second branch of research could be to enhance our procedure such that it can identify, in case instability is detected, which components of the multi-dimensional Markov chain are unstable. A third branch is of an empirical nature, and relates to models of which the stability region is not yet known. By performing systematic

simulation studies one could possibly state conjectures.

Part II

Loss models and capacity management

CHAPTER 5

Introduction to loss models and capacity management

This part of the thesis is about systems where congestion or resource shortages result in requests being rejected without service. Classically, models of this type have been used for management of circuit switched telephony networks [95], and now they are finding applications in areas such as cloud computing (see e.g., [96]) and logistics (see e.g., [97, 98]). Interestingly, the analytical methods that are applicable to these systems are often vastly different to those which are appropriately modelled on an infinite state space — in particular, stability is no longer an issue. In this chapter we explore this and other key issues that arise in this kind of model.

Much of the literature on loss systems focuses on equilibrium descriptions, performance measures and controls and so a discussion along these lines is imperative for context. We then discuss results and methods from the literature which apply to the transient setting. It is this finite time horizon machinery that Chapter 6 and Chapter 7 directly build upon. In both chapters we focus on making short term capacity decisions for loss systems. In the former chapter an existing method is extended to a more sophisticated model that allows potential system failures to be incorporated into the capacity management decision. In the latter chapter a new method is developed which is applicable to even more sophisticated models which are of relevance to cloud computing platform capacity management decisions.

The results of Chapter 6 utilise Laplace–Stieltjes transform and related Tauberian theorem results along with methods for orthogonal polynomials to analyse a model of a faulty loss system. These intricate methods are generally difficult to expand to new models, so in Chapter 7 we rely on matrix analytic methods to study a more sophisticated model in the context of cloud computing. Finally, in Chapter 8 we consider a model relevant to cloud edge computing that is out of reach of both of these approaches and develop a novel elegant simulation-based method to optimise the model and consequently guide management of the system.

5.1 Loss systems

While the first part of this thesis focused on a broad range of models exhibiting queueing behaviour, in this part we are more specific — we focus on a particular well known

loss model and develop some generalisations. In the simplest case, suppose jobs arrive according to a Poisson process with rate λ and, if upon arrival there is capacity available, begin service. Jobs which successfully enter the system require an exponential amount of service time with mean μ^{-1}, while jobs which do not enter the system are permanently lost. Assume that the capacity of the system is a positive integer number $C \in \mathbb{Z}_0$ (i.e. there can be at most C jobs being concurrently served at any given time).

Let $(X(t), t \geq 0)$ be a random process evolving on the state space $\mathcal{S} = \{0, 1, \ldots, C\}$ that records the number of jobs in the system over time. Since this is a Markov process, as with the individual k server queue introduced in Chapter 1, a collection of non-negative numbers $(\pi_i, i \in \mathcal{S})$ summing to unity that satisfy

$$\pi_i \sum_{j \in \mathcal{S}} q(i,j) = \sum_{j \in \mathcal{S}} \pi_j q(j,i), \quad \forall\, i \in \mathcal{S}$$

is defined to be the *stationary distribution* of the system (which must exist for a finite state space system). Similarly to (1.2) and (1.3) we have equilibrium equations

$$\pi_i \lambda = \pi_{i+1}(i+1)\mu\,, \qquad 0 \leq i \leq C\,. \tag{5.1}$$

These equations differ from those given by (1.2) and (1.3) due to the removal of transitions which would allow the process to evolve to a state greater than C. Setting transition rates to 0 in this way is called *truncation*. More formally, a Markov process evolving on $\mathcal{S}$ with transition rates $(q(i,j),\, i,j \in \mathcal{S})$ is truncated to form a new Markov process with transition rates $(q'(i,j),\, i,j \in \mathcal{S}')$ where $\mathcal{S}' \subseteq \mathcal{S}$ if we take

$$q'(i,j) = \begin{cases} q(i,j), & i,j \in \mathcal{S}', \\ 0, & \text{otherwise}, \end{cases}$$

and if the resulting Markov process is irreducible within $\mathcal{S}'$. Solving the truncation of (1.2) and (1.3) to (5.1) (together with the condition $\sum_{i=0}^{C} \pi_i = 1$) we find that $\pi_i = \left(\sum_{k=0}^{C} \varrho^k/k!\right)^{-1} \varrho^i/i!$, recalling $\varrho = \lambda/\mu$.

In particular the above solution to the equilibrium equations implies that the stationary probability that the system is at capacity is given by

$$\pi_C = \frac{\varrho^C/C!}{\sum_{k=0}^{C} \varrho^k/k!}\,. \tag{5.2}$$

This is the famous Erlang B formula, initially developed to guide capacity decisions for circuit switched telephony systems. Initial work by Erlang leading to this result is among the earliest known on modelling for management of random systems [99, 100, 101]. Although it is perhaps intuitively clear that there is a relationship between (5.2) and the probability a job arriving at an arbitrary time in the distant future is blocked, the theory behind this was not made rigorous until much later [102]. This property is known as Poisson arrivals see time averages (PASTA). Put simply, the PASTA property means that the probability of a state of the system as seen by an outside observer at a random time is the same as the probabilty of the state seen by an arriving job. This property holds for the Kelly and Whittle networks introduced in Chapter 1 and their truncations, necessary and sufficient conditions for the property to hold are given in [103]. Another notable feature of this formula (5.2) for the stationary distribution is that it also holds

for non-exponential service time distributions, so long as they also have finite expected value μ^{-1}, which extends (5.2) to a non-Markovian setting (see [104] for details).

In the preceding discussion we considered a system consisting of only a single resource, a more appropriate model in many cases consists of multiple resources. Let the set of resources be indexed by the set $\mathcal{J} = \{1, 2, \ldots, J\}$ and suppose that there exists a set $\mathcal{R}$ of subsets of $\mathcal{J}$ with elements which we call *routes* (due to the classic telephone application of this type of model). Now we suppose that jobs from route r arrive according to a Poisson process of rate λ_r and upon arrival request service from all resources $j \in r$ simultaneously. Jobs utilise the resources for an exponential amount of time which, for simplicity, we assume to have unit mean. Take $(\boldsymbol{X}(t), t \in \mathbb{R}_+)$ to be a stochastic process with $|\mathcal{R}|$ coordinates which records the number of jobs present in the system for each route, and define the link-route incidence matrix A with entries

$$A_{jr} = \begin{cases} 1, & j \in r, \\ 0, & j \notin r. \end{cases}$$

If the total available quantity of resource $j \in \mathcal{J}$ is C_j, then a classic *loss network* is the Markov process X with state space $\mathcal{S} = \{x \in \mathbb{Z}_+^{|\mathcal{R}|} : Ax \leq C\}$.

We saw earlier in this section that the stationary distribution for a single resource system was closely related to that of an infinite server queue, as introduced in Chapter 1, with normalising constant $\left(\sum_{k=0}^{C} \varrho^k/k!\right)^{-1}$ needed to connect the infinite state system to the finite state system. In fact, this is a specific example of a more general result for reversible processes, which we will now give and then use to determine an expression for the stationary distribution of the loss network just described.

Lemma 29 (Reproduced from [105] Lemma 3.4)**.** *If a reversible Markov process with state space $\mathcal{S}$ and equilibrium distribution $(\pi_i, i \in \mathcal{S})$ is truncated to $\mathcal{A} \subset \mathcal{S}$, the resulting Markov process is reversible and has equilibrium distribution*

$$\pi_i \left(\sum_{k \in \mathcal{A}} \pi_k \right)^{-1}, \quad i \in \mathcal{A}.$$

Letting $C_i \to \infty$ for all $i \in \mathcal{J}$ in the loss network model and associating each coordinate of $\boldsymbol{X}$ to an individual classification, we obtain a system of the type studied by Jackson and Whittle [7, 8, 75] which we explored in Chapter 1. Combining the stationary distribution expression for this infinite state system (1.5) with Lemma 29 we find that the stationary distribution for the loss network model is given by

$$\pi_x = G \prod_{r \in \mathcal{R}} \frac{\lambda_r^{x_r}}{x_r!}, \quad x \in \mathcal{S},$$

where x_r is the r-th coordinate of x and

$$G = \left(\sum_{x \in \mathcal{S}} \prod_{r \in \mathcal{R}} \frac{\lambda_r^{x_r}}{x_r!} \right)^{-1}$$

is a normalising constant. Computation of this stationary distribution is extremely difficult [106]; so a famous approximation was developed [107, 108] to approach optimisation

[109]. The model we study in Chapter 8 is of comparable complexity to the loss network model just discussed, hence we use a simulation-based approach to optimisation.

Several schemes for decentralised control of loss networks have been developed. Dynamic alternative routing is one such method which is applicable for systems where jobs of a particular route are able to be serviced by an alternative (usually larger) subset of $\mathcal{J}$ when their designated resources $r \subset \mathcal{J}$ are not available upon arrival. Trunk reservation [110, 111] is an example of a control mechanism that aims to ensure alternative routes do not increase congestion and result in reduced network performance. Other studies have also focused on optimising network performance in equilibrium using Markov decision theory [112, 113]. Notable recent work combines simulation with analysis to efficiently tackle this difficult class of optimisation problem [114] — this move towards a simulation approach mirrors the modern approach to optimising queueing networks [11] that motivates our simulation-based approach to stability in Chapter 4. In Chapter 8 we develop a simulation-based approach to optimal stationary capacity allocation for a class of loss models which is distinctly different to the loss networks introduced in this section. In Chapter 6 and Chapter 7, however, we develop a centralised (i.e. transient and state dependent) approach to provisioning of capacity for generalisations of the single resource loss network introduced in this section. We will now discuss the work which inspired our centralised approach and the methods needed to obtain our results in these chapters.

5.2 Capacity value function and Laplace–Stieltjes transforms

For many systems (e.g. circuit switched telephony networks) it may be difficult or impossible to make changes in the quantity of resource made available over short time horizons. As we will see in more detail in Chapter 7, in modern cloud computing systems it may in fact be beneficial to remove idle capacity from service (e.g. by switching off servers) during periods of low demand. One way to do this is to periodically monitor the system state and at the beginning of each period make a management decision with regards to capacity. Due to PASTA there is a clear relationship between a system's stationary distribution and performance, however the relationship between the distribution of the number of jobs in the system at some finite time t and system performance is not as clear. In [115] and [116] a highly practical performance measure is introduced: the number of jobs rejected (or accepted) during the time interval $[0, t]$. The expected value of this performance measure, which the authors call the *capacity value function*, enables system managers to determine an ideal capacity adjustment in each time period.

The analysis in [115] and [116] of a single resource loss system is based on the Laplace–Stieltjes transform, which for some function $r : \mathbb{R}_0 \to \mathbb{R}$ is defined by

$$\int_0^\infty r(t)\, \mathrm{e}^{-s\,t}\, \mathrm{d}t\,, \quad s \in \mathbb{C}\,,$$

when this quantity exists. In [115] the authors use results for orthogonal polynomials to find the Laplace–Stieltjes transform of the capacity value function and invert the obtained expression numerically to explore system behaviour. Properties of Laplace–Stieltjes transforms relating to time shift and integration are key reasons the model becomes tractable to orthogonal polynomial results after transformation. Then, in [116], a Tauberian theorem commonly known as the *final value theorem* is utilised to find a second-order approxi-

mation to the capacity value function. This theorem, which is standard to the field of modern control engineering is as follows.

Theorem 8 (Final value theorem, reproduced from [117]). *Suppose $r : \mathbb{R}_+ \to \mathbb{R}$ possesses Laplace–Stieltjes transform $\widetilde{r}$ and $s\widetilde{r}(s)$ exists and has no poles on or to the right of the imaginary axis in the s-plane, then*

$$\lim_{t\to\infty} r(t) = \lim_{s\to 0} s\widetilde{r}(s)\,.$$

In Chapter 6 we generalise the results in [115] and [116] to the case of a single resource system where the resource experiences random periods of inaccessibility. These results are useful for incorporating faulty system behaviour into loss models.

This approach to determining the capacity value function requires delicate manipulation of orthogonal polynomials to determine the Laplace–Stieltjes transform. It is usually a cumbersome process to show that the conditions of the final value theorem hold. In Chapter 7 we show how approaching determination of the capacity value function using matrix analytic methods, as introduced in the next section, can greatly simplify the modelling process and allow for a much broader class of models to be studied.

5.3 Markovian arrival processes

A Markovian arrival process is a counting process with arrivals of different classifications governed by the transitions and holding times of another finite state *background* Markov chain (see e.g. [118] and [119]). Suppose that the set $\mathcal{C}$ provides indexes for the classifications and $(N_h(t),\, t \geq 0)$ for $h \in \mathcal{C}$ counts the arrivals of classification h in the time interval $[0,\, t]$. In Chapter 7 we utilise *batch Markovian arrival processes* (BMAP), where work arrives as super-jobs consisting of a random quantity of jobs. We now provide a formal construction for BMAPs.

Definition 30 (Reproduced from [118] Definition 2.4.1). *Assume that square matrices $\{D_1, D_2, \ldots, D_N\}$ of order m have non-negative entries, D_0 has non-negative off-diagonal elements and negative diagonal elements, m is a positive integer, N is a positive integer (possibly infinite), and $D = D_0 + D_1 + \cdots + D_N$ is an infinitesimal generator. Let $D_0 = (d_{0,(i,j)})$, $D_n = (d_{n,(i,j)})$, for $n = 1, \ldots, N$. Then $(D_0, D_1, \ldots, D_N)$ defines BMAP $\big((N(t), I(t)),\, t \geq 0\big)$ as follows.*

1. *Set $N(0) = 0$.*
2. *Define a continuous-time Markov chain $(I(t),\, t \geq 0)$ by D as follows.*
3. *For state i with $d_{n,(i,i)} > 0$, for $n = 1, \ldots, N$ and $i = 1, \ldots, m$ define a Poisson process with arrival rate $d_{n,(i,i)}$ when $I(t) = i$ and otherwise 0.*
4. *For $i = 1, \ldots, m$, if $I(t) = i$ and an arrival from the imposed Poisson process $d_{n,(i,i)}$ occurs, $N(t)$ increases by n, for $1 \leq n \leq N$.*
5. *At the end of each stay in state i, with probability $d_{0,(i,j)}/(-d_{(i,i)})$ (note $d_{i,i} = d_{0,(i,i)} + d_{1,(i,i)} + \cdots + d_{N,(i,i)}$) $I(t)$ transits from state i to state j and $N(t)$ remains the same; and, with probability $d_{n,(i,j)}/(-d_{(i,i)})$, $I(t)$ transits from state i to state j and $N(t)$ increases by n, for $1 \leq n \leq N$ and $i \neq j$ and $i, j = 1, \ldots, m$.*

In Chapter 7 we show that the process governing lost jobs in the single resource loss system is a specific simple example of a BMAP. By considering multiple BMAPs together we are able to incorporate baulking behaviour into the model. In this setting we also show how to incorporate batch arrivals with bursty arrival processes. Finally, we also show that it is possible to include a finite buffer where jobs wait before service — resulting in a model that exhibits both loss and queueing behaviour. We are able to combine these BMAPs together using the linearity property of expectation, and our computation of the individual expectations is performed using the following theorem. Let $\mathbf{1}$ be a column vector with unit entries, and I be the identity matrix.

Theorem 9 (Reproduction of Theorem 2.4.2 in [118])**.** *Assume that the background process $(I(t), t \geq 0)$ is irreducible. Given initial distribution $\boldsymbol{\pi}_0$ (row vector), we have*

$$\mathbb{E}[N(t)] = \boldsymbol{\pi}\left(\sum_{j=1}^{N} jD_j\right)\mathbf{1} + \boldsymbol{\pi}_0(\exp(Dt) - I)(D - \mathbf{1}\boldsymbol{\pi})^{-1}\left(\sum_{j=1}^{N} jD_j\mathbf{1}\right), \quad t \geq 0,$$

where (row vector) $\boldsymbol{\pi}$ is the stationary distribution of I (i.e. $\boldsymbol{\pi} \geq 0$, $\boldsymbol{\pi}D = 0$, and $\boldsymbol{\pi}\mathbf{1} = 1$).

The expression in Theorem 9 is crucial to our results in Chapter 7. Further work in the area of matrix analytic methods has succeeded in finding second-order approximations to this function [120]. The slope term in these second-order approximations, when applied to the capacity value function, measures the asymptotic rate at which revenue is lost. This can be used to conduct equilibrium performance evaluation when doing so is more appropriate. In Chapter 8 we see that even these matrix analytic based methods have limitations when it comes to analysing networks of loss systems, and so in that chapter we develop a novel simulation-based method to guide management decisions.

5.4 Outline of Part II

Here we provide a summary of the subsequent chapters of Part II of this thesis. As we progress through the chapters the models we study become more complex and new methods are developed to handle the increasing complexity. The overarching theme is guiding system managers in choosing the capacity of a system that best addresses the trade-off between losing work and the cost of providing additional capacity. Each chapter is presented in an accessible and self contained fashion with notation optimised for presentation on a per-chapter basis.

Chapter 6 — Management of faulty loss systems. We consider a finite capacity Erlang loss system that alternates between active and inactive states according to a two state modulating Markov process. Work arrives to the system as a Poisson process but is blocked from entry when the system is at capacity or inactive. Blocked jobs cost the owner a fixed amount that depends on whether blockage was due to the system being at capacity or due to the system being inactive. Jobs which are present in the system when it becomes inactive pause processing until the system becomes active again.

A Laplace transform expression for the expected undiscounted revenue lost in $[0, t]$ due to blocking is found. Further, an expression for the total time discounted expected lost revenue in $[0, \infty)$ is provided. We also derive a second order approximation to the

former that can be used when the computing power to invert the Laplace transform is not available. These expressions can be used to ascribe a value to four alternatives for improving system performance: (i) increasing capacity, (ii) increasing the service rate, (iii) increasing the repair rate, or (iv) decreasing the failure rate.

Chapter 7 — Loss system models for cloud computing platforms. User demand on the computational resources of cloud computing platforms varies over time. These variations in demand can be predictable or unpredictable, resulting in time-varying and 'bursty' fluctuations in demand. Furthermore, demand can arrive in batches, and users whose demands are not met can be impatient. We demonstrate how to compute the expected revenue loss over a finite time horizon in the presence of all these model characteristics through the use of matrix analytic methods. We then illustrate how to use this knowledge to make frequent short term provisioning decisions — transient provisioning. It is seen that taking each of the characteristics of fluctuating user demand (predictable, unpredictable, batchy) into account can result in a substantial reduction of losses. Moreover, our transient provisioning framework allows for a wide variety of system behaviours to be modelled and gives simple expressions for expected revenue loss which are straightforward to evaluate numerically.

Chapter 8 — Functional form based optimisation for stochastic networks with blocking. Many stochastic networks encountered in practice are affected by blocking, where network traffic is lost due to congestion. A key decision when designing such networks is how to allocate resources to limit losses, while maintaining low costs. Roughly speaking, there are two categories of approaches for solving capacity management problems in stochastic networks: analytical and simulation-based optimisation. In this chapter we describe a hybrid approach to optimising stochastic networks with blocking where simulation is augmented with a functional form to improve efficiency. We apply this approach to a realistic example, test the method using standard and non-standard networks, and show that it can outperform the existing state-of-the-art.

The following publication has been incorporated as Chapter 6.
[121] **B. Patch**, T. Taimre, and Y. Nazarathy, Performance of faulty loss systems with persistent connections, *ACM SIGMETRICS Performance Evaluation Review*, 43.2 (2015). pp 16–18.

Contributor	Statement of contribution	%
Brendan Patch	writing of text	33
	proof-reading	33
	theoretical derivations	33
	preparation of figures	33
	initial concept	33
Thomas Taimre	writing of text	33
	proof-reading	33
	supervision, guidance	50
	theoretical derivations	33
	preparation of figures	33
	initial concept	33
Yoni Nazarathy	writing of text	33
	proof-reading	33
	supervision, guidance	50
	theoretical derivations	33
	preparation of figures	33
	initial concept	33

CHAPTER 6

Management of faulty loss systems

6.1 Introduction

As discussed in the previous chapter, loss systems are a model for processes in which jobs arrive randomly throughout time and simultaneously utilise some quantity of a resource for a time period of random length, before departure from the system. In the loss system model, if a job arrives to the system and there are not enough resources available for it to begin processing, then it is lost. Landline telephone connections between cities are a classic example. Motivated by this, the resource is often called a *link* and the arriving jobs *calls*. When there are multiple types of resource available and an arriving job may only require some subset of the resources, the model is known as a loss network. A classic review of loss networks is [95], and a more recent review is [122].

This chapter focuses on loss systems where links are prone to failure. Arrivals occur according to a Poisson process of rate λ and, if possible, immediately begin service at rate μ (i.e. the sojourn time of a job in the system is exponential with mean μ^{-1}). Links which are *active* become *inactive* at rate α and inactive links repair at rate β. Arrivals to a link when it is inactive experience *failure blocking*, and arrivals when the link is active but at full capacity experience *capacity blocking*. Blocked arrivals result in a loss of revenue — the size of which depends on the reason for blockage. In this preliminary exposition we focus on a system where existing connections on the link *persist* during link failure — continuing their service upon link reactivation. Our model and analysis is useful for managing communication systems which are subject to sabotage or adverse environmental (e.g. weather) conditions.

Our work extends results in [115] and [116]. In [115] a Laplace transform expression in terms of Charlier polynomials is given for the undiscounted value of an additional unit of capacity during the planning horizon $[0, t]$ on a link which never fails. This expression acts as a performance measure for the system. The authors also demonstrate how the expression can be used to ascribe a value to an extra unit of capacity on the link over the planning horizon. In [116] a second order approximation to the inverted expression is given. The key difference between these papers and other classic studies of loss systems (see e.g., [109]) is that they focus on transient expected performance indicators rather than approximations to equilibrium distributions of the system.

Our methodological contribution is two-fold. First, we extend the model and results of [115] and [116] to allow for link failure. Second, we obtain an expression for infinite horizon total discounted lost revenue when interest is compounded continuously at rate r. As opposed to the expression for finite horizon undiscounted lost revenue, this expression does not require numerical inversion.

The expressions that we obtain can be used to ascribe a value to four alternatives for improving system performance: (i) increasing capacity, (ii) increasing the service rate, (iii) increasing the repair rate, or (iv) decreasing the failure rate. The set of *control parameters*, denoted by $\mathcal{X} \stackrel{\text{def}}{=} \{C,\, \mu,\, \beta,\, \alpha\}$, is used to invoke these changes. We envisage that a system manager can vary the control parameters through mechanisms such as equipment purchases, training programs, and wages.

6.2 Model

Consider the Markov process $\{(N(t),\, J(t))\}_{t\in\mathbb{R}_0}$, where $N(t) \in \{0, 1, \ldots, C\}$ and represents the number of connections in use at time t, and $J(t)$ takes the value 1 if the link is active at time t and 0 otherwise. More precisely, this process has state space

$$\big\{(n, j) : n \in \{0, 1, \ldots, C\}, j \in \{0, 1\}\big\}$$

with evolution governed by the transition rates as given in Table 6.1. When the process is in state $(C, 1)$ or any state $(n, 0)$, $n \in \{0, 1, \ldots, C\}$, potential calls continue to arrive according to a Poisson process of rate λ. These calls are blocked and losses are recorded. Let $\theta > 0$ denote the revenue lost if a call is blocked when the system is in state $(C, 1)$ (capacity blocking) and $\overline{\theta} > 0$ denote the revenue lost if a call is blocked when the system is in a state $(n, 0)$ (failure blocking). When the system is in state $(C,\, 1)$ during a time

Table 6.1: Transition rates of our faulty loss system.

Transition	Rate	States
$(n,\, 1) \to (n,\, 0)$	α	$0 \le n \le C$
$(n,\, 0) \to (n,\, 1)$	β	$0 \le n \le C$
$(n,\, 1) \to (n+1,\, 1)$	λ	$0 \le n < C$
$(n,\, 1) \to (n-1,\, 1)$	$n\,\mu$	$0 < n \le C$

interval $[a,\, b)$, an expected loss of $\lambda\,\theta\,(b-a)$ is incurred due to the Poisson call arrival process of rate λ. Similarly, if the system is in a state $(n,\, 0)$ during a time interval $[a,\, b)$ then a loss of $\lambda\,\overline{\theta}\,(b-a)$ is expected. Therefore, the expected lost revenue during $[0,\, t]$ for a link with $N(0) = n$ and $J(0) = j$, which we denote by $r^{(j)}_{n,\mathcal{X}}(t)$, can be written as

$$\mathbb{E}\left[\int_0^t \lambda\left(\overline{\theta}\;\mathrm{I}_{\{J(\tau)=0\}} + \theta\,\mathrm{I}_{\{N(\tau)=C\}}\,\mathrm{I}_{\{J(\tau)=1\}}\right)\mathrm{d}\tau \;\middle|\; \big(N(0), J(0)\big) = (n, j)\right]. \tag{6.1}$$

The function defined in (6.1) is analogous to the capacity value function of [115]. The value over the planning horizon of $[0,\, t]$ of a parameter adjustment from $\mathcal{X}$ to $\widetilde{\mathcal{X}} \stackrel{\text{def}}{=} \{\widetilde{C},\, \widetilde{\mu},\, \widetilde{\beta},\, \widetilde{\alpha}\}$

is

$$\Delta r^{(j)}_{n,t}(\mathcal{X}, \widetilde{\mathcal{X}}) \stackrel{\text{def}}{=} r^{(j)}_{n,\mathcal{X}}(t) - r^{(j)}_{n,\widetilde{\mathcal{X}}}(t)\,. \tag{6.2}$$

We call this the *finite horizon performance value function.*

Furthermore, the expected rate at which revenue is lost from the system is $\lambda\,\theta$ when it is in state $(C,\,1)$ and $\lambda\,\overline{\theta}$ when it is in a state $(n, 0)$. So the expected rate at which revenue is lost at time t is

$$\ell^{(j)}_{n,\mathcal{X}}(t) \stackrel{\text{def}}{=} \lambda\,\mathbb{E}\Big[\overline{\theta}\;\mathrm{I}_{\{J(t)=0\}} + \theta\,\mathrm{I}_{\{N(t)=C\}}\,\mathrm{I}_{\{J(t)=1\}}\;\Big|\;\big(N(0), J(0)\big) = (n, j)\Big]\,.$$

Assuming that interest is compounded continuously at rate r, the discounted value of the lost revenue during $[0,\,\infty)$ is

$$\mathscr{L}^{(j)}_{n,\mathcal{X}}(r) \stackrel{\text{def}}{=} \int_0^\infty \ell^{(j)}_{n,\mathcal{X}}(t)\,\mathrm{e}^{-r\,t}\mathrm{d}t\,. \tag{6.3}$$

Note that this functional is equivalently the Laplace transform.

Similar to (6.2),

$$\Delta\mathscr{L}^{(j)}_{n,r}(\mathcal{X},\,\widetilde{\mathcal{X}}) \stackrel{\text{def}}{=} \mathscr{L}^{(j)}_{n,\mathcal{X}}(r) - \mathscr{L}^{(j)}_{n,\widetilde{\mathcal{X}}}(r) \tag{6.4}$$

gives the difference in total time discounted value obtained from variations in the control parameters. We call this the *discounted performance value function.*

It is straightforward to combine (6.2) or (6.4) with a budget constraint to obtain an optimization problem that can be solved, and thus help direct the manager of the system on how best to vary the control parameters.

6.3 Results

In this section we give explicit expressions for (6.1) and (6.3) that can be used to calculate (6.2) and (6.4). Let $r^{(j)}_{n,\mathcal{X}}(t\,|\,x)$ be $r^{(j)}_{n,\mathcal{X}}(t)$ conditional on the fact that the first time that the link departs from state $(n,\,j)$ is x. Now,

$$r^{(1)}_{n,\mathcal{X}}(t\,|\,x) = \begin{cases} 0\,, & n < C,\ t < x\,, \\ \theta\lambda t\,, & n = C,\ t < x\,, \\ \frac{n\,\mu r^{(1)}_{n-1,\mathcal{X}}(t-x) + \lambda r^{(1)}_{n+1,\mathcal{X}}(t-x) + \alpha r^{(0)}_{n,\mathcal{X}}(t-x)}{n\,\mu+\lambda+\alpha}\,, & n < C,\ t \geq x\,, \\ \theta\lambda x + \frac{C\,\mu\,r^{(1)}_{C-1,\mathcal{X}}(t-x) + \alpha\,r^{(0)}_{C,\mathcal{X}}(t-x)}{C\,\mu+\alpha}\,, & n = C,\ t \geq x\,, \end{cases} \tag{6.5}$$

and

$$r^{(0)}_{n,\mathcal{X}}(t\,|\,x) = \begin{cases} \overline{\theta}\,\lambda\,t\,, & t < x\,, \\ \overline{\theta}\lambda x + r^{(1)}_{n,\mathcal{X}}(t-x)\,, & t \geq x\,. \end{cases} \tag{6.6}$$

For $(N(0), J(0)) = (n, j)$, $j \in \{0, 1\}$ let $X_n^{(j)}$ be the time until the first transition, with distribution $F_n^{(j)}$, and consider the Riemann–Stieltjes integral

$$r_{n,\mathcal{X}}^{(j)}(t) = \int_0^\infty r_{n,\mathcal{X}}^{(j)}(t \,|\, x)\, \mathrm{d}F_n^{(j)}(x)\,. \tag{6.7}$$

Due to the Markovian nature of the model, $F_n^{(1)}$ is exponential with parameter $\lambda + n\,\mu + \alpha$ when $n < C$ and $C\,\mu + \alpha$ when the link is at capacity and $F_n^{(0)}$ is exponential with parameter β. Substituting (6.5), (6.6) into (6.7), and removing the $\mathcal{X}$ subscript for brevity, we obtain

$$\begin{aligned}
r_0^{(1)}(t) &= \int_0^t \Big(\lambda r_1^{(1)}(t-x) + \alpha r_0^{(0)}(t-x)\Big)\, \mathrm{e}^{-(\lambda+\alpha)x}\mathrm{d}x\,,\\
r_n^{(1)}(t) &= \int_0^t \Big(n\,\mu\, r_{n-1}^{(1)}(t-x) + \lambda r_{n+1}^{(1)}(t-x) + \alpha\, r_n^{(0)}(t-x)\Big)\, \mathrm{e}^{-(n\,\mu+\lambda+\alpha)x}\mathrm{d}x\,, \quad 0 < n < C\\
r_C^{(1)}(t) &= \int_0^t \Big(C\,\mu\, r_{C-1}^{(1)}(t-x) + \alpha\, r_C^{(0)}(t-x)\Big)\, \mathrm{e}^{-(C\,\mu+\alpha)}\mathrm{d}x + \frac{\theta\lambda}{C\,\mu+\alpha}\left(1 - \mathrm{e}^{-(C\,\mu+\alpha)t}\right),\\
r_n^{(0)}(t) &= \int_0^t \beta\, r_n^{(1)}(t-x)\, \mathrm{e}^{-\beta\, x}\mathrm{d}x + \frac{\overline{\theta}\,\lambda}{\beta}\big(1 - \mathrm{e}^{-\beta t}\big)\,.
\end{aligned}$$

Upon taking the Laplace–Stieltjes transform $\widetilde{r}_n^{(j)}(s) = \int_0^\infty r_n^{(j)}(t)\, \mathrm{e}^{-s\,t}\, \mathrm{d}t$ this becomes

$$\begin{aligned}
\widetilde{r}_0^{(1)}(s) &= \frac{\lambda}{s+\lambda+\alpha}\, \widetilde{r}_1^{(1)}(s) + \frac{\alpha}{s+\lambda+\alpha}\, \widetilde{r}_0^{(0)}(s)\,,\\
\widetilde{r}_n^{(1)}(s) &= \frac{n\,\mu\, \widetilde{r}_{n-1}^{(1)}(s)}{s+n\,\mu+\lambda+\alpha} + \frac{\lambda\, \widetilde{r}_{n+1}^{(1)}(s)}{s+n\,\mu+\lambda+\alpha} + \frac{\alpha\, \widetilde{r}_n^{(0)}(s)}{s+n\,\mu+\lambda+\alpha}\,, \qquad 0 < n < C\\
\widetilde{r}_C^{(1)}(s) &= \frac{1}{s+C\,\mu+\alpha}\left(C\,\mu\, \widetilde{r}_{C-1,1}(s) + \alpha\, \widetilde{r}_{C,0}(s) + \frac{\theta\lambda}{s}\right),\\
\widetilde{r}_n^{(0)}(s) &= \frac{1}{s+\beta}\left(\beta\, \widetilde{r}_{n,1}(s) + \frac{\overline{\theta}\,\lambda}{s}\right).
\end{aligned} \tag{6.8}$$

Which implies

$$\begin{aligned}
\widetilde{r}_0^{(1)}(s) &= \frac{1}{s+\lambda+\alpha}\left(\lambda\, \widetilde{r}_1^{(1)}(s) + \frac{\alpha}{s+\beta}\left(\beta\, \widetilde{r}_0^{(1)}(s) + \frac{\overline{\theta}\,\lambda}{s}\right)\right),\\
\widetilde{r}_n^{(1)}(s) &= \frac{n\,\mu\, \widetilde{r}_{n-1,1}(s) + \lambda\, \widetilde{r}_{n+1}^{(1)}(s) + \frac{\alpha}{s+\beta}\left(\beta\, \widetilde{r}_n^{(1)}(s) + \frac{\overline{\theta}\,\lambda}{s}\right)}{s+n\,\mu+\lambda+\alpha}\,, \quad 0 < n < C\,,\\
\widetilde{r}_C^{(1)}(s) &= \frac{1}{s+C\,\mu+\alpha}\left(C\,\mu\, \widetilde{r}_{C-1}^{(1)}(s) + \frac{\alpha}{s+\beta}\left(\beta\, \widetilde{r}_C^{(1)}(s) + \frac{\overline{\theta}\,\lambda}{s}\right) + \frac{\theta\lambda}{s}\right),\\
\widetilde{r}_n^{(0)}(s) &= \frac{1}{s+\beta}\left(\beta\, \widetilde{r}_n^{(1)}(s) + \frac{\overline{\theta}\,\lambda}{s}\right).
\end{aligned} \tag{6.9}$$

Furthermore,

$$\widetilde{r}_1^{(1)}(s) = \frac{s+\lambda+\alpha-\alpha\,\beta\,(s+\beta)^{-1}}{\lambda}\,\widetilde{r}_0^{(1)}(s) - \frac{\alpha\,\overline{\theta}}{s\,(s+\beta)}\,, \tag{6.10}$$

$$\widetilde{r}_{n+1}^{(1)}(s) = \frac{s+n\,\mu+\lambda+\alpha-\alpha\,\beta\,(s+\beta)^{-1}}{\lambda}\,\widetilde{r}_n^{(1)}(s) - \frac{n\,\mu}{\lambda}\,\widetilde{r}_{n-1}^{(1)}(s) - \frac{\alpha\,\overline{\theta}}{s\,(s+\beta)}\,, \quad 0<n<C\,, \tag{6.11}$$

$$\widetilde{r}_C^{(1)}(s) = \frac{1}{s+C\,\mu+\alpha-\alpha\,\beta\,(s+\beta)^{-1}}\left(C\,\mu\,\widetilde{r}_{C-1}^{(1)}(s) + \frac{\theta\,\lambda}{s} + \frac{\alpha\,\overline{\theta}\,\lambda}{s\,(s+\beta)}\right). \tag{6.12}$$

The solution of (6.9)–(6.12) gives the result. Now let

$$B(s) \stackrel{\text{def}}{=} \alpha\,\overline{\theta}\,\lambda/\big(s\,(s+\beta)\big(s+\alpha-\alpha\,\beta\,(s+\beta)^{-1}\big)\big)$$

and set

$$Q_n(s) \stackrel{\text{def}}{=} \widetilde{r}_{n,1}(s) - B(s)\,.$$

We can recast (6.10) and (6.11) as

$$Q_1(s) = \frac{s-\alpha\,\beta\,(s+\beta)^{-1}+\lambda+\alpha}{\lambda}\,Q_0(s) \tag{6.13}$$

$$Q_{n+1}(s) = \frac{s-\alpha\,\beta\,(s+\beta)^{-1}+n\,\mu+\lambda+\alpha}{\lambda}\,Q_n(s) - \frac{n\,\mu}{\lambda}\,Q_{n-1}(s)\,, \quad 0<n<C\,. \tag{6.14}$$

Or alternatively

$$P_{n+1}(\xi) = \big(\xi-(d\,n+f)\big)\,P_n(\xi) - n\,(g\,n+h)\,P_{n-1}(\xi)$$

where $d=-\mu/\lambda$, $f=-1$, $g=0$, $h=\mu/\lambda$, and $\xi=(s-\alpha\,\beta\,(s+\beta)^{-1}+\alpha)/\lambda$. This recurrence relation describes the class of Meixner polynomials (see e.g. [123]). If, as is the case here, the recurrence relation can further be written as

$$P_{n+1}(s) = \big(\xi - d\,(n+h\,d^{-2})\big)P_n(\xi) - h\,n\,P_{n-1}(\xi)\,, \tag{6.15}$$

then the solution is known to be $P_n(s) = d^n\,C_n^{(\ell)}(\xi/d)$, where $\ell = h/d^2 = \lambda/\mu$ and

$$C_n^{(\lambda/\mu)}(\xi) = \sum_{k=0}^{n}\binom{n}{k}\binom{-\lambda\,\xi/\mu}{k}(-\lambda/\mu)^{n-k}\,k!\,, \tag{6.16}$$

is a Charlier polynomial. These polynomials can be generally expressed in terms of Laguerre polynomials via the relation $C_n^{(\ell)}(\xi) = n!\,L_n^{(\xi-n)}(\ell)$. Hence,

$$P_n(s) = d^n\,n!\,L_n^{(\xi/d-n)}(\ell).$$

It follows that the solution to (6.13) and (6.14) is $Q_n(s) = D(s)\,P_n\,(s)$, where $D(s)$ is a function of s and $P_n(s)$ was given previously.

Therefore the solution to (6.10) and (6.11) is

$$\widetilde{r}_n^{(1)}(s) = Q_n(s) + B(s) = D(s)\, P_n\,(s) + B(s).$$

Using (6.12) we obtain

$$D(s)\, P_C(s) + B(s) = \frac{C\,\mu\left(D(s)\, P_{C-1}(s) + B(s)\right) + \frac{\theta\,\lambda}{s} + \frac{\alpha\,\overline{\theta}\,\lambda}{s\,(s+\beta)}}{s + C\,\mu + \alpha - \alpha\,\beta\,(s+\beta)^{-1}},$$

which gives us

$$D(s) = (\theta\,\lambda/s)/\left(P_C(s)\left(s + C\,\mu + \alpha - \alpha\,\beta\,(s+\beta)^{-1}\right) - C\,\mu\,P_{C-1}(s)\right).$$

Substituting this into the expression for $\widetilde{r}_n^{(1)}(s)$ above and combining with the definition of a Laguerre polynomial gives the following expression for the Laplace–Stieltjes transform of the undiscounted value of lost revenue for our faulty loss system during the planning horizon $[0, t]$.

Proposition 31.

$$\widetilde{r}_{n,\mathcal{X}}^{(j)}(s) = \frac{P_n(s)\,\theta\,\lambda}{s\left(s\,P_C(s) + C\,\mu\left(P_C(s) - P_{C-1}(s)\right)\right)} + B(s) \tag{6.17}$$

and $\widetilde{r}_{n,\mathcal{X}}^{(0)}(s) = \left(\beta\,\widetilde{r}_{n,\mathcal{X}}^{(1)}(s) + \overline{\theta}\,\lambda/s\right)/(s+\beta)$, *where*

$$B(s) = \alpha\,\overline{\theta}\,\lambda/\left(s\,(s+\beta)\left(s + \alpha - \alpha\,\beta\,(s+\beta)^{-1}\right)\right), \tag{6.18}$$

$$A_i(s) = s + \alpha - \alpha\,\beta\,(s+\beta)^{-1} + i\,\mu, \quad \textit{and} \tag{6.19}$$

$$P_n(s) = \sum_{k=0}^{n} \binom{n}{k} \lambda^{-k} \prod_{i=0}^{k-1} A_i(s)\,. \tag{6.20}$$

This generalizes the result in [115] for a loss system that never fails. Taking $\alpha = 0$ (no failures) or $\beta \to \infty$ (instantaneous repairs) recovers equation (15) of [115].

While this expression seems nice and compact, it does require a summation involving binomial coefficients, making inversion numerically cumbersome. In this case approximations to the inverted expression are a sensible alternative. We will now study approximations for a system with $J(0) = 1$, it is a simple extension to examine a system with $J(0) = 0$.

A first order linear approximation is given by

$$r_{n,\mathcal{X}}^{(1)}(t) = a\,t + \mathrm{o}(t)\,, \tag{6.21}$$

where $a = \lambda\,(\theta\,\pi + \overline{\theta}\,\overline{\pi})$, $\mathrm{o}(t)/t \to 0$ as $t \to \infty$,

$$\pi = \left(\frac{\varrho^C}{C!}\right)\left((1+\psi)\sum_{n=0}^{C}\frac{\varrho^n}{n!}\right)^{-1},$$

and $\overline{\pi} = \psi/(1+\psi)$, with $\varrho = \lambda/\mu$, and $\psi = \alpha/\beta$.

The values π and $\overline{\pi}$ represent the equilibrium probability that $\{(N(\cdot),\, J(\cdot))\}$ is in state

$(C, 1)$ or, respectively, a state $(n, 0)$. Combined with the Poisson call arrival process of rate λ it is clear that a is the equilibrium rate of loss. Note the similarity of π to Erlang's B formula, which is retrieved for $\alpha = 0$ or $\beta \to \infty$.

The error introduced to (6.21) as it transitions to equilibrium can be corrected by the following second order linear approximation.

Proposition 32.

$$r_{n,1}(t) = a_r\, t + b_{r,n} + \mathrm{o}(1)\,, \tag{6.22}$$

with $\mathrm{o}(1) \to 0$ *as* $t \to \infty$, $a_r = \lambda\,(\theta\,\pi + \mathring{\theta}\,\overline{\pi})$

$$\pi = \left(\frac{\varrho^C}{C!}\right)\left((1+\psi)\sum_{n=0}^{C}\frac{\varrho^n}{n!}\right)^{-1}\,, \quad \overline{\pi} = \frac{\psi}{1+\psi}\,,$$

$\varrho = \lambda/\mu$, $\psi = \alpha/\beta$

$$b_{r,n} = \frac{\gamma_{1,n} + 2\theta\lambda + \gamma_2\gamma_3 - 2a_r\gamma_4}{2\beta\gamma_4(1+\psi)}\,,$$

where

$$\begin{aligned}
\gamma_{1,n} &= 2\,(1+\psi)\,\beta\,\theta\,\lambda\,g_1(n)\,,\\
\gamma_2 &= \alpha\,\mathring{\theta}\,\lambda - (1+\psi)\,\beta\,a_r\,,\\
\gamma_3 &= -2\,\psi/\beta + 2\,(1+\psi)\,g_1(C) + C\,\mu\,\big(g_2(C) - g_2(C-1)\big)\,,\\
\gamma_4 &= 1 + \psi + C\,\mu\,\big(g_1(C) - g_1(C-1)\big)\,,\\
g_1(n) &= \frac{\psi+1}{\mu}\sum_{k=1}^{n}\varrho^{-k}\binom{n}{k}(k-1)!\,, \ \textit{and} \qquad (6.23)\\
g_2(n) &= \frac{2\,(\alpha+\beta)^2}{\beta^2\,\mu^2}\sum_{k=2}^{n}\varrho^{-k}\binom{n}{k}(k-1)!\sum_{m=1}^{k-1}\frac{1}{m} - \frac{2\,\psi}{\beta\,\mu}\sum_{k=1}^{n}\varrho^{-k}\binom{n}{k}(k-1)!\,. \qquad (6.24)
\end{aligned}$$

This result generalizes Theorem 4.1 in [116]. Again, their result can be retrieved by taking $\alpha = 0$ or $\beta \to \infty$.

Finally, observe that $\ell^{(j)}_{n,\mathcal{X}}(t) = \partial r^{(j)}_{n,\mathcal{X}}(t)/\partial t$. Hence since $r^{(j)}_{n,\mathcal{X}}(0) = 0$, using the properties of the Laplace transform,

$$\mathscr{L}^{(j)}_{n,\mathcal{X}}(r) = r^{-1}\,\widetilde{r}^{(j)}_{n,\mathcal{X}}(r)\,.$$

Proof of Proposition 32. Differentiation of (6.20) and evaluation at $s = 0$ yields the expressions for $g_1(n)$ and $g_2(n)$. It is also easily seen that $P_n(0) = 1$. Note that

$$A_n(s) \stackrel{\text{def}}{=} s\,\widetilde{r}^{(1)}_n(s) - \frac{\theta\lambda E}{s} = \frac{s\,N(s)\,D_2(s) - s\,\alpha\,\theta\,\lambda\,D_1(s) - \theta\,\lambda\,\pi\,D_1(s)\,D_2(s)\,/\,s}{D_1(s)\,D_2(s)}\,, \tag{6.25}$$

where

$$N(s) = P_n(s)\left(-s\,\alpha\,\theta\,\lambda\left(s+\alpha-\frac{\alpha\,\beta}{s+\beta}\right)\Big/ D_2(s) + \theta\,\lambda + \frac{\alpha\,\theta\,\lambda}{s+\beta}\right)$$
$$D_1(s) = s\left(P_C(s)\left(s + C\,\mu + \alpha - \frac{\alpha\,\beta}{s+\beta}\right) - C\,\mu P_{C-1}(s)\right), \quad \textit{and}$$
$$D_2(s) = s\,(s+\beta)\left(s+\alpha-\frac{\alpha\,\beta}{s+\beta}\right).$$

Now define

$$G_C(s) \stackrel{\text{def}}{=} (s + C\mu)P_C(s) - C\mu P_{C-1}(s)$$

so that the denominator of the first term on the right hand side of (6.17) is $s\,G_C(s)$. From (6.16) we see that

$$C_n^{(\lambda/\mu)}(0) = (-\lambda/\mu)^n\,,$$

so that G_C has a zero at $s = 0$. Consequently we can write $G_C(s) = s\,F_C(s)$ for some polynomial F_C. Recall that the zeros of Charlier polynomials are all real valued (see e.g. [123]). For each zero of $C_n^{(\lambda/\mu)}$, say x, again by (6.16), we see that there are two zeros of P_n at points satisfying

$$\frac{-\alpha-\beta-x\,\mu \pm \sqrt{-4\,x\,\beta\,\mu + (\alpha+\beta+x\,\mu)^2}}{2}\,.$$

Hence, given the properties of our parameters, the zeros of G_C are all real. Furthermore, upon substituting (6.20) into $G_C(s)$ we obtain

$$G_C(s) =$$
$$s\sum_{k=0}^{C}\binom{C}{k}\lambda^{-k}\prod_{i=0}^{k-1}(s+\alpha-\alpha\beta(s+\beta)^{-1}+i\,\mu) + C\,\mu\lambda^{-C}\prod_{i=0}^{C-1}(s+\alpha-\alpha\beta(s+\beta)^{-1}+i\,\mu)$$
$$+\,C\,\mu\sum_{k=0}^{C-1}\left(\binom{C}{k}-\binom{C-1}{k}\right)\lambda^{-k}\prod_{i=0}^{k-1}(s+\alpha-\alpha\beta(s+\beta)^{-1}+i\,\mu)\,.$$

In this form it is clear that for $s > 0$ we have $G_C(s) > 0$, implying that the remaining zeros of G_C are all non-positive. Thus the rational function $s\,\widetilde{r}_C^{(1)}(s)$ has one pole at $s = 0$ with all the other poles real and negative. It then follows by a standard Tauberian theorem (final value) that $\lim_{s\to 0} A_n(s)$ exists and is equal to $b_{r,n}$.

For the gradient term, note that the denominator in (6.25) and its first three derivatives are all equal to zero as $s \to 0$, implying that the numerator and its first three derivatives must also be equal to zero as $s \to 0$. Using (6.23) we have that $1 + \psi + C\,\mu\big(g_1(C) -$

$g_1(C-1)\big)$ is equal to

$$
\begin{aligned}
&1+C\,(1+\psi)\left(\sum_{k=1}^{C}\binom{C}{k}\left(\frac{\mu}{\lambda}\right)^{k}(k-1)!-\sum_{k=1}^{C-1}\binom{C-1}{k}\left(\frac{\mu}{\lambda}\right)^{k}(k-1)!\right)\\
&=1+C\,(1+\psi)\left(\sum_{k=1}^{C-1}\binom{C-1}{k-1}\left(\frac{\mu}{\lambda}\right)^{k}(k-1)!\right)+(1+\psi)\,C!\left(\frac{\mu}{\lambda}\right)^{C}\\
&=1+C!\,(1+\psi)\left(\sum_{k=0}^{C-1}\binom{C-1}{k-1}\left(\frac{\mu}{\lambda}\right)^{C-k}/\,k!\right)\\
&=(1+\psi)\,\frac{C!}{\varrho^{C}}\left(\sum_{j=0}^{C}\frac{\varrho^{j}}{j!}\right)=\frac{1+\psi}{(1+\psi)\,\pi-\psi}\,.
\end{aligned}
$$

This only holds for π as given in the theorem and is necessary for the third derivative of the numerator in (6.25) to equal zero as $s \to 0$, which proves that the given a is the correct gradient in the approximation. □

6.4 Illustration

Consider a system where $\mathcal{X} = \{6,\, 3,\, 0.5,\, \cdot\}$, $\lambda = 3$, $\theta = 1$, $\overline{\theta} = 2$, and $J(0) = 1$. Figure 6.1 displays the expected lost revenue of this system over planning horizons $t \in [0,\, 7]$. We see that the second order approximation (dashed) converges to the numerically-inverted Laplace transform (solid). The left panel is a system where there is no failure ($\alpha = 0$) and in the right panel ($\alpha = 0.002$) the system is expected to fail approximately as often as 1500 calls arrive and then take six calls worth of time to repair. It can be seen from comparing the left and right panels that without accounting for these faults a (potentially substantial) error in the evaluation of expected losses can occur.

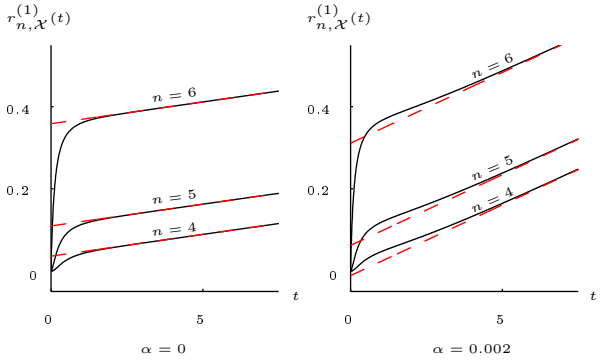


Figure 6.1: Expected lost revenue in $[0,\, t]$ without failure (left) and with failure (right).

Now consider the same system but with $\alpha = 0.5$. Figure 6.2 shows the discounted ($r = 0.1$) performance function (i.e. increase in revenue) that occurs when either the failure rate is decreased (left) or the repair rate is increased (right). We see that, for this system, increasing β increases revenue at a decreasing rate, while decreasing α increases

revenue at an increasing rate. As $\beta \to \infty$ the right graph will asymptote to the vertical intercept of the left graph.

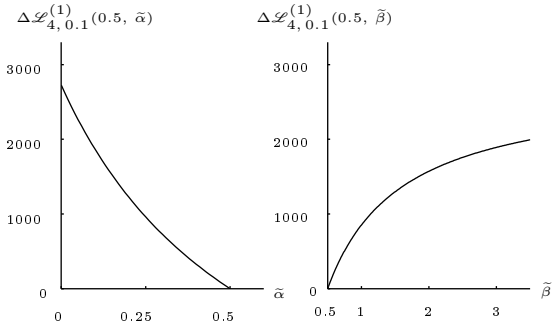


Figure 6.2: Discounted performance for decreases in failure rate (left) or increases in repair rate (right).

6.5 Concluding remarks

The performance value functions introduced here could play a role in more complex networks, in which a routing decision plays a role. Relaxing the persistent connections assumption to permit disconnection when the system is inactive would be both interesting and practical. It would also be useful to generalize these results by replacing the exponential distributions used with phase-type distributions. In the next chapter we will see another method for approaching the type of model studied in this chapter that allows for a broader class of generalisations to be approached.

The following publication has been incorporated as Chapter 7.
[124] **B. Patch** and T. Taimre. Transient provisioning and performance evaluation for cloud computing platforms: A capacity value approach, *Performance Evaluation*, 118 (2018). pp. 289–314.

Contributor	Statement of contribution	%
Brendan Patch	writing of text	50
	proof-reading	50
	theoretical derivations	50
	preparation of figures	50
	initial concept	50
Thomas Taimre	writing of text	50
	proof-reading	50
	supervision, guidance	100
	theoretical derivations	50
	preparation of figures	50
	initial concept	50

CHAPTER 7

Loss system models for cloud computing platforms

7.1 Introduction

Highly complex systems are becoming an integral contributor to the productivity of many industries. The introduction of these systems is accompanied by an increase in the demand for computational resources. Distributed cloud computing platforms have emerged as the leading method for provision of these resources to end users. In addition to the environments available online (e.g. Amazon EC2, Microsoft Azure, Google AppEngine, GoGrid), many private organizations and universities now have computer clusters that allow users to distribute computing tasks across many nodes. This substantially reduces the need for each user to have expensive individually held computing resources that become idle when not needed or which are impractical to store at the users' geographical location.

Distributed computing constitutes a substantial portion of the energy consumption in modern computer and communication networks [125]. As such, well designed provisioning policies, which match the availability of resources with the demand for resources remains an active area of research [125]. An obvious avenue to reducing the energy use of a distributed cloud platform is to switch compute nodes off, or place them into a power saving mode, when they are not needed. For example, in [126] it is estimated that perfectly provisioning capacity to match demand in a production compute cluster at Google would result in a 17–22% reduction in energy use.

Resource allocation problems of this type naturally fall into the realm of queueing theory. Specifically, the *loss network model*, as discussed in the previous two chapters, has been extensively used to analyse circuit switched systems in which tasks arrive randomly throughout time, require a random service time, and are lost if the resources required to begin their service are not available at the time of their arrival (see for example [95, 122]). In these models, the key quantity of system capacity is usually viewed as static, unable to be altered in response to short term fluctuations in system demand. As such, work on capacity selection is typically based on equilibrium properties of the system. Probably the most famous result of this type is Erlang's [127] expression for the probability that a task arriving to the system in steady state is blocked from entry, when tasks arrive according to a Poisson process with rate λ, have a mean service time of μ^{-1}, and there

are m servers:

$$\frac{(\lambda/\mu)^m/m!}{\sum_{i=0}^{m}(\lambda/\mu)^i/i!}. \tag{7.1}$$

Much of the literature on the analysis of cloud computing platforms, which we review in Section 7.1.1, develops performance measures (e.g. probability of a blocked task, waiting times, response times) from an equilibrium perspective. In order to improve the matching between provisioned capacity and demand throughout time, and to achieve the energy savings which motivate our work, it is necessary to take a transient view of the system. Due to the elasticity, or short term capacity flexibility, of distributed cloud computing platforms this is especially relevant in our case [125].

Moving away from equilibrium analysis of these systems in favour of transient analysis allows provisioning to be performed over short time intervals in a way that is dependent on the current state of the system and knowledge of the arrival rate over the near future. Since obtaining analytic results for the transient distribution of loss network type systems is notoriously difficult, one often resorts to numerical inversion of Laplace transforms [128, 129] or approximations [130]. In [115], its companion [116], and more recently [131] and also as discussed in the previous chapter, a useful alternative to the consideration of the transient distribution for queueing type models is proposed. In these papers the authors assume that tasks which fail to enter a loss system due to capacity constraints result in the system's manager incurring a predetermined amount of lost revenue. By comparing the amount of lost revenue during a finite time interval $[0, t]$ that results from different capacity choices, the authors are able to determine buying and selling prices for a unit of capacity using only information on the value of lost tasks and the current number of tasks in the system. They call the function underlying these rules the *capacity value function*.

The results in [116] and [115] are, however, derived using delicate manipulations of orthogonal polynomials, and it is difficult to use the same analytical techniques to generalize these findings to models which are more applicable to the distributed cloud computing setting. In this chapter we overcome this issue by making the pivotal observation that the capacity value function can be expressed in terms of matrix inverses and exponentials using *matrix analytic methods* (MAMs). Moreover, by adding a term to the capacity value function that reflects the operating costs of maintaining different levels of capacity over time we allow the trade-off between energy use and service degradation, that is controlled through the provisioning decision, to be explicitly modeled.

Based on these observations, in this chapter we show how to utilize the well established MAM literature to effectively obtain a transient performance measure, similar to the capacity value function, for a wide variety of potential cloud computing models. The extension of the capacity value function methodology to more general settings is the main contribution of this chapter. Although our framework is widely applicable, for clarity we illustrate it using a model that incorporates the following features:

- Batch jobs.
- Bursty arrival process (predictable and unpredictable).
- A buffer.
- Abandonments due to impatience.

In the outlook we provide a more detailed discussion on the types of settings we envisage our framework is applicable to. In general we are concerned with any cloud system

where tasks are sent to the system by users with the aim of undergoing processing before eventual departure. We envisage that in order to be processed the tasks may require access to a database or directory, specific software (e.g. a compiler), and/or specific hardware (e.g. CPU cores or RAM). Throughout the chapter we have emphasized the use of our framework on cloud platforms, there is nothing however in principle that stops it from being applied to cloud infrastructure or applications.

In our model tasks arrive to the system according to a *batch Markovian arrival process* (BMAP). We call an arrival a job and each server request that a job makes is a task. Specifically, allowing batch arrivals means that jobs may request a random number of units of server upon arrival. When the random variable governing the number of tasks that may be requested by a job has higher expectation or takes on a greater range of values we consider the arrival process to be more 'batchy'. It is important to highlight that allowing tasks to arrive in batches can be viewed as modeling jobs as Erlang distributed with a random shape parameter, which is substantially more general than the exponential distribution employed in [116] and [115]. Our notion of a batch is similar to the notion of a 'supertask' in [133].

In addition, this arrival process allows time varying behaviour to be modelled. We incorporate 'burstiness' (or unpredictability) into our model by using the property of BMAPs that the arrival rate of tasks for these processes may change randomly throughout time according to an underlying modulating Markov process. For simplicity we suppose that this underlying process alternates between a baseline state and a state where the arrival rate is increased. We view the difference between the baseline arrival rate and the randomly increased arrival rate, as well the frequency with which the randomly increased rate is expected to occur, as measures of the system's burstiness. For example, if the system experiences large, frequent, unpredictable increases in the arrival rate then we would say that the system is more bursty than a system experiencing infrequent minor increases in the arrival rate. It will become clear to the reader how this simplifying assumption can easily be relaxed to permit a modulating Markov process with any finite number of states, rather than just two (and we provide details on this in the concluding section). In addition, our framework can be applied in the case that it is known when a change in the arrival rate will occur during our planning horizon. We call instances of known changes in the arrival rate predictable bursts.

The presence of a buffer allows a cloud platform provider to store tasks which arrive when all of the servers are in use, so that the tasks may be processed once a server becomes available. In many cases it may be possible for a user to be aware that their task is waiting in the buffer, rather than actively undergoing service. In a competitive environment the user may choose in such a circumstance to attempt service with an alternative cloud platform provider (i.e. abandon the system). Accounting for user behaviour such as this may be highly beneficial to cloud providers, and a key advantage of our framework is that such extensions are often easily incorporated.

Our key result is an explicit matrix expression for the expected lost revenue during $[0, t]$ when m servers are active, tasks may wait in a buffer of size r, there is a cost per unit time per unit of server, and a potentially different cost per unit time per unit of buffer. In addition to the revenue lost from a task failing to enter the system we also suppose that when tasks which are waiting for service in the buffer abandon the system a loss is also incurred by the system manager. We suppose that tasks will wait an exponential amount of time before abandoning.

In the context of the model just described we show how our transient performance measure (expected lost revenue during $[0, t]$) can be used by system managers to make

short term provisioning decisions. Furthermore, analysing the performance of the system in terms of revenue losses due to blockages and abandonments is particularly relevant in this setting since it realistically reflects the penalties imposed on service providers associated with violations of service level agreements. Our framework provides a key prerequisite for the development of a capacity provisioning module that could be combined with other tools in a realistic cloud setting to provide improvements in performance. In Figure 7.1 we illustrate how our framework would combine with existing cloud architecture such as micro services, service composition, and load balancing (summarized as a scheduler for simplicity) and a cloud computing platform (represented as a cluster). In this setting, these other aspects of the cloud architecture will have implications for the arrival rate of jobs to a particular cloud computing platform, upon taking this into account our framework provides a service that is complementary to the existing architecture.

Any improvements in performance from using our framework to provision a cloud computing platform will clearly depend on the parametrisation of the real world system to which it is applied. Parametrisation of MAPs is an area of ongoing research (see e.g., [134, 135, 136, 137, 138]), and tools are becoming readily available for real world managers to utilize. Finally, some cloud providers now allow users to set automatic decision rules on when to scale up or down capacity (see e.g., [139]) — after parametrisation of an appropriate application specific model our framework could be applied to guide these decisions.

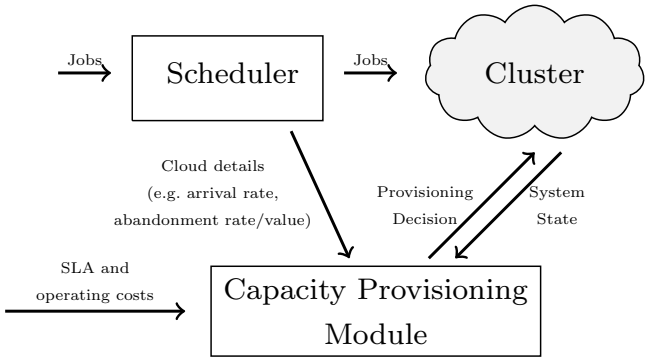


Figure 7.1: Interaction of our framework with a cluster and existing cloud platform architecture.

The analytical expressions that we detail and the increment in performance that is gained when using them in place of traditional equilibrium based performance measures is illustrated through several examples. In these examples we see that when a system is subject to regular predictable bursts our method can lead to a substantial reduction in losses. More modest improvements in performance are also seen when fluctuations in demand are unpredictable.

An important aspect of our framework is that the computations needed for its online implementation do not *necessarily* need to be performed online (although in many cases an online implementation would be possible); the manager only needs access to the decision corresponding to each potential system state. For a system with m servers, r buffers, and simple burst behaviour, obtaining these decision rules will require the inverse and matrix exponential of matrices with dimension $2(m+r+1)$ to be found. Using the basic form of Gauss–Jordan elimination, a $n \times n$ matrix inverse can be computed in $\mathrm{O}(n^3)$ steps.

Various algorithms, each with their individual strengths and weaknesses, are available to compute the matrix exponential (c.f. [140]): for a discussion of these issues see [141]. For large systems it may be more practical to store decisions (i.e. two $2(m+r+1)$-dimensional vectors which give the optimal server and buffer choices for each system state) and access them when needed.

7.1.1 Related work

One of the main triggers behind our work is the observation that much of the work on performance evaluation of cloud platforms is based on a description of the system in equilibrium. Our work is based on a *transient* description of the system, which is prudent since: i) the system may never reach equilibrium between changes in demand, and ii) a transient procedure allows the manager to take advantage of information on the present state of the system when making a provisioning decision.

In [96] Khazaei et al. propose an m server, queueing system with a capacity r buffer as an approximation to the type of real world distributed cloud computing platform system we are interested in. In their model compute tasks arrive to an m server system according to a homogeneous Poisson process, have a generally distributed service time, and are able to occupy r input buffer places if the system is already processing m tasks upon arrival, but are otherwise lost. Through an analysis based on the equilibrium state of the system the authors are able to determine the relationship between the number of servers and input buffer size and equilibrium performance indicators such as mean queue size, blocking probability, and the probability that a task will enter service immediately upon arrival. More recently Atmaca et al. [142] have provided a generalization of this work that uses Phase-type distributions to model service times and the time between arrivals. There are also generalizations of the model so that bursty (see e.g. [143]) and batchy (see e.g. [144, 145, 133]) behaviour can be investigated, again from an equilibrium perspective.

A related presentation is given in [146] where Tan and Xia model the distributed cloud from a revenue management perspective as a multi-class loss network with jobs of different types arriving according to a general renewal process. In [147] Bruneo gives a model based on stochastic reward nets that is scalable to very large system sizes and is flexible enough to be adapted to different scenarios — similar performance metrics are again analysed from an equilibrium point of view.

In [148] Maccio and Down introduce a model that has a single server which switches between on and off states according to the length of the queue. Again using steady state analysis, several similar performance metrics are connected to the system parameters and some key observations on how the system behaves are made.

A notable exception to the dominant equilibrium analysis is a discrete time model predictive control based approach given by Zhang et al. in [126]. The empirical approach taken in their work is presented as a promising initial step towards provisioning cloud computing platforms and represents an approach that is methodologically complementary to the one we present. Also of note is the recent work of van Leeuwaarden et al. in [149], where a cloud provisioning algorithm based on heavy traffic approximations and asymptotic analysis is introduced and verified using simulation.

Although MAMs have previously been used to study cloud computing platforms (see e.g. [150, Chapter 21]), they have not yet been used to evaluate effective *transient* performance measures. A related modelling formalism that also shows promise for the effective transient analysis of cloud computing platforms is stochastic Petri nets. A key piece of work investigating this avenue is [151], where a similar notion of transient and equilibrium

losses from blockages is investigated. The authors of this work do not incorporate system operating costs into their performance measure, and so the trade off between alternative capacity choices and blocking is not explicitly considered. Rather the focus is on determining the resiliency of the system to exogenous uncontrolled changes in capacity (for example due to system failure).

In the next section we will detail our model of a cloud computing platform, and then in the subsequent section we will develop an encompassing model that takes the system model as an input to allow our performance measure to be computed. This is followed by some examples that show how to use the performance measures to make short term provisioning decisions.

7.1.2 Organisation

The remainder of this chapter is structured as follows. In Section 7.2 we give a formal description of our illustrative model for a cloud computing platform. Section 7.3 develops a framework around this model that allows us to present a method of performance evaluation in Section 7.4. In Section 7.5 we illustrate our method. We provide an outlook to future research in Section 7.6.

7.2 Model of cloud computing platforms

In this section we introduce a model of cloud computing platforms that reflects the features we discussed in the introduction and which we will later use in the development of our novel encompassing performance evaluation model. We have been careful to be very clear about the assumptions of the model which we use to illustrate the framework by detailing the features that we *do* include, while simultaneously emphasizing the scope of applicability by providing some detail on the features that we *do not* include.

We assume that each arrival to the system consists of at most ℓ tasks, each of which requires its own server (e.g. a CPU core) to be processed or unit of buffer to be held in. Furthermore, we assume that the system exists in a random environment where traffic usually arrives according to some 'normal' rate, but occasionally arrives at some other 'bursty' rate. Throughout this chapter we use the term *rate* in the following sense: when an event occurs at rate λ at time t we take that to mean that the probability of the event occurring during $[t, t+h]$ is $\lambda h + \mathrm{o}(h)$ where $\mathrm{o}(h)/h \to 0$ as $h \to 0$, i.e. $\mathrm{o}(h)$ converges to zero faster than a linear function of h. Let $Y(t) \in \{1, 2\}$ equal 1 when arrivals are occurring at the normal rate at time t and equal 2 when arrivals are occurring at a bursty rate at time t. Moreover, when Y is in state 1 it transitions to state 2 at rate α, and when Y is in state 2 it transitions to state 1 at rate β. Let the number of tasks that a job brings to the system be governed by the random variable K, we assume that K has finite support. We denote the arrival rate of jobs consisting of $k \in \{1, \ldots, \ell\}$ tasks when $Y(\tau) = y$ at time $\tau > 0$ by $\lambda_y^{(k)}$. Note that the transitions of Y govern the frequency and duration of unpredictable bursty periods. It is a simple matter to extend our model to have multiple burst types of different frequency and duration.

Let $X(t) \in \{0, 1, \ldots, m, m+1, \ldots, m+r\}$ be the number of tasks being processed at time t by a cloud platform with the preceding arrival process, m servers, and a buffer of size r. When $X(t) \leq m$, then all of the tasks in the system are being served, however when $X(t) > m$, then m tasks are being served and $X(t) - m$ are waiting in the buffer. We assume that tasks require an exponentially distributed service time with mean μ_s^{-1}.

Therefore, the set of parameters $(\lambda_y^{(k)}, y \in \{1,2\}, k \in \{1,\ldots,\ell\})$ and μ_s^{-1} together allow for the distribution of job processing times to be Erlang distributed with a random shape parameter K and rate parameter μ_s^{-1}.

If a job consisting of k tasks arrives to the system when there are fewer than k servers or buffer units available, (i.e. $X(t)$ greater than $m+r-k$), then the job is blocked from entry and lost. Furthermore, we also incorporate impatience into our model. When a task is in the buffer, it will wait up to an exponentially distributed amount of time with mean μ_a^{-1} for service to commence, but will otherwise abandon the system without being served. The states and transitions of this finite state Markov process are illustrated in Figure 7.2 for the case when each arrival to the system always consists of only a single task. For clarity, we also compactly summarize the full set of transitions in Table 7.1.

At this point we reiterate that the performance evaluation framework we develop in the next section can also be applied to many models other than the one we have detailed in this section. For example, the servers in a cloud may not necessarily be homogeneous (as we have assumed). So long as the simple task-based workload that we assume applies to the system under consideration, the processing rates of each server can be modified through the entries in Table 7.1. To see this, consider the case that a system possesses a fixed amount of processing power that is shared between the tasks being processed, then $x\mu_s$ would be replaced by μ_s/x in the table.

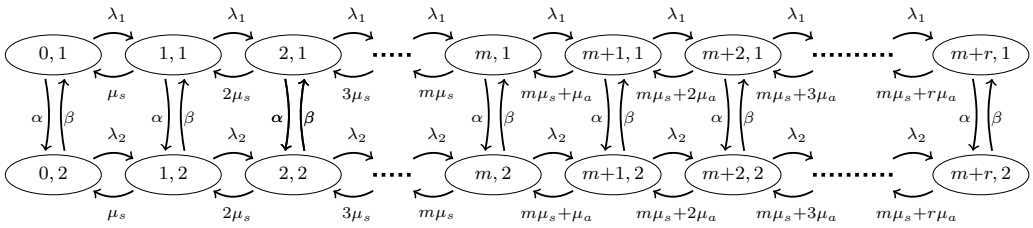


Figure 7.2: State transition diagram for a distributed computing platform with time homogeneous unpredictable arrivals, m servers, r units of buffer, and arrivals always consisting of a single task ($\ell = 1$).

Table 7.1: Transition rates of our bursty batch cloud computing platform.

Transition	Rate at time τ	States
$(x,\,1) \to (x,\,2)$	α	$0 \le x \le m+r$
$(x,\,2) \to (x,\,1)$	β	$0 \le x \le m+r$
$(x,\,1) \to (x+k,\,1)$	$\lambda_1^{(k)}$	$0 \le x \le m+r-k,\ 1 \le k \le \ell$
$(x,\,2) \to (x+k,\,2)$	$\lambda_2^{(k)}$	$0 \le x \le m+r-k,\ 1 \le k \le \ell$
$(x,\,y) \to (x-1,\,y)$	$x\,\mu_s$	$0 < x \le m,\ y \in \{1,2\}$
$(x,\,y) \to (x-1,\,y)$	$m\,\mu_s + (x-m)\,\mu_a$	$m < x \le m+r,\ y \in \{1,2\}$

7.3 Encompassing performance evaluation model

Now that we have a general model of how the system operates, we must define some additional stochastic processes that allow us to perform transient performance evaluation. To see why it is necessary to utilize these additional processes, observe that $X(t)$ does not provide any information on blocked and abandoned tasks during $[0,\, t]$ — we must develop an encompassing model that also records these losses. As Chiera et al. do in [115] and [116], we assume that each blocked task costs the manager of the system some pre-determined amount $\theta_b > 0$. Similarly, each task that abandons the system without being served incurs a cost of θ_a. Furthermore, we suppose that each active server results in a cost of θ_s per unit of time, and similarly each active unit of buffer results in a cost of θ_u per unit of time. Specifically, over any time interval $[t_1,\, t_2]$ the system manager incurs a deterministic cost of $(m\theta_s + r\theta_u)\,(t_2 - t_1)$ to maintain m servers and r units of buffer. Similarly, when X is in a state x such that $x > m + r - \ell$ during a time interval $[t_1,\, t_2]$ and Y is in state y, a loss of

$$(t_2 - t_1)\,\theta_b \sum_{k=m+r-x+1}^{\ell} k\,\lambda_y^{(k)}$$

is expected to be incurred from blocked tasks.

Table 7.2 summarizes the notation used for different types of losses for the reader's convenience.

Table 7.2: Cost parameters.

Parameter	Definition
θ_b	Lost revenue from blocked task.
θ_a	Lost revenue from abandoning task.
θ_s	Server cost per unit time.
θ_u	Buffer cost per unit time.

Let $R_{x,y}^{m,r}(t)$ denote the revenue lost during $[0,\, t]$ from blocked tasks (i.e. tasks that attempt to enter the system when it is at capacity) when $(X(0), Y(0)) = (x, y)$ and during $[0,\, t]$ there are m servers available with r units of buffer. Similarly, let $A_{x,y}^{m,r}(t)$ denote the revenue lost during $[0,\, t]$ from abandonments (i.e. tasks that leave the buffer due to impatience) and let $M_{x,y}^{m,r}(t)$ denote the cost of operating the system for t time units, each with capacity consisting of m servers and r units of buffer, and initial condition $(x,\, y)$. Therefore the expected revenue loss during $[0,\, t]$, under the specified initial condition $(x,\, y)$ and system size $(m,\, r)$, can be written as

$$g_{x,y}^{m,r}(t) := \mathbb{E}[R_{x,y}^{m,r}(t)] + \mathbb{E}[A_{x,y}^{m,r}(t)] + \mathbb{E}[M_{x,y}^{m,r}(t)]\,. \tag{7.2}$$

The function $g_{x,y}^{m,r}$ is reminiscent of the capacity value function of [116] and [115], and so we will also refer to it by that name. For a system with unit arrivals, no bursts in the arrival process, no buffer, no abandonments, and no capacity or buffer costs $g_{x,y}^{m,r}$ reduces to the original capacity value function formulation. Our novel augmentation of the original capacity value function with the additional operating cost term $\mathbb{E}[M_{x,y}^{m,r}(t)]$ is

necessary for the method to be implemented as a provisioning framework. Despite this, the key technical challenge in the evaluation of $g^{m,r}_{x,y}(t)$ lies in the computation of $\mathbb{E}[R^{m,r}_{x,y}(t)]$ and $\mathbb{E}[A^{m,r}_{x,y}(t)]$. The foremost reason that evaluation of $\mathbb{E}[M^{m,r}_{x,y}(t)]$ does not present a technical challenge is that we assume system operating costs depend deterministically on m and r. Within this assumption, a variety of cost formulations are possible. For example, economies of scale may exist in the provisioning of servers. So that issues such as these do not distract from our primary technical contribution (which is the determination of $\mathbb{E}[R^{m,r}_{x,y}(t)] + \mathbb{E}[A^{m,r}_{x,y}(t)]$) we assume that the expected system operating costs accrue according to the simple linear function

$$\mathbb{E}[M^{m,r}_{x,y}(t)] = (m\theta_s + r\theta_u)\, t\,,$$

independently of $X(0)$ and $Y(0)$. In the case that operating costs were to vary deterministically with time, this function could be modified accordingly.

It is instructive to write an integral expression for $R^{m,r}_{x,y}(t)$ so that some intuition for our performance evaluation model can be obtained, with $\mathrm{I}\{B\}$ the indicator function for event B, as follows:

$$\int_0^t \theta_b \left(\sum_{y=1}^{2} \sum_{s=0}^{\ell-1} \sum_{j=s+1}^{\ell} \lambda^{(j)}_y j \mathrm{I}\{X(\tau) = m + r - s\} \mathrm{I}\{Y(\tau) = y\} \right) \mathrm{d}\tau\,.$$

This random variable can be understood as the value of an accumulation of the arrivals from underlying Poisson processes that are switched 'on' and 'off' as needed by the pair of binary valued random processes that indicate when X and Y are in states that result in blockages. The term $\mathbb{E}[R^{m,r}_{x,y}(t)]$ is the expected value of this integral conditioned on $X(0) = x$ and $Y(0) = y$. Soon we will see that MAMs provide a powerful and convenient avenue to evaluation of this expression. Since abandonment losses result from transitions of (X, Y) rather than holding times of (X, Y), it is not straightforward to write a similar expression for $\mathbb{E}[A^{m,r}_{x,y}(t)]$; nonetheless MAMs may still be used for its evaluation.

The value over the planning horizon of $[0,\, t]$ of a change in m and r to $\widetilde{m}$ and $\widetilde{r}$ is

$$g^{\widetilde{m},\widetilde{r}}_{x,y}(t) - g^{m,r}_{x,y}(t)\,. \tag{7.3}$$

This expression is the basis of our transient provisioning framework. By choosing values of $\widetilde{m}$ and $\widetilde{r}$ that maximize revenue during the chosen planning horizon, the system manager is able to improve performance. Server and buffer space will only be active if it is expected to generate more revenue than the operating costs of having it active. Importantly, finding these optimal values is a simple numerical procedure that only needs to be performed once for any given set of parameters. Although not necessarily required, for large systems it may be sensible that the decision rules be stored in memory that is fast to access.

We will now outline a novel method (that generalizes results in [131], [116], and [115]) for obtaining explicit values of (7.2) (and therefore (7.3)). A key observation of this chapter is that the process $(X(t), Y(t) : t \geq 0)$, or simply $(X,\, Y)$, which gives the current number of tasks held by the system and the current mode of arrivals, can be viewed as the background process of a pair of batch Markovian arrival processes (BMAPs). The first of the BMAPs records the number of tasks which are blocked from entry to the system during $[0,\, t]$ due to capacity constraints, while the second BMAP records the number of abandonments during $[0,\, t]$. We will refer to these as the 'blocking BMAP' and 'abandonment BMAP' respectively. The rest of this section aims to show how viewing

the system in this way allows MAMs to be exploited so that the explicit computation of $\mathbb{E}[R^{m,r}_{x,y}(t)]$ and $\mathbb{E}[A^{m,r}_{x,y}(t)]$ can be performed, which in turn allows (7.2) and (7.3) to be computed.

A MAP is a counting process with arrivals of different types governed by the transitions and holding times of another finite state Markov chain (see e.g. [118] and [119]). Generally for MAPs each arrival is indexed by an element from a set $\mathcal{C}$. For our blocking MAP we will denote the elements of this index set by the numbers 1 to ℓ as follows $\mathcal{C}_b = \{1, \ldots, \ell\}$. This notation follows from the fact that a type k arrival results in k lost tasks. For our abandonment MAP the index set consists of a singleton representing an abandonment type arrival, that is $\mathcal{C}_a = \{a\}$. Letting $N^{m,r}_k(t)$ be the number of type k arrivals during $[0, t]$ when there are m servers and r units of buffer, it is clear from the linearity property of expectation that

$$\mathbb{E}[R^{m,r}_{x,y}(t)] = \theta_b \sum_{k=1}^{\ell} k\, \mathbb{E}[N^{m,r}_k(t) \,|\, X(0) = x, Y(0) = y] \tag{7.4}$$

and, letting $N^{m,r}_a(t)$ be the number of abandonments during $[0, t]$ when there are m servers and r units of buffer, we similarly have

$$\mathbb{E}[A^{m,r}_{x,y}(t)] = \theta_a\, \mathbb{E}[N^{m,r}_a(t) \,|\, X(0) = x, Y(0) = y]\,. \tag{7.5}$$

Hence our focus is on determining these values.

Aside from the set $\mathcal{C}$, MAPs, as described earlier in Chapter 5, are parametrised by a sequence of matrices $(D_0, D_h, h \in \mathcal{C})$ with the properties:

(i) the matrices $(D_h, h \in \mathcal{C})$ are non-negative;

(ii) the matrix D_0 has negative diagonal elements and non-negative off diagonal elements;

(iii) the matrix D_0 is nonsingular; and

(iv) the matrix $D = D_0 + \sum_{h\in\mathcal{C}} D_h$ is an irreducible infinitesimal generator.

The matrix D governs the transition rates of a background Markov process, while the matrices $(D_h, h \in \mathcal{C})$ specify the arrivals that are associated with relevant holding times and transitions of the background process. The matrix D_0 specifies the transitions which do not have arrivals associated with them and can be calculated from property (iv). To obtain our transient performance measures we must specify particular parametrisations of these matrices.

In our case (X, Y) is the background process which governs the arrivals of blocked and abandoned tasks for each of our MAPs. We encode the transitions of this background process, as given by Table 7.1 and illustrated for the case where arrivals only ever bring a single task in Figure 7.2, in the $2(m + r + 1)$ dimensional matrix D with states arranged as follows:

$$(0,1),\ (1,1),\ (2,1),\ \ldots,\ (m+r,1),\ (0,2),\ (1,2),\ \ldots,\ (m+r,2)\,.$$

For our blocking MAP, when (X, Y) is in a state (x, y) with $x \geq m+r+2-h$ arrivals of types $k \in \{1, \ldots, h\}$ occur according to Poisson processes with rates $\lambda^{(k)}_y$. In order to record these losses we define for $y \in \{1, 2\}$ the $(m + r + 1) \times (m + r + 1)$ square matrices

$D_{y,k}$ for $k \in \{1, \ldots, \ell\}$, which for diagonal elements (i,i) with $i \geq m + r + 2 - k$ have entries $\lambda_y^{(k)}$ and 0 otherwise. For example, if $m = 2$, $r = 1$, and $\ell = 2$ then we require

$$D_{y,1} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_y^{(1)} \end{pmatrix}, \quad D_{y,2} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \lambda_y^{(2)} & 0 \\ 0 & 0 & 0 & \lambda_y^{(2)} \end{pmatrix},$$

for $y = 1, 2$.

Combining the bursty and normal arrivals (blocking losses) together we have the matrices

$$D_k = \begin{pmatrix} D_{1,k} & 0 \\ 0 & D_{2,k} \end{pmatrix}, \quad k = 1, \ldots, \ell\,. \tag{7.6}$$

Recalling that $D_0 = D - \sum_{k=1}^{\ell} D_k$, based on this parametrisation we are able to write down an infinite dimensional block matrix that is an infinitesimal generator of a process which describes the state of the cloud computing platform model, introduced in Section 7.2, augmented by a state that counts cumulative lost tasks (i.e. $\sum_{k=1}^{\ell} k\, N_k$), as follows:

$$Q = \begin{pmatrix} D_0 & D_1 & D_2 & \cdots & D_\ell & & & \\ & D_0 & D_1 & \cdots & & D_\ell & & \\ & & D_0 & \cdots & & & D_\ell & \\ & & & \ddots & \ddots & \ddots & \ddots & \ddots \\ & & & & \ddots & \ddots & \ddots & \ddots \end{pmatrix}.$$

In this block matrix each row of matrices corresponds to a different total number of blocked tasks during $[0, t]$, or the 'level' of the overall blocking MAP. The first row corresponds to no blockages, while the second row corresponds to a single blockage, and so on for the further rows. Within each row the block D_0 corresponds to movements of the background or 'phase' process (X, Y), which are the transitions that are not associated with any blockages (i.e. of the system model). In our case this is services, arrivals (when not at capacity), and changes between bursty and normal arrival behaviour.

Similarly, for our abandonment MAP a transition of (X, Y) from a state (x, y) to $(x-1, y)$ with $x \geq m+1$ results in an arrival of type a (abandonment loss) with probability

$$(x - m)\mu_a \left[m\,\mu_s + (x - m)\mu_a \right]^{-1}, \tag{7.7}$$

and otherwise a service has occurred. To see that this is the case, observe that when the process is in state (x, y) with $x \geq m+1$, this implies that there must be m tasks undergoing processing and $x - m$ tasks waiting in the buffer. Each of the tasks undergoing processing has an independent exponentially distributed amount of time with mean μ_s^{-1} until it completes, and each of the tasks waiting in the buffer has an independent exponentially distributed amount of time with mean μ_a^{-1} until it abandons. Equation (7.7) then follows from the properties of the exponential distribution.

To obtain an infinite dimensional block matrix that is an infinitesimal generator for N_a we place the departures that correspond to an abandonment in the matrix D_a. This $2(m+r+1)$ dimensional matrix has entries $i\,\mu_a$ for $i = m, m+1, \ldots, m+r, 2(m+1), 2(m+$

$2), \ldots, 2(m+r)$ at coordinates $(i+1,\, i+2)$. Now, using $D_0' = D - D_a$, we have that

$$Q_a = \begin{pmatrix} D_0' & D_a & & & \\ & D_0' & D_a & & \\ & & \ddots & \ddots & \\ & & & \ddots & \end{pmatrix}.$$

Similar to the case for Q_k, in the matrix Q_a each row of matrices corresponds to a different total number of abandonments during $[0,\, t]$, or the level of the overall abandonment MAP. Since only a single abandonment can occur at a time, each row can be parametrised using only D_0' and D_a.

We have now completely defined our model of a distributed cloud computing platform and the encompassing machinery that we will use to perform a transient performance evaluation.

7.4 Transient performance evaluation

The simplest application of MAMs is to find the equilibrium distribution of a finite state Markov process. Given that the process has infinitesimal generator Q, if we denote the equilibrium distribution by (row vector) $\boldsymbol{\pi}$ and a vector of ones with the same dimension as $\boldsymbol{\pi}$ by $\mathbf{1}$ then this simply amounts to solving the equation $\boldsymbol{\pi} Q = 0$ subject to $\boldsymbol{\pi}\mathbf{1} = 1$ where each component of $\boldsymbol{\pi}$ must be non-negative. This computation is straightforward on modern computers for most finite state space Markov processes of interest. For example, this simple computation provides an alternative method for obtaining the probability given by (7.1). The classical application of MAMs is to find the matrix-geometric stationary distribution of a $GI/M/1$-type Markov chain, which is an infinite-state process.

Generally speaking, the field of MAMs, and more broadly algorithmic probability, is concerned with augmenting computational methods, such as the one just discussed, with analytical results. Through this it is often possible to answer questions that are computationally difficult in the absence of analysis and analytically infeasible in the absence of computational resources. The development of the framework that we are presenting in this chapter falls exactly into this category of methodology. By utilizing the special structure of MAPs, usual Markov process theory, and our formulation of the problem, we are able to arrive at expressions that can be evaluated numerically to answer the challenging questions faced by managers of distributed cloud platforms. In the next subsection we will derive the expected value of lost revenue during $[0,\, t]$ conditional on particular values of $(X(0),\, Y(0))$ for the unpredictable case, where the rates of the process are time homogeneous, before incorporating predictable behaviour in the subsequent subsection.

7.4.1 Unpredictable arrival rate expected value

Using $(D_k : 1 \leq k \leq \ell)$ and D_a we have shown how to construct a pair of two dimensional Markov processes, $R^{m,r}_{x,y}$ and $A^{m,r}_{x,y}$ where the second dimension gives the value of lost tasks from blockages and abandonments respectively. Using standard MAMs we show how to find the expected value of these processes at a finite time t in the form of an analytical expression that can be evaluated numerically. Our novel formulation of the encompassing performance evaluation model allows for systems of the type detailed in Section 7.2 to be

evaluated using Theorem 2.3.2 in [118].

Proposition 33 (given below) summarizes the application of [118, Theorem 2.3.2] to our illustrative model. The proposition gives the expected lost revenue for finite time horizons as a function of different choices of m and r, the current number of tasks in the system x, and the current state of the unpredictable process y. Through straightforward optimization of this function in terms of m and r at a chosen planning interval T a cloud platform manager can dimension their system at the time points $0, T, 2T, \ldots$ to obtain performance improvements. We again highlight that this optimization need not *necessarily* be performed online, the manager only needs access to the mapping $(x,\, y) \to (m,\, r)$ for each $(x,\, y)$ of interest up to some maximal capacity value.

Proposition 33. *For constant $(D_k : 1 \leq k \leq \ell)$, D_a, and D the capacity value function can be evaluated as*

$$g^{m,r}_{x,y}(t) = (m\theta_s + r\theta_u + \boldsymbol{\pi} D^* \mathbf{1})\, t - \boldsymbol{\pi}_0 \big(\exp(D\, t) - I \big)\, D^- D^* \mathbf{1}\,, \tag{7.8}$$

where

$$D^* = \theta_a\, D_a + \theta_b \sum_{j=1}^{\ell} j\, D_j\,, \quad D^- = (\mathbf{1}\boldsymbol{\pi} - D)^{-1}\,,$$

$\mathbf{1}$ is a $2(m+r+1)$ column vector of ones, $\boldsymbol{\pi}$ is the equilibrium distribution of the Markov process $(X,\, Y)$, $\boldsymbol{\pi}_0$ is a vector indicating that the process $(X,\, Y)$ starts in state $(x,\, y)$, and I is an identity matrix of appropriate dimension.

The matrix D^- is sometimes called the fundamental or deviation matrix and, to the author's knowledge, was first introduced in [152]. This matrix is closely related to another matrix via the relationship

$$\int_0^\infty (\boldsymbol{\pi}_0 \exp(D\, t) - \mathbf{1}\boldsymbol{\pi}) \mathrm{d}t = D^- \mathbf{1}\boldsymbol{\pi}$$

where the matrix on the left hand side also commonly goes by the name deviation matrix (see e.g., [153]). A well developed theory has been developed surrounding these matrices, as detailed in the references just given.

7.4.2 Incorporating predictable bursts

If the rate at which tasks arrive to a system is constant over any particular planning horizon but varies between planning intervals, then a manager can use the framework developed in the previous subsection tailored to each interval. The purpose of this subsection is to show how our framework can be adjusted when the rate at which tasks arrive to the system is known (or believed) to change at predictable time instances *within* a particular planning horizon $[0, t]$. Specifically, we adapt the framework we presented in the previous subsection to the case of a predictably time-varying arrival rate when arrival rate changes occur only at a countable number of distinct time points within a particular planning horizon.

Suppose we have h distinct time periods, defined by their end points $0 < t_1 < t_2 < \cdots < t_h := t$, which form a partition of $[0, t]$, each having its own arrival rate. Denote by $D_{(i)}$, $D^*_{(i)}$, and $D^-_{(i)}$ the corresponding parametrisation in time period i. Then, using the standard expression for the transient distribution of a time-homogeneous finite state

Markov process for each interval (for details see e.g. [12, p. 259]), we may compute the capacity value function as

$$g_{x,y}^{m,r}(t) = (m\theta_s + r\theta_u)\, t + \sum_{i=1}^{h} \boldsymbol{\pi}_{t_{i-1}} D_{(i-1)}^{*} \mathbf{1}(t_i - t_{i-1}) \\ - \sum_{i=1}^{h} \boldsymbol{\pi}_{0,t_{i-1}} \Big(\exp\left(D_{(i-1)}\left(t_i - t_{i-1}\right)\right) - I \Big) D_{(i-1)}^{-} D_{(i-1)}^{*} \mathbf{1}\,, \quad (7.9)$$

where $t_0 = 0$,

$$\boldsymbol{\pi}_{0,t_i} := \boldsymbol{\pi}_0 \prod_{j=1}^{i} \exp\left(D_{(j-1)}(t_j - t_{j-1})\right),$$

and $\boldsymbol{\pi}_{t_i}$ is the equilibrium distribution of the Markov process with infinitesimal generator $D_{(i)}$.

This expression may be used in the same way as Proposition 33. In this case the mapping $(x, y) \to (m, r)$ will also be dependent on the predictable bursts within the planning horizon.

7.5 Illustrations

This section presents four illustrative applications of our method to models of distributed cloud computing platforms. Each subsection illustrates a different aspect of the method. We will first apply the method to the simplest possible setting, that of a system with only predictable and time homogeneous arrivals. We illustrate the connection between our transient performance measure and transient decision making, as well as demonstrate that this may provide modest improvements compared to decisions based on equilibrium methods. Subsequently, we allow arrivals to vary with time in a predictable manner and show that in this case the improvement over equilibrium methods may be quite substantial. We then investigate unpredictable arrivals, where we again see that transient decision making can outperform equilibrium decision making, even when the decision maker is unaware of the unpredictable nature of the arrival process. Finally, we investigate the effect of batch arrivals on system performance, and again see our policy may provide improvements.

In our case the traffic intensity varies according to the value of m that is selected by our provisioning framework. Hence in this section we will fix the arrival process and study the resulting changes in m and r. Similarly, the mean service time μ_s will be absorbed into the choice of m and r in the same fashion. In the final example, however, the coefficient of variation is varied through the distribution of batch size, when arrivals of this type are considered. We will evaluate the performance of our framework subject to an increasing coefficient of variation.

We note that for a cloud computing platform to be sustainable the values of the cost parameters θ_b and θ_s (see Table 7.2) must be such that the expected cost of keeping a server active for the duration of a job is lower than the revenue that would be lost from a blocked job, and similarly for θ_a and θ_u.

Later in this section we will give an example analysis of batchy and bursty behavior through specific parameter choices. These examples will suggest that, with our parameter choices, accounting for batchiness of the arrival process may be more important than

accounting for burstiness.

7.5.1 Simple time homogeneous system

In this subsection we will explore the simplest case of our model using Figure 7.3 and Figure 7.4. Assume that tasks of unit size arrive to a cloud computing platform according to a Poisson process with rate 80 (that is, we expect 80 tasks to arrive per time unit), tasks take an exponential amount of time with mean 1/2 to be processed, and will wait in a buffer for up to an exponential time with mean 1 to begin processing before abandoning the system. Blockages and abandonments incur losses to the manager of sizes 0.5 and 0.55 respectively. Units of server cost 0.5 and units of buffer cost 0.1 to operate per unit of time. Since there are no unpredictable changes in the arrival rate we may set $\lambda_1^{(1)} = \lambda_2^{(1)}$ and arbitrarily take $\alpha = \beta = 1$.

The first term in (7.8) gives the equilibrium rate of loss from this system given a choice of m and r. Minimizing over this term in (m, r) is equivalent (or superior) to many of the equilibrium performance measures considered by the papers discussed in the introduction. In Figure 7.3 we show this function for different buffer and server choices. It is evident that low (< 30) or high (> 50) values of m result in a greater rate of loss (lighter shading) for a wide range of values of r. For $m = 40$ there is a low rate of loss (dark shading) for a wide range of values of r; as r increases the loss rate experience a mild decrease. In fact, if we were to only consider the equilibrium of the system in our decision making process, then we would choose to have 40 servers and 8 units of buffer; although m and r combinations near to this point experience a similar rate of loss.

In the first panel of Figure 7.4 we display the function (7.8) of our system with $r = 15$, lower initial tasks in the system $x = 10$ (black lines), higher initial tasks in the system $x = 50$ (grey lines), lower number of servers $m = 35$ (solid lines), and higher number of servers $m = 45$ (dashed lines). For the lower initial tasks in the system it can be seen that choosing the lower number of servers is expected to reduce the loss incurred by the system's manager. On the other hand, the higher number of initial tasks has reduced losses when there are more servers utilized. This illustrates the relationship between expected losses, server and buffer choice, and the current number of tasks in the system.

Using the relationship illustrated in the first panel of Figure 7.4 we are able to choose

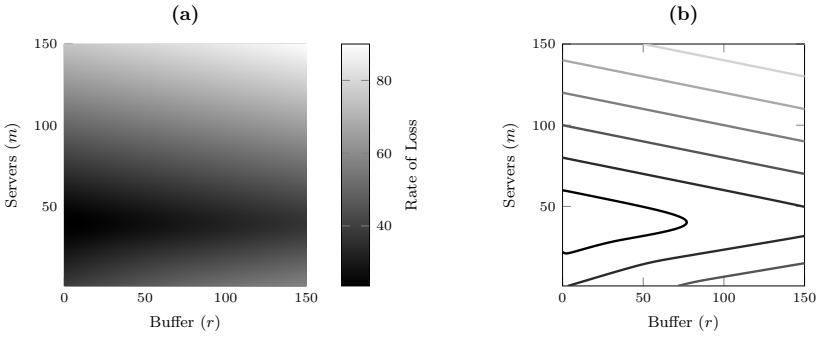


Figure 7.3: **(a)** Equilibrium rate of loss for different m and r combinations for equilibrium policy and **(b)** contour plot of equilibrium rate of loss for different m and r combinations for equilibrium policy.

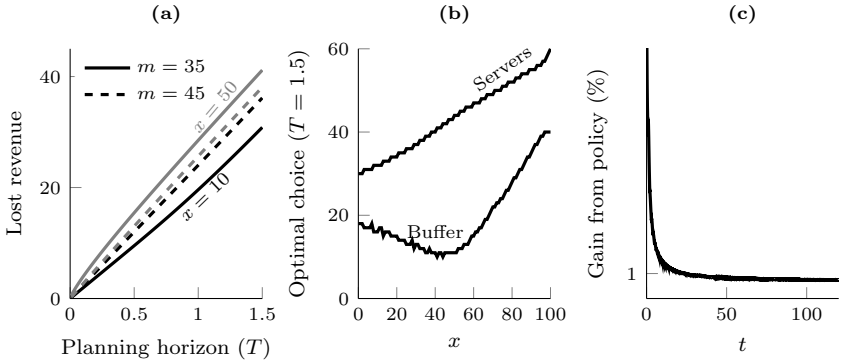


Figure 7.4: **(a)** expected losses during $[0, T]$ when $r = 15$ for different choices of m and initial condition x, **(b)** optimal buffer (m) and server (r) choices for different initial conditions x, and **(c)** estimated expected gain from adjusting according to the optimal choice in (b) every 1.5 time units in place of using the optimum given by (a).

m and r optimally for a planning horizon of 1.5 for each x, which we display in the third panel of the same figure. It can be seen that as there are more tasks in the system a higher number of servers is optimal. Interestingly, the optimal buffer size is convex in the number of tasks, with a minimum value at the equilibrium choice.

In the third panel of Figure 7.4, we are finally able to illustrate that the transient performance measure based framework developed in this chapter provides an improvement over equilibrium based frameworks. We compare 10^6 sample paths of the system operating using the equilibrium server and buffer choice with 10^6 sample paths where the server and buffer choice is adjusted each 1.5 time units according to the optimal choices displayed in the third panel. In this case our policy results in an improvement of approximately 1% compared to fixed server and buffer sizes chosen according to equilibrium system behaviour.

7.5.2 System with predictable bursts

This short subsection has the simple goal of highlighting that our framework may perform extremely well when a system is subject to predictable bursts in the arrival rate. Suppose that the system is the same as in the previous subsection, except that now tasks arrive at rate 60 during $\bigcup_{i\in\{0,2,4\}}[10i, 10(i+1))$ and at rate 80 otherwise, as illustrated in the lower panel of Figure 7.5. If the system manager was to make an equilibrium based server and buffer choice, then there are three obvious alternatives: i) provision according to $\lambda = 80$ to avoid SLA violations during bursts, ii) provision according to $\lambda = 60$ to avoid unnecessary operating costs during non burst periods, or iii) provision according to some mix of the previous two alternatives.

The solid curve in the upper panel of Figure 7.5 compares the lost revenue of a manager who uses the first of these alternatives (over-provision with $\lambda = 80$) with the lost revenue of a manager who uses our framework. This illustration suggests that the manager that adjusts server and buffer choices every 1.5 time units according to our transient provisioning framework reduces their losses by approximately 7.5% after 60 time units, compared to the manager who provisions statically according to an equilibrium performance measure.

The use of equilibrium performance measures does not, however, preclude the use of

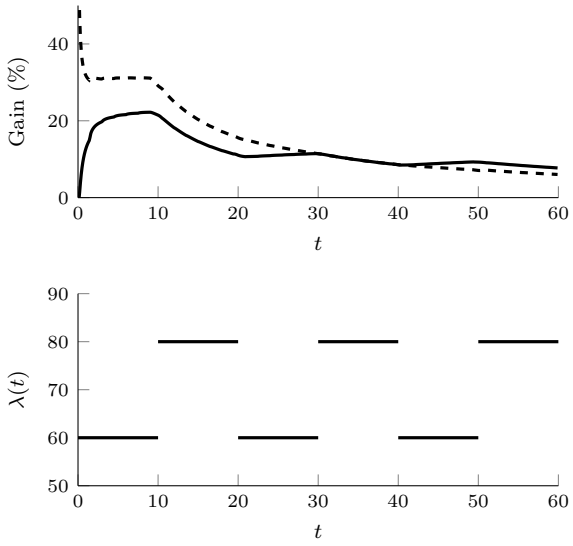


Figure 7.5: Lower panel: predictable bursts in the arrival rate. Upper panel: gain from using our transient policy in place of a static equilibrium policy (solid) and from using our policy in place of a dynamic equilibrium policy (dashed).

dynamic provisioning. The dashed curve in the upper panel of Figure 7.5 compares the lost revenue of a manager who provisions at any given time according to the equilibrium policy computed for the arrival rate of tasks at that time. In this case we still see a substantial gain from dynamically using a transient policy in place of dynamically using equilibrium policies. The strength of the transient policy likely primarily arises from its incorporation of present state information at each decision epoch (which is impossible using equilibrium methods).

7.5.3 Unpredictable time-varying system

In this subsection we demonstrate that our framework can improve system performance in the presence of unpredictable time varying behaviour. Furthermore, we investigate

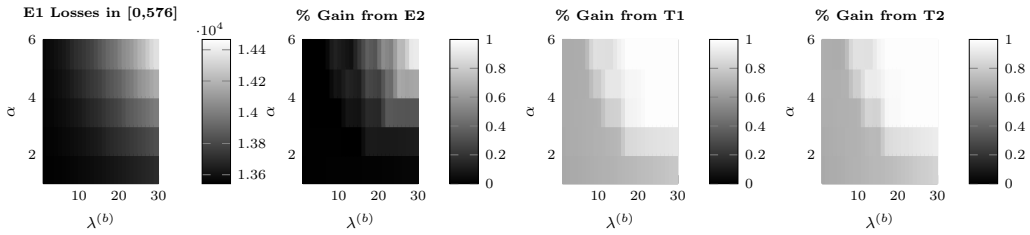


Figure 7.6: Comparison of losses in [0, 576] for an equilibrium policy that does not account for bursty behaviour (E1), an equilibrium policy that does account for bursty behaviour (E2) with our transient policy when bursty behaviour is not (T1) and is (T2) accounted for.

and compare the effects of the burst frequency and the burst magnitude on the expected lost revenue. The key message of the subsection, as illustrated by Figure 7.6, is that using our transient policy may provide an improvement on system performance, even if it is applied without knowledge that bursty behaviour is occurring. We will compare the policies E1, E2, T1, and T2, where E and T correspond to equilibrium and transient decision making respectively, and 1 and 2 correspond to bursty unaware and aware decision making respectively.

To measure burst frequency and magnitude we allow arrivals of unit size to occur according to a Markovian arrival process with two underlying states: the states correspond to a 'normal' demand regime and an 'increased' demand regime. In Figure 7.7 we depict these two states, their arrival rates, and the transition rates between them. When the system is in the increased demand regime it moves to the normal regime at rate 5. In this case each time the arrival rate increases due to a burst, the increase is expected to persist for 0.2 time units. On the other hand, when the system is in the normal regime it moves to the increased regime at rate α. A higher value of α therefore corresponds to more frequent bursts. We assume that a burst results in the arrival rate increasing by $\lambda_{(b)}$. Since we wish to analyse the effect of burst frequency and magnitude, in order to keep the overall time average arrival rate equal to 80 (as for the time homogeneous case already considered), we counterbalance the increased demand regime arrival rate with a decrease in the normal arrival rate of $\lambda_{(b)}\alpha/(\alpha+5)$. This follows from the fact that the ergodic distribution of the background process that alternates between normal and increased demand is $\big(5/(5+\alpha),\, \alpha/(5+\alpha)\big)$.

Increases in $\lambda_{(b)}$ result in a higher arrival rate during burst periods and a decreased arrival rate during the normal regime. Increases in α result in a smaller difference between normal and increased demand, but increase the frequency with which bursts occur.

Now that it is clear how we are measuring and evaluating burstiness, we return to Figure 7.6. In the first panel of this figure we consider a manager who uses the equilibrium policy from the homogeneous arrival subsection — that is, they do not account for the bursty behaviour at all. It can be seen that individual increases in α or $\lambda_{(b)}$ have a minor effect on losses, but that when both of these parameters increase together larger losses are incurred.

In the second panel of Figure 7.6 the manager is aware of the values of α and $\lambda_{(b)}$. Since individual increases in the burstiness parameters do not seem to affect losses substantially, we do not expect that a change in policy will be very beneficial. This is confirmed by the second panel of Figure 7.6. When the bursty parameters are high together however, we see that accounting for the presence of burstiness is beneficial, even from an equilibrium

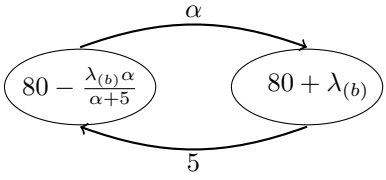


Figure 7.7: Two state background process. The left state represents the normal arrival rate while the right state represents bursty periods. The parameters α and $\lambda_{(b)}$ determine burst size and duration, however the average rate of arrivals is fixed at 80.

perspective.

The third panel shows that a gain of approximately 0.8–1% is generated through the usage of our transient policy, even when the burstiness of the system is not accounted for. Indeed, the fourth panel of the figure is highly similar to the third panel, indicating that in this example the policy performs equally well when the manager does not account for burstiness. Notably, our transient policy performs better than the equilibrium policy, even when the equilibrium policy is bursty aware and the transient policy is bursty unaware.

7.5.4 Batchiness

After having seen that our transient policy reduces losses in a bursty system, even when the manager is unaware of the bursty behaviour, we will now investigate if that is also true for a batchy system. In this case we will define a system to be more batchy when holding the expected number of arrivals in any time interval $[0, t]$ to be the same, the arrivals are able to come in batches of larger size.

To model this we fix a maximum batch size ℓ and let the arrival rate of batches of size k be $\lambda/(k\,\ell)$. From the basic properties of Poisson processes, we see that

$$\sum_{k=1}^{\ell} \mathbb{E}[N_k(t)] = \sum_{k=1}^{\ell} k\,\frac{\lambda t}{k\,\ell} = \lambda\, t\,,$$

so that an overall expected number of arrivals λt during $[0, t]$ is maintained. A higher value of ℓ is, however, clearly more batchy. Specifically, recalling that K is a random variable which governs the size of an arbitrary batch we have that

$$\mathbb{P}(K = k) = \frac{1}{k\,H_\ell}\,, \quad k \in \{1, 2, \ldots, \ell\}\,,$$

where $H_\ell = \sum_{k=1}^{\ell} k^{-1}$ is a harmonic number. Since it can be shown that the coefficient of variation of a job will be

$$\frac{\sqrt{\mathbb{E}[K] + \mathrm{Var}(K)}}{\mathbb{E}[K]}\,,$$

from this we can see that the coefficient of variation for an arbitrary job is

$$\sqrt{\frac{H_l\,(\ell + 2)}{2} - 1}\,.$$

This function is increasing in ℓ (as illustrated in the top panel of Figure 7.8), meaning that increased batchiness (i.e. higher ℓ) results in a higher coefficient of variation. In this example the coefficient of variation lies in the range of approximately 0.7071 to 7.9322.

For our ongoing illustrative parameters, an increase in batchiness (or variation) results in higher losses. Figure 7.8 focuses on the percentage reduction in losses that can be achieved compared with provisioning according to the equilibrium policy of the earlier homogeneous section (i.e. $m = 40$, $r = 8$) during the arbitrarily chosen time period $[0, 54]$. The figure shows that in this case the gains from accounting for batchiness may be substantially greater than the gains from accounting for burstiness. For lower levels of batchiness, $\ell \in \{1, \ldots, 10\}$ using an equilibrium batchy aware policy appears to provide similar gains to the transient batchy aware policy, and performs better than the transient batchy unaware policy. This contrasts with the bursty scenario where the transient policy

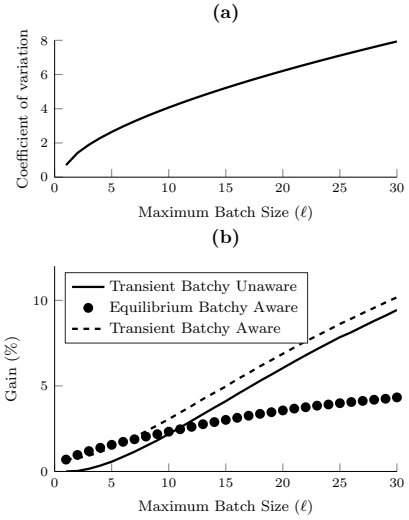


Figure 7.8: **(a)** Coefficient of variation of job size, **(b)** % reduction in lost revenue over [0, 54] from incorporating batch information into the equilibrium policy (dots), updating server and buffer sizes periodically (solid), or both incorporating batch information and using our transient policy (dashed) compared to a batchy unaware equilibrium policy over the period.

was superior even when bursty behaviour was not accounted for by the decision maker. For higher levels of batchiness the transient policies are each superior, yielding gains of approximately 10%, even if they are not aware that the system is batchy (with the batchy aware transient policy performing approximately 1% better than the batchy unaware transient policy).

7.6 Concluding remarks

The main contribution of this chapter is the observation that the capacity value function can be found using matrix analytic methods. Based on this observation, and upon incorporating operating costs for servers, we have presented a framework that may be used for transient provisioning and performance evaluation of cloud computing platforms. The purpose of the present chapter is to introduce a framework for the analysis of models which are applicable to the domain of transient provisioning of cloud computing platforms. As such we presented our framework in the context of a model that is comparable to those which are currently state of the art in the literature of this area (e.g. [142, 147, 144, 96, 145, 133, 148, 143]). The complexity of the illustrative model was chosen as a balance between presentation and relevance. We envisage that many possible extensions to the underlying model could be implemented within our encompassing performance evaluation framework with varying degrees of complexity.

Generalizing the system model to have more than two states in the underlying burst modulating process Y follows from simply augmenting the matrix D_k in (7.6) with an additional matrix $D_{y,k}$ for each additional state in Y. Such modifications result in a linear increase in complexity, the dimension of the matrices for which it is necessary to compute

exponentials and inverses of is the number of states of Y multiplied by $(m+r+1)$. In the same way, to model a system failure that does not result in removal of tasks but simply a pause in service, the matrix D_k could be augmented with a matrix representing a set of states with a reduced (or 0) service rate; which is again a linear increase in complexity. It would also be possible to model the case that such a failure removes all (or a random number) of tasks from the system through some careful adjustment of the rates in D; in this case the formulation is changed but the complexity is not affected. Furthermore, it would also be possible to apply our framework in the case of multiple resource types by taking the process X to be multidimensional (which is simply a notational increment in methodology).

The above additional applications of our framework rely on modifications of the structure of $g_{x,y}^{m,r}(t)$. Recall, however, that underlying our framework is the function

$$g_{x,y}^{\widetilde{m},\widetilde{r}}(t) - g_{x,y}^{m,r}(t)\,,$$

which could also be modified. For example, by adding a term that depends on the difference $m - \widetilde{m}$, costs related to the provisioning of new resources could be included.

Despite the flexibility of our approach, there do exist limitations to our method that remain a challenge. For example, we have assumed that abandonments occur on a task-wise basis, when in reality all of the tasks from a particular batch arrival may abandon together. Another limitation is that the model is inherently Markovian, so that we have not been able to incorporate generally distributed service times for tasks (e.g. to model heavy tailed behaviour). It is not obvious how to extend our framework to incorporate these features.

Another avenue of future research could be aimed at obtaining higher order moments of

$$R_{x,y}^{m,r}(t) + A_{x,y}^{m,r}(t) + M_{x,y}^{m,r}(t)\,.$$

Doing so would enable risk taking preferences to be incorporated into the decision making process that we have developed and allow more sophisticated performance evaluation.

CHAPTER 8

Functional form based optimisation for stochastic networks with blocking

8.1 Introduction

Many stochastic networks encountered in practice are affected by blocking, where network traffic is lost due to congestion. For instance, packets in a communication network may be dropped as a result of excessive delays, computing jobs lost because of insufficient capacity, and logistical processes disrupted due to low stock levels. A key decision when designing such networks is how to allocate resources to limit losses, while maintaining low costs.

For instance, consider a supply chain network designed for maintaining capital goods such as aircraft or MRI machines. Downtime of these goods is expensive and disruptive, and should be minimized to the extent possible. So when a capital good breaks down, the necessary spare parts are quickly dispatched from the nearest stock point to its location. When unavailable at the nearest stock point, the parts must be sent from a more distant location, increasing the downtime. Blocking in this case occurs when a spare part is unavailable at the closest stock point, and stochasticity arises from the uncertain lifetime of capital goods. In 2003 sales and services related to spare parts accounted for 8% of the gross domestic product in the United States [154], and consequently significant research attention has been devoted to the question of how to design the spare parts network in order to balance swift resp onse to breakdowns with the costs of maintaining a large supply of spare parts, see, e.g., [155, 156, 97]. A similar problem appears in the context of emergency services, where ambulances and fire trucks should be carefully located in order to ensure a fast response to emergencies [157, 158, 98].

A second example of stochastic networks with blocking is found in cellular wireless networks, where calls have to be handed over between cell towers as the user moves around, but may be dropped if the receiving tower has insufficient capacity available. Naturally, call dropping is to be avoided, and capacity management in cellular networks has been widely studied [159, 160, 161]. A similar issue plays a role in the recent development of mobile cloud computing, where computational jobs are served at small 'cloudlets' close to the user, rather than at a remote cloud computing facility [162]. While this reduces

delay compared to traditional cloud computing, each cloudlet only has limited capacity, and migrating jobs between cloudlets due to user mobility may lead to performance issues and blocking.

In the context of the examples just given the problem under consideration is often summarised as an optimisation problem,

$$\min_{\boldsymbol{c}\in\mathcal{C}} \; f(\boldsymbol{c})\,, \tag{8.1}$$

where $f(\boldsymbol{c}) := \mathbb{E}F(\boldsymbol{c})$ is a function $\mathcal{C} \to \mathbb{R}$ from a set of parameters to be optimised over to the real numbers, that represents a performance metric of interest. Here $F(\boldsymbol{c})$ is a random variable that follows a different law depending on the parameter $\boldsymbol{c}$. Let $\boldsymbol{c}^\star$ denote a solution to this optimisation problem. In many circumstances of practical interest this problem is intractable for the real-world system under consideration, yet it is still hoped to find a parameter choice $\widehat{\boldsymbol{c}}^\star$ which ensures $f(\widehat{\boldsymbol{c}}^\star)$ is as close to $f(\boldsymbol{c}^\star)$ as possible. Roughly speaking, there are two categories of approaches for solving capacity management problems in stochastic networks: analytical and simulation-based optimisation.

Analytical approaches typically take the complex, intractable stochastic network and optimisation problem of interest (8.1), and approximate it with a simpler network for which a related tractable optimisation problem,

$$\min_{\boldsymbol{c}\in\mathcal{C}} \; \widetilde{f}(\boldsymbol{c})\,, \tag{8.2}$$

can be formulated. For instance by approximating it with a product-form network [163, 95, 164], by assuming that the components of the network behave independently [165, 166], by looking at a fluid scaling [167], or by looking at large-scale systems [168, 169]. This closed-form approximation can then be used for solving the capacity management problem, either analytically [9, 170], numerically, or using a combination of both [114].

The second class of solution methods is simulation-based optimisation, where the capacity allocation is evaluated and updated using simulation of the stochastic network of interest. In such a set-up it is assumed that it is possible to obtain samples $\widehat{f}(\boldsymbol{c})$ of the random variables $F(\boldsymbol{c})$ evaluated at specific parameters $\boldsymbol{c}$, which are equal to $f(\boldsymbol{c})$ in expectation. These algorithms assume that it is possible to generate a sequence of samples $\big(\widehat{f}(\boldsymbol{c}^{(n)}),\, n \in \{1, 2, \ldots, N\}\big)$. Based on these samples, algorithms are developed which aim to solve the optimisation problem (8.1). In general the goals of this area of research are to provide rigorous performance guarantees that $\boldsymbol{c}^{(n)} \to \boldsymbol{c}^\star$ as $n \to \infty$ and to ensure this convergence occurs as fast as possible (i.e., $\boldsymbol{c}^{(n)}$ as close to $\boldsymbol{c}^\star$ as possible for N as small as possible).

The canonical simulation-based approach is stochastic approximation [171, 172], where the sequence of approximate optimisers is generated according to

$$\boldsymbol{c}^{(n+1)} = \boldsymbol{c}^{(n)} - \alpha^{(n)} H^{(n)} \nabla \widehat{f}(\boldsymbol{c}^{(n)})\,,$$

where $\alpha^{(n)}$ is a step-size or learning rate, $H^{(n)}$ is a linear map, and $\nabla\widehat{f}(\boldsymbol{c}^{(n)})$ is an estimate of the Jacobian $\nabla f(\boldsymbol{c}^{(n)})$. When $H^{(n)}$ is an identity matrix and $\alpha^{(n)} = \beta/n$ where $\beta \in \mathbb{R}_+$ is some positive real number, we recover the classical Robbins–Monro algorithm [173]. When $H^{(n)}$ is an identity matrix and $\nabla\widehat{f}(\boldsymbol{c}^{(n)})$ is estimated using finite differences, we recover the classical Kiefer-Wolfowitz algorithm [174]. In this case $\sum_{n=1}^{\infty} \alpha^{(n)} = \infty$ and some other conditions on the sequence of finite difference estimates must hold for convergence to be guaranteed. These algorithms are adaptations of gradient descent to

the stochastic setting. Adapting Newton and quasi-Newton type methods, where $H^{(n)}$ is taken to be the inverse of the Hessian matrix, to the stochastic setting, is currently a hot research topic [175].

Both analytical-approximation and simulation-based approaches have weaknesses that prevent straightforward application to complex stochastic networks. In the case of analytical approximations, it is often impossible to find an approximation for the network of interest that is sufficiently accurate and captures all the relevant features. As a result, any resource management decision made based on the approximate model may not work well in the original model. Simulation-based optimisation typically displays better accuracy, but suffers from large computational costs, which makes its application to high-dimensional stochastic networks infeasible.

Recently, the authors of [11] introduced a hybrid approach, that exploits theoretical knowledge of the stochastic network to significantly reduce computational time of the simulation-based approach. The authors suppose that the performance metric f is unknown in closed form but can, as in the simulation-based approaches discussed above, be evaluated subject to noise at particular capacity choices according to a random variable $\widehat{f}(\boldsymbol{c})$ with $\mathbb{E}\widehat{f}(\boldsymbol{c}) = f(\boldsymbol{c})$. As in the analytic-approximation approach, they propose to approximate the performance metric f with some functional form $\widetilde{f}$ based on analytical approximations for the network and to then optimise $\widetilde{f}$. This analytic approximation is then augmented with simulation: the optimisation is performed ensuring that $\widehat{f}(\boldsymbol{c}^{(0)}) = \widetilde{f}(\boldsymbol{c}^{(0)})$ for some $\boldsymbol{c}^{(0)} \in \mathcal{C}$. These two steps are then iterated, creating a sequence $\left(\boldsymbol{c}^{(n)}, n \in \mathbb{N}_0\right)$, and it is shown that for a specific model, with a well chosen $\widetilde{f}$ form, $\boldsymbol{c}^{(n)} \to \widehat{\boldsymbol{c}}$ where $\widehat{\boldsymbol{c}}$ is close to $\boldsymbol{c}^\star$. The hybrid approach was shown in [11] to work exceptionally well in stochastic networks without blocking. Notably this approach does not require estimates of the Jacobian or Hessian matrices.

Their approach, however, does not easily carry over to our setting. The addition of blocking fundamentally changes the dynamics of the stochastic network, and the relevant objective function changes as well, from a functional of the queue length to one of the blocking probabilities. Moreover, we introduce additional complexity by looking at the discrete problem of optimizing over the number of servers at each station in the network rather than the service rate as in [11]. In this chapter we describe how to modify the approach from [11] to stochastic networks with blocking, by proposing and evaluating the practical performance of several functional forms. We apply this approach to a realistic example, test the method using standard and non-standard networks, and show that it can outperform stochastic approximation.

The remainder of this chapter is organised as follows. In the next section an overview of the functional-form optimisation approach introduced in [11] is given. In Section 8.3 the model we are interested in and the relevant objective function are given. Following this, in Section 8.4 we detail our search for a good functional form. Then, in Section 8.5 we summarise our findings in the form of two algorithms that can be used to find approximate optimisers for our objective function. In Section 8.6 we demonstrate that these algorithms work well on a realistic network under a variety of scenarios. We conclude in Section 8.8.

8.2 Outline of the functional form approach

In this section we provide an overview of the functional-form optimisation approach developed in [11], which we will later apply to our stochastic network setting with blocking. As introduced in the previous section, the object of study is an objective function which

takes values in the real numbers

$$f : \mathcal{C} \to \mathbb{R},$$

that represents a performance metric of interest, where $\mathcal{C}$ is a set of network parameters that we aim to optimise over. For instance, $\boldsymbol{c} \in \mathcal{C}$ could parametrize the service rates or number of servers at each station of a queueing network. The f could incorporate the transient or steady-state behaviour of the network. For instance, in [11] the authors looked at the weighted expected number of jobs in the system in equilibrium, and in this chapter we consider the rate in steady-state at which jobs depart from the system without entering service (fraction of blocked jobs).

Determining a value

$$\boldsymbol{c}^\star \in \arg\max_{\boldsymbol{c}\in\mathcal{C}} f(\boldsymbol{c}), \tag{8.3}$$

is often crucial to ensure that the system of interest operates in an acceptable manner. As discussed in Section 8.1, owing to the complexity of our stochastic network, the function f is typically unknown and can only be evaluated according to a random variable $\widehat{F}$ such that $\mathbb{E}\widehat{F}(\boldsymbol{c}) = f(\boldsymbol{c})$. Using time-consuming simulation it is possible to obtain samples of $\widehat{F}$, which we denote $\widehat{f}$. It is desired to use the information contained in these samples to find an approximation of $\boldsymbol{c}^\star$. The objective of this chapter is to develop a method for closely approximating $\boldsymbol{c}^\star$ using as few evaluations of $\widehat{F}$ as possible, for a specific model that is detailed in the next section.

The functional-form optimisation approach utilises pre-determined or assumed knowledge of f to augment the information obtained from simulation, to speed up the process of approximating $\boldsymbol{c}^\star$. This structural information is expressed in a closed-form expression $\widetilde{f}(\boldsymbol{c}, \boldsymbol{\tau})$, which is both a function of the network parameters $\boldsymbol{c}$ and some (potentially vector-valued) $\boldsymbol{\tau}$, that is used to tune $\widetilde{f}$ so that it fits f well locally. Depending on the complexity of $\widetilde{f}$ we then solve either analytically or numerically for

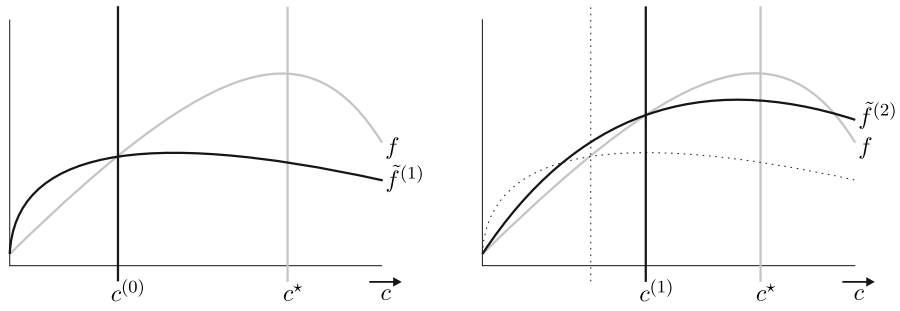
$$\boldsymbol{c}' \in \arg\max_{\boldsymbol{c}\in\mathcal{C}} \widetilde{f}(\boldsymbol{c}, \boldsymbol{\tau}), \tag{8.4}$$

to approximate $\boldsymbol{c}^\star$. Here the $\widetilde{f}$ is selected to ensure that (8.4) can be solved in closed form, or using a fast numerical procedure, in contrast to (8.3).

We assume that $\boldsymbol{\tau}$ is chosen from some set. The quality of the approximation (8.4) to (8.3) depends on: (i) whether the set of functions $\{\widetilde{f}(\cdot, \boldsymbol{\tau})\}_{\boldsymbol{\tau}}$ has elements which approximate f well, and (ii) whether we can reliably identify these high performance elements (in terms of $\boldsymbol{\tau}$). As we will see, a successful application of this approach relies on choosing a good $\widetilde{f}$, which requires knowledge of the fundamental behaviour of the network. For the remainder of this section we assume that a good form is known and give an iterative procedure for choosing $\boldsymbol{\tau}$. The sequence of samples $(\widehat{f}^{(n)}, n = 1, \ldots, N)$ are used to guide our selection of $\boldsymbol{\tau}$.

Given an initial value of $\boldsymbol{c} = \boldsymbol{c}^{(0)}$, which can be chosen randomly or using expert opinion, we first evaluate $\widehat{f}(\boldsymbol{c}^{(0)})$ using simulation. We then set $\widehat{f}(\boldsymbol{c}^{(0)}) = \widetilde{f}(\boldsymbol{c}^{(0)}, \boldsymbol{\tau}^{(1)})$ and solve for $\boldsymbol{\tau}^{(1)}$. We thus arrive at our first approximation function $\widetilde{f}^{(1)}(\cdot) := \widetilde{f}(\cdot, \boldsymbol{\tau}^{(1)})$, see Figure 8.1a[1]. Then, as in (8.4), we find $\boldsymbol{c}^{(1)} \in \arg\max_{\boldsymbol{c}} \widetilde{f}^{(1)}(\boldsymbol{c})$ as our first approximation for $\boldsymbol{c}^\star$.

[1] Note that while in Figure 8.1a the domain of f is one-dimensional for ease of presentation, in general we are interested in high-dimensional stochastic networks, which is one of the main causes for the complexity of the problem under consideration.



(a) step 1 (b) step 2

Figure 8.1: Two iteration steps of a functional-form optimisation algorithm.

Having found our first approximation for $\boldsymbol{c}^\star$ in $\boldsymbol{c}^{(1)}$, we then refine our approximation for f around $\boldsymbol{c}^{(1)}$. This is done by evaluating $\widehat{f}(\boldsymbol{c}^{(1)})$ using simulation, and solving $\widehat{f}(c^{(1)}) = \widetilde{f}(\boldsymbol{c}^{(1)}, \boldsymbol{\tau}^{(2)})$ for $\boldsymbol{\tau}^{(2)}$. This results in a new approximation $\widetilde{f}^{(2)} = \widetilde{f}(\cdot, \boldsymbol{\tau}^{(2)})$ that intersects f at $\boldsymbol{c} = \boldsymbol{c}^{(1)}$ in expectation, see Figure 8.1b. So while the selection of the functional form $\widetilde{f}$ requires fundamental insights into the behaviour of queueing networks, in contrast to purely analytic approximations this approach does not rely solely on inaccurate queueing formulas, since the simulation is used to evaluate the stochastic network.

Generalising this procedure we arrive at the sequence of approximating functions:

$$\widetilde{f}^{(n)}(\cdot) := \widetilde{f}(\cdot, \boldsymbol{\tau}^{(n)}), \quad n = 1, 2, \dots, \tag{8.5}$$

with corresponding maximisers

$$\boldsymbol{c}^{(n)} := \operatorname*{arg\,max}_{\boldsymbol{c} \in \mathcal{C}} \widetilde{f}^{(n)}(\boldsymbol{c}), \quad n = 1, 2, \dots . \tag{8.6}$$

In each iteration the tuning parameter $\boldsymbol{\tau}^{(n)}$ is obtained by solving

$$\widehat{f}(\boldsymbol{c}^{(n-1)}) := \widetilde{f}(\boldsymbol{c}^{(n-1)}, \boldsymbol{\tau}^{(n)}), \quad n = 1, 2, \dots . \tag{8.7}$$

Unless it appears that the procedure will not converge, we continue with this until the difference $||\boldsymbol{c}^{(n)} - \boldsymbol{c}^{(n-1)}||$ is sufficiently small. This approach relies on the fact that $\widetilde{f}(\cdot, \boldsymbol{\tau}^{(n)})$ provides a good approximation for f around $\boldsymbol{c} = \boldsymbol{c}^{(n-1)}$ and consequently the algorithm is likely to move in the correct direction in each iteration step. This is also where the gains in computational costs are made compared to methods which exclusively use simulation-based optimisation. Instead of, for instance, having to run many simulations in order to estimate the Jacobian (or Hessian) as is done in stochastic approximation, the functional-form approach essentially uses $\widetilde{f}$ (which is a function $\mathcal{C} \to \mathbb{R}$) to provide $\nabla \widetilde{f}(\boldsymbol{c})$ as an approximation for the gradients $\nabla f(\boldsymbol{c})$, and requires only a single evaluation $\widehat{f}$ per iteration. In addition to these computational savings, we also believe that this approach is less sensitive to the variance of $\widehat{F}$ than approaches that rely on estimates of gradients (since it uses only point estimates which are likely to be less noisy than gradient estimates).

In case this approximation needs to be refined further, one can use the final $\boldsymbol{c}^{(n)}$

as a starting point for a simulation-based optimisation approach with guaranteed convergence properties, such as stochastic approximation. This second stage may have improved accuracy over the functional-form approach described above, but can be computationally costly. However, by first obtaining a near-optimal solution using the fast hybrid functional-form approach, the more expensive purely simulation-based approach requires fewer iterations to find the optimiser, thus significantly reducing the overall computational costs compared to using simulation-based optimisation exclusively.

8.3 Stochastic networks with blocking

We next describe the model and the objective function considered in the remainder of the chapter in Section 8.3.1, and provide some examples in Section 8.3.2. It is worth remarking upon that the applicability of our approach and even of our eventual choice of functional form is by no means limited to the class of models described below. As discussed in Section 8.2, the full model is only evaluated using simulation (in order to estimate the $f(\boldsymbol{c}^{(n-1)})$), and one may add many details and extensions to the simulation model that are not present in the model outline below. However, it is important to retain the basic components of this model (e.g., blocking, no waiting room) in order to obtain the best possible results.

8.3.1 Model outline

We consider a network consisting of a set of stations $\mathcal{L} := \{1, \ldots, L\}$. Each station could, for example, represent a base station in a cellular network, or a stock point in a spare parts network. Customers arrive into the network according to extraneous arrival processes, move between stations to receive service, and then depart from the network. Each station l has c_l servers that can each serve a job. We refer to c_l as the capacity of station l, and denote $\boldsymbol{c} = (c_1, \ldots, c_L)$. If a job arrives at a station to find no servers available it will depart from the network immediately, and we say it has been *blocked*. If a job is accepted for service, then it will remain at the station for a generally distributed amount of time with finite expectation — there are no other restrictions imposed on the service distributions.

There are R job classes $\mathcal{R} = \{1, \ldots, R\}$. Each class r is associated with N_r stations collected in the vector $\psi_r = (l_1^{(r)}, l_2^{(r)}, \ldots, l_{N_r}^{(r)})$, $l_i^{(r)} \in \mathcal{L}$. We assume that any station appears at most once in any set ψ_r, i.e., $l_i^{(r)} \neq l_j^{(r)}$ for all $i \neq j$ and all fixed $r \in \mathcal{R}$. A class-r job arriving to the system is first served at station $l_1^{(r)} \in \mathcal{L}$. For $i = 1, \ldots, N_r - 1$, upon completion of service at station $l_i^{(r)}$ the job next attempts service at station $l_{i+1}^{(r)}$. Upon completion of service at station $l_{N_r}^{(r)}$ the job leaves the network.

There is a network operator who must expend resources to maintain the servers, and is incentivised to do this by revenue collections when jobs complete service at each station. Each server at station l costs $\theta_l > 0$ per unit time to provision, and we denote $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_L)$. A service completion of a class- r job at station $l_i^{(r)}$ generates $\overline{\theta}_{r,i} > 0$ revenue for the network operator, quantities which we store in $\overline{\boldsymbol{\theta}}$. If a job is blocked at a station $l_i^{(r)}$ with $i > 1$, then the operator retains the revenue generated by the successful service completions at the preceding stations $l_1^{(r)}, \ldots, l_{i-1}^{(r)}$. Addressing the trade-off between keeping the capacity costs low and having high revenue collection (by ensuring few blockages) is the purpose of our model.

We assume that we are able to observe arrivals and blockages at each station over a long time horizon (through simulation), and that we can distinguish the class of each arrival and blocked job. The number of class-r arrivals to station $l_i^{(r)}$ in the time interval $[0,t]$ is given by $A_{r,i}(t)$. For $i>1$ the quantity $A_{r,i}(t)$ clearly depends on the blocking experienced by that job class at stations $l_1^{(r)},\ldots,l_{i-1}^{(r)}$ and the exogenous arrivals to the network of class-r jobs. The quantity $A_{r,1}(t)$ is the number of exogenous class r arrivals to the network (and station $l_1^{(r)}$) in the time interval $[0,t]$. Based on our observations we are able to determine the long-run average arrival rate of class-r, which we assume exists and converges with probability 1 to a deterministic constant $\lambda_r \in \mathbb{R}_+$ as time t goes to infinity:

$$\frac{1}{t}\int_0^t A_{r,1}(s)\mathrm{d}s \to \lambda_r\,, \quad \text{as} \quad t\to\infty\,. \tag{8.8}$$

We make no further assumptions on the arrival processes. They could for example be Markovian, renewal type with independent and identically distributed inter-arrival times (see e.g., [176, 132]), or could have inter-arrival times that are not identically distributed by being within the class of Markovian arrival processes (see e.g., [118, 120]).

The number of class-r blockages at station $l_i^{(r)}$ in $[0,t]$ is given by $B_{r,i}(t)$, which we are also able to observe over very long time-periods. Combining this with our observations of the arrivals we are able to determine the long-run proportion of class-r jobs arriving at station $l_i^{(r)}$ which are subsequently blocked at that station. We suppose this quantity, which depends on the capacity of the stations (potentially all of them), converges with probability 1 to a deterministic constant $p_{r,i}(\boldsymbol{c}) \in (0,1]$:

$$\frac{B_{r,i}(t)}{A_{r,i}(t)} \to p_{r,i}(\boldsymbol{c}) \quad \text{as} \quad t\to\infty\,. \tag{8.9}$$

We aim to optimise the performance of this system, as just described, operating in equilibrium. Let us denote by $\langle\cdot,\cdot\rangle$ the usual inner product operator, then the expected net rate of revenue generation by the system in equilibrium can be written as

$$f(\boldsymbol{c}) = -\langle \boldsymbol{c},\boldsymbol{\theta}\rangle + \sum_{r\in\mathcal{R}}\lambda_r \sum_{i=1}^{N_r}\overline{\theta}_{r,i}\prod_{j=1}^{i}\left(1-p_{r,j}(\boldsymbol{c})\right), \tag{8.10}$$

which is our objective function. This objective function is inspired by the *capacity value function* investigated in [115, 116] in the context of Erlang-B loss systems and later generalised in the previous chapter in the context of provisioning cloud computing platforms. We aim to find

$$\boldsymbol{c}^\star \in \operatorname*{arg\,max}_{\boldsymbol{c}\in\mathbb{N}^L} f(\boldsymbol{c})\,. \tag{8.11}$$

Later, in order to apply our optimization approach we need to work with continuous functions of $\boldsymbol{c}$, although the capacity presented in this section is discrete. In order to relax this feature we make the following assumption.

Assumption 34. *When a job arrives to a station with (non-integer capacity) c, if the current number of jobs at the station is equal to $\lfloor c\rfloor+1$, then the job is blocked. If upon arrival there are $\lfloor c\rfloor$ jobs already present, then the newly arriving job is accepted with probability $c-\lfloor c\rfloor$. If there are less than or equal to $\lfloor c\rfloor-1$ jobs present at an arrival epoch, then the arrival is accepted.*

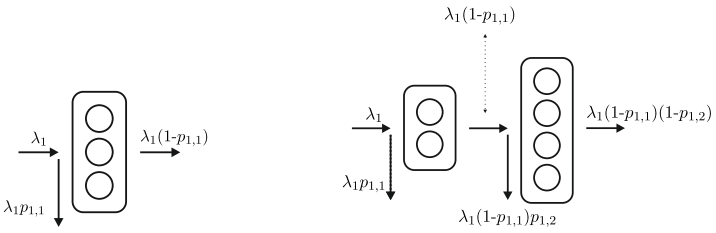
This results in a continuous relaxation of the objective function that matches (8.10) at integer points. In case our (approximate) optimal solution is non-integer, we round to the closest integer.

We also make a natural non-degeneracy assumption that the optimal capacity allocation is strictly positive. The implication of this assumption is that we are essentially working with systems that are known from naive capacity choices to be profitable, but that it is aimed to improve the profitability.

Assumption 35. *The optimal capacity is greater than or equal to one for all stations, that is* $c_l^\star \geq 1$ *for* $l = 1, \ldots, L$.

8.3.2 Specific examples

In this subsection we give four examples of systems which can be embedded into the class of models we described in the previous subsection. The first is based on extremely classical work in the field of telephony communications and consists of a single station with only one class of jobs, see Figure 8.2a. In order to show the large increase in the complexity of (8.10) that results from small changes to this simple model, in our next example we add one more station and keep the network single class. We then return to the single station case but add another class to see that this adds a comparable level of complexity. Our final example is more complex and is intended to give a flavour of the type of system our approach is intended for.



(a) single station (b) tandem network

Figure 8.2: Two small network instances.

Example 1. Consider a system where there is only a single station ($L = 1$) and c_1 units of capacity are allocated. Customers arrive to this station according to a Poisson process with rate λ_1 and have generally distributed service times with expected value μ^{-1}. Here $\mathcal{L} = \{1\}$ and $\psi_1 = (1)$ with $l_1^{(1)} = 1$. This system can be recognized as a M/G/c/c queue, and the long-run proportion of blocked jobs is given by the well-known Erlang-B loss formula (see e.g. [95])

$$p_{1,1}(c_1) = \frac{(\lambda_1/\mu)^{c_1}/c_1!}{\sum_{i=0}^{c_1} \left(\lambda_1/\mu\right)^i/i!}\,. \tag{8.12}$$

Capacity costs θ_1 per time unit to provision, and a successful service completion results in the operator collecting $\overline{\theta}_{1,1}$ revenue, so the objective function (8.10) can be explicitly evaluated using (8.12) as

$$f(c_1) = -c_1\,\theta_1 + \lambda_1\,\overline{\theta}_{1,1}\left(1 - p_{1,1}(c_1)\right), \tag{8.13}$$

which must be maximised over $\mathcal{C} = \mathbb{N}$. Notice that this function is not directly amenable to calculus-based methods of optimization, due to the discrete nature of the denominator of the blocking probability (8.12). It is, however, quite straightforward to evaluate on modern computers and so even for large values of c_1 the optimal capacity can be found using brute force. □

Example 2. Consider a system with two stations of capacity c_1 and c_2 respectively, see Figure 8.2b. In this example there is still only a single class of jobs, which arrive to the first station according to a Poisson process with rate λ_1. Each job will first attempt service at the first station and then, upon a successful service completion, attempt service at the second station. Here $\mathcal{L} = \{1, 2\}$, $\mathcal{R} = \{1\}$, $\psi_1 = (1, 2)$ with $l_1^{(1)} = 1$ and $l_2^{(1)} = 2$. A key observation to make is that although $p_{1,1}(c_1)$ is independent of the capacity of the second station, $p_{1,2}(\boldsymbol{c})$ depends on both c_1 and c_2. This is because greater capacity at the first station results in a greater number of successfully processed jobs continuing to the second station.

Since the operator collects $\overline{\theta}_{1,1}$ and $\overline{\theta}_{1,2}$ revenue for completions at the first and second stations respectively, our objective function (8.10) for this system is

$$f(\boldsymbol{c}) = -\theta_1 c_1 - \theta_2 c_2 + \lambda_1 \overline{\theta}_{1,1} \big(1 - p_{1,1}(c_1)\big) + \lambda_1 \overline{\theta}_{1,2} \big(1 - p_{1,1}(c_1)\big)\big(1 - p_{1,2}(\boldsymbol{c})\big), \quad (8.14)$$

which must be maximised over $\mathcal{C} = \mathbb{N}^2$. In Section 8.7.1 we provide an explicit expression for $f(\boldsymbol{c})$, that we then use to evaluate the performance of our algorithm in Section 8.4.3. It is seen that the complex interactions between blocking proportions at each station and capacity allocations at other stations are the primary reasons our model is inherently more difficult to handle than the models considered in [11]. □

Example 3. Consider a system with one station of capacity c_1. In this example we add a second class of jobs to Example 1, so $\mathcal{L} = \{1\}$ and $\mathcal{R} = \{1, 2\}$. Jobs of class r arrive to the station according to a Poisson process with rate λ_r. Each job attempts service at the station and then, upon a successful service completion departs the system. Here $\psi_1 = \psi_2 = (1)$ with $l_1^{(1)} = 1$ and $l_1^{(2)} = 1$.

Since the operator collects $\overline{\theta}_{1,1}$ and $\overline{\theta}_{2,1}$ revenue for completions of the first and second classes respectively, our objective function (8.10) for this system is

$$f(\boldsymbol{c}) = -\theta_1 c_1 + \lambda_1 \overline{\theta}_{1,1} \big(1 - p_{1,1}(c_1)\big) + \lambda_2 \overline{\theta}_{2,1} \big(1 - p_{2,1}(c_1)\big) \quad (8.15)$$

which must be maximised over $\mathcal{C} = \mathbb{N}^2$. □

Example 4. Consider a system of six stations $\mathcal{L} = \{1, \ldots, 6\}$ with finite capacities $\boldsymbol{c} = (c_1, \ldots, c_6)$ and 12 customer classes $\mathcal{R} = \{1, \ldots, 12\}$. Suppose these classes visit stations in the following orders $\psi_1 = (1)$, $\psi_2 = (1, 3)$, $\psi_3 = (1, 3, 5)$, $\psi_4 = (1, 4)$, $\psi_5 = (1, 4, 5)$, $\psi_6 = (1, 4, 6)$, $\psi_7 = (2)$, $\psi_8 = (2, 4)$, $\psi_9 = (2, 4, 6)$, $\psi_{10} = (2, 3)$, $\psi_{11} = (2, 3, 6)$, and $\psi_{12} = (2, 3, 5)$. Jobs arrive to the first station of each class according to a renewal process and each station has a different service time that is the same for all classes. Capacity at each station costs θ_l to provision per time unit, so $|\boldsymbol{\theta}| = 6$, and completion of a class-r job at station l generates $\overline{\theta}_{r,l}$ revenue, implying $|\overline{\boldsymbol{\theta}}| = 28$. We apply our algorithm to this system in Section 8.6. □

8.4 Finding the right functional form

We are now in a position to use the functional-form approach outlined in Section 8.2 to find a good approximation for the optimal resource allocation (8.11). Recall that we cannot solve (8.11) directly because the objective function (8.10) is typically not known in closed-form, and we must rely on simulation. We believe that a tailormade functional form approach to a specific model that we are interested in has significant benefits over existing frameworks, such as gradient ascent, which assume a generic objective function.

In order to apply our approach we need to find a good functional-form $\widetilde{f}$. We do so in this section in four stages:

(i) In Section 8.4.1 we consider several classes of functional forms for a single-station network;

(ii) In Section 8.4.2 we modify the functional forms to remove the need for a numerical solver, and speed up the procedure;

(iii) In Section 8.4.3 we consider functional forms for networks with multiple stations; and

(iv) In Section 8.4.4 we consider functional forms for multiple customer classes.

In Section 8.5 we consider the general case, using the insights obtained in Sections 8.4.1–8.4.4. The reason for taking this gradual approach is to allow us to illustrate our methodology using small network instances first, and to find the right class of functional-form approximations by eliminating those that do not work well even for small instances. So while the algorithm for the general case in Section 8.5 is self-contained, the present section provides the necessary intuition and justification for the general algorithm.

8.4.1 Single-station case

We now focus on the single-station single-class network outlined in Example 1 of Section 8.3.2 and depicted in Figure 8.2a. Recall from (8.13) that in this case the objective function is given by

$$f(c_1) = -c_1\,\theta_1 + \lambda_1\,\overline{\theta}_{1,1}\left(1 - p_{1,1}(c_1)\right),$$

and we aim to find the maximizing server allocation $c_1 \in \mathbb{N}$. Here the costs θ_1 and rewards $\overline{\theta}_{1,1}$ are known constants, and the arrival rate λ_1 is independent of the capacity and is assumed known (alternatively it can be found from simulation). The blocking probability $p_{1,1}(c_1)$, however, depends on the long-term behaviour of the underlying stochastic process, and is in general not known explicitly. The exception of course is the case of Poisson arrivals, where the $p_{1,1}$ is given by (8.12).

Thus, in order to find a good functional-form approximation $\widetilde{f}$, we need to find a good function $\widetilde{p}_{1,1}(\cdot, \tau_{1,1})$ to approximate $p_{1,1}$:

$$\widetilde{f}(c_1, \tau_{1,1}) = -c_1\,\theta_1 + \lambda_1\,\overline{\theta}_{1,1}\left(1 - \widetilde{p}_{1,1}(c_1, \tau_{1,1})\right). \tag{8.16}$$

This is because $p_{1,1}$ contains all of the complex aspects in f, the remainder of f only serves to connect the intricate revenue-reducing blocking behaviour in the system with the simple cost structure we have assumed. Let $\varepsilon > 0$ be a desired level of accuracy and $N \in \mathbb{N}$ be the maximum number of steps allowed. Given the right functional-form choice

for $\widetilde{p}_{1,1}$, we can implement the outline of the iterative algorithm described in (8.5)–(8.7) as follows.

Algorithm 36. *Single-Station Optimisation*
`Initialize: Set` $n = 1$ `and choose` $c_1^{(0)} > 0$, $\varepsilon > 0$, $N \in \mathbb{N}$.

(I) `Evaluate` $\widehat{p}_{1,1}(c_1^{(n-1)})$ `using simulation.`

(II) `Compute` $\tau_{1,1}^{(n)}$ `by solving`

$$\widehat{p}_{1,1}(c_1^{(n-1)}) = \widetilde{p}_{1,1}(c_1^{(n-1)}, \tau_{1,1}^{(n)}) \,. \tag{8.17}$$

(III) `Approximate the optimal capacity by finding`

$$c_1^{(n)} \in \operatorname*{arg\,max}_{c_1 > 0} \quad -c_1\,\theta_1 + \lambda\,\overline{\theta}_{1,1}\left(1 - \widetilde{p}_{1,1}(c_1, \tau_{1,1}^{(n)})\right) .$$

(IV) `If` $||c_1^{(n-1)} - c_1^{(n)}|| < \varepsilon$ `or` $n > N$: `output` $c_1^{(n)}$.
`Else: set` $n = n + 1$ `and return to` (I).

In order to find a good $\widetilde{p}_{1,1}$, we first list some key properties we would like it to have. It should behave similar to the blocking probability in a loss system, as a function of c_1. First, $\widetilde{p}_{1,1}$ should be decreasing and convex in c_1, so that as the capacity increases, blocking probability decreases. Moreover, we require $p_{1,1}(0,\cdot) = 1$ (if there is no capacity, always block) and $\lim_{c\to\infty} p_{1,1}(c,\cdot) = 0$ (as the capacity grows large, blocking probability goes to 0). Finally, we would like there to be a unique $\tau_{1,1}$-solution to (8.17), and ideally $\widetilde{p}_{1,1}$ should be analytically differentiable in c_1 with a solution that we can identify so that numerical solutions of Step (III) are not necessary. Summarising, we search for $\widetilde{p}_{1,1}$ satisfying the properties:

(i) $\widetilde{p}_{1,1}$ is convex and decreasing in c_1;

(ii) $\widetilde{p}_{1,1}(0,\cdot) = 1$;

(iii) $\lim_{c\to\infty} \widetilde{p}_{1,1}(c,\cdot) = 0$;

(iv) for all $c_1 \geq 0$ there is a unique $\tau_{1,1}$-solution to (8.17); and

(v) $\widetilde{p}_{1,1}$ is differentiable in $c_1 \geq 0$.

Although there exists a substantial literature analysing the Erlang-B blocking formula (8.12) (see, e.g., [177, 178, 179, 180]), functional forms inspired by these works did not meet the requirements (iv) and (v), and lead to poor approximations. Instead, after extensive numerical exploration of a variety of forms, we selected two far simpler expressions that showed promise: the ratio form

$$\widetilde{p}_{1,1}(c_1, \tau_{1,1}) = \left(1 + (\tau_{1,1}\,c_1)^k\right)^{-1} , \tag{8.18}$$

and the Weibull form

$$\widetilde{p}_{1,1}(c_1, \tau_{1,1}) = \exp\left(-(\tau_{1,1}\,c_1)^k\right) . \tag{8.19}$$

Note that both expressions contain an additional variable $k \geq 0$ that we can tweak, thus each inducing a family of functional forms. Our primary source of inspiration for these

functional forms was complements of cumulative probability distribution functions for random variables with non-negative support. Some of these forms were immediately ruled out due to their complexity, for example the gamma and log-normal distribution functions were eliminated due to the presence of factorials and error functions. We performed the computations presented later in this section for a variety of different forms (i.e., Pareto, trapezoidal, reciprocal), and (8.18) and (8.19) showed the most potential. We also investigated switching the role of k and $\tau_{1,1}$ in (8.19), but the results from doing so were not encouraging.

For both the ratio form and the Weibull form, Step (II) of Algorithm 36 can be computed in closed form. After some straightforward manipulations, it turns out that for the ratio form, the solution of (8.17) can be written as

$$\tau_{1,1}^{(n)} = \frac{1}{c_1^{(n-1)}} \left(\frac{1 - \widehat{p}_{1,1}(c_1^{(n-1)})}{\widehat{p}_{1,1}(c_1^{(n-1)})} \right)^{1/k}, \tag{8.20}$$

and for the Weibull form we have that

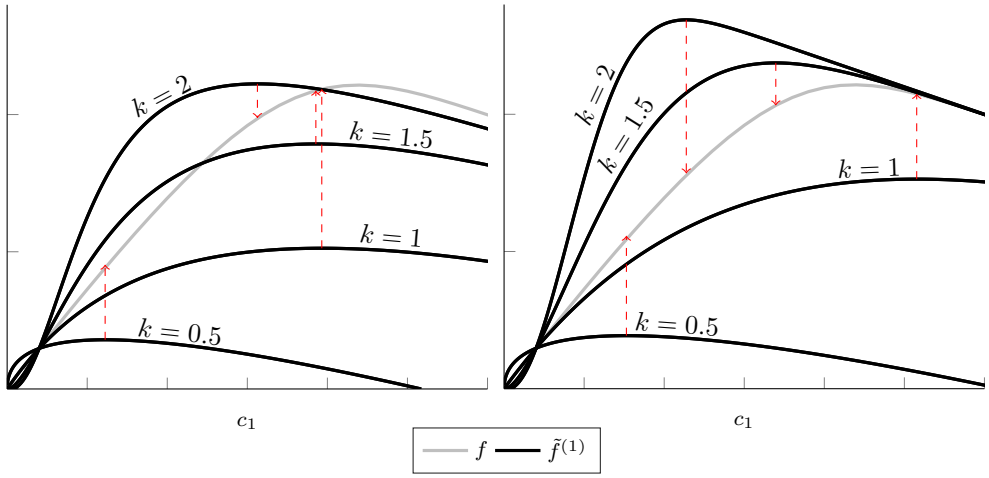
$$\tau_{1,1}^{(n)} = \frac{1}{c_1^{(n-1)}} \left(-\log(\widehat{p}_{1,1}(c_1^{(n-1)})) \right)^{1/k}. \tag{8.21}$$

In both cases, Step (III) of Algorithm 36 is solved numerically.

In order to illustrate the quality of these functional forms, we plot in Figure 8.3 the objective function f and several typical approximations $\widetilde{f}$ based on both the ratio and Weibull forms for different values of k. We consider the case with $\lambda_1 = 16$, $\theta_1 = 0.2$, and $\overline{\theta}_{1,1} = 1$; although the following discussion holds more generally to the other parameter choices we experimented with. In Figure 8.3a we plot $f(c_1)$ and $\widetilde{f}(c_1, \tau_{1,1}^{(1)})$ from (8.16) with $\widetilde{p}_{1,1}$ the ratio form (8.18) for $k = 0.5, 1, 1.5, 2$. Here $\tau_{1,1}^{(1)}$ is obtained from (8.20), starting from $c_1^{(0)} = 2$. In Figure 8.3b we do the same for the Weibull form (8.19) and the corresponding value of $\tau_{1,1}^{(1)}$ is obtained from (8.21).

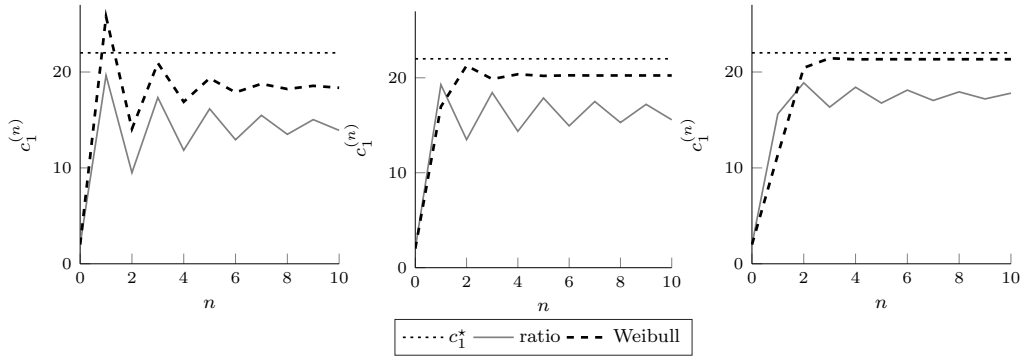
Inspection of these figures shows that the ratio form with $k = 1.5$ or $k = 1$ appears to result in a value of $c_1^{(1)}$ close to optimal. This is despite these functions being quite different in terms of how closely they track the actual value of the objective function, which highlights the importance of the approximating function having similar curvature (derivative values) to the actual objective function near the true optimiser. For both families of functions, $k = 0.5$ performs poorly.

The results in Figure 8.3 can be viewed as a single iteration step of Algorithm 36. In Figure 8.4 we show how the $c_1^{(n)}$ evolve for both forms, with $k = 1, 1.5, 2$. It can be seen that the Weibull form with $k = 2$ returns the best approximate optimiser; this is perhaps surprising given that after 1 iteration this form appears to perform quite poorly compared to the others, as seen in Figure 8.4. To further investigate this, we plot in Figure 8.5 both $f(c_1)$ and $\widetilde{f}^{(n)}(c_1, \tau)$ from (8.16) with $\widetilde{p}_{1,1}(c_1, \tau_{1,1}^{(n)})$ the Weibull form (8.19) with $k = 2$, $n = 1, 2, 3, 4$. That is, we compare the true objective function to the approximate objective functions obtained from the first four iterations of Algorithm 36. It is clear that by the fourth iteration this form performs very well locally around the true optimiser.



(a) ratio (b) Weibull

Figure 8.3: Approximations with $k = 0.5, 1, 1.5, 2$.



(a) $k = 1$ (b) $k = 1.5$ (c) $k = 2$

Figure 8.4: Evolution of Algorithm 36.

8.4.2 Speeding up the algorithm

In Section 8.4.1 we presented two classes of functional form, and demonstrated that for the single-station case under consideration these perform very well with the right parameters. The Weibull form with $k = 2$ was especially promising in all of the experiments we conducted. We also observed, however, that our functional forms are generally not amenable to standard calculus-based analytical (i.e, non-numerical) methods for solving Step (III) of Algorithm 36. Resorting to numerical methods to solve this step does not generally introduce speed or implementation issues for small to moderate sized systems, but for large networks a closed-form solution is preferred. The goal of this section is to modify the functional forms from Section 8.4.1 to allow for an analytical solution of Step (III) of Algorithm 36, while retaining their good performance.

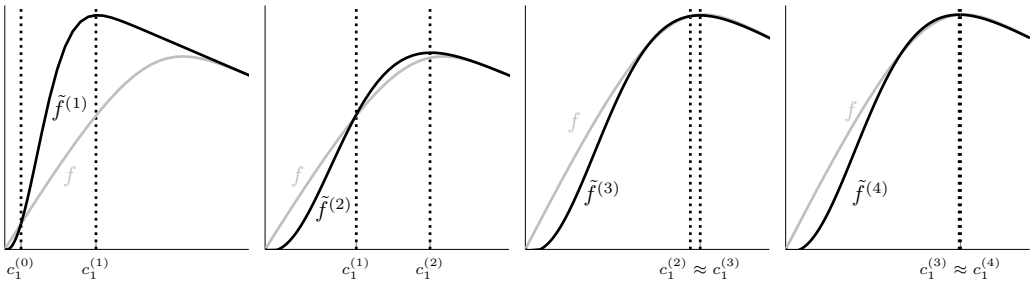


Figure 8.5: First four iterations for Weibull form with $k = 2$.

Recall that Step (III) of Algorithm 36 requires us to find $c^{(n)}$ that maximizes

$$\widetilde{f}(c_1^{(n)}, \tau_{1,1}^{(n)}) = -c_1^{(n)}\,\theta_1 + \lambda_1\,\overline{\theta}_{1,1}\,(1 - \widetilde{p}_{1,1}(c_1^{(n)}, \tau_{1,1}^{(n)}))\,.$$

Noting concavity and differentiability, doing so analytically requires solving

$$\mathrm{d}\widetilde{f}(c_1^{(n)}, \tau_1^{(n)})/\mathrm{d}c_1^{(n)} = 0\,.$$

For the ratio form (8.18) this means solving for $c_1^{(n)}$

$$k{c_1^{(n)}}^{k-1}{\tau_{1,1}^{(n)}}^{k}\left(1 + (\tau_{1,1}^{(n)} c_1^{(n)})^k\right)^{-2} = -\frac{\theta_1}{\lambda_1\overline{\theta}_{1,1}}\,, \tag{8.22}$$

and for the Weibull form (8.19),

$$k{c_1^{(n)}}^{k-1}{\tau_{1,1}^{(n)}}^{k}\exp\left(-(\tau_{1,1}^{(n)} c_1^{(n)})^k\right) = -\frac{\theta_1}{\lambda_1\overline{\theta}_{1,1}}\,. \tag{8.23}$$

To illustrate why solving (8.22) and (8.23) is analytically intractable in general, consider the Weibull functional form with $k = 2$, in which case (8.23) reduces to

$$c_1^{(n)}\exp\big(-(\tau_{1,1}^{(n)} c_1^{(n)})^2\big) = -\frac{\theta_1}{2{\tau_{1,1}^{(n)}}^2\lambda_1\overline{\theta}_{1,1}}\,. \tag{8.24}$$

Solving this relies on evaluation of the *Lambert W function*, which is well-known to require numerical approximation to be evaluated.

The key difficulty in solving (8.22) and (8.23) in terms of $c_1^{(n)}$ is that $c_1^{(n)}$ appears twice in each equation, once in an exponential term and once in a polynomial term. We propose to replace one of the $c_1^{(n)}$ appearances with $c_1^{(n-1)}$ and to the solve for the former in terms of the latter. It turns out that this straightforward solution turns out to be very effective. In all cases we found that it works well if we replace the $c_1^{(n)}$ which appears as a polynomial term. So for the ratio function, (8.22) reduces to

$$k{c_1^{(n-1)}}^{k-1}{\tau_{1,1}^{(n)}}^{k}\left(1 + (\tau_{1,1}^{(n)} c_1^{(n)})^k\right)^{-2} = \frac{\theta_1}{\lambda_1\overline{\theta}_{1,1}}\,, \tag{8.25}$$

and for the Weibull function we obtain

$$kc_1^{(n-1)^{k-1}}\tau_{1,1}^{(n)^k}\exp\left(-(\tau_{1,1}^{(n)}c_1^{(n)})^k\right)=\frac{\theta_1}{\lambda_1\overline{\theta}_{1,1}}\,. \tag{8.26}$$

By integrating the left-hand side of (8.25) and (8.26) with respect to $c_1^{(n)}$ we obtain the modified ratio functional form and the modified Weibull functional form, respectively. These results are summarized in Table 8.1, together with the functional forms from Section 8.4.1. Note that $\tau_{1,1}^{(n)}$ in Table 8.1 is the same for (c) and (d) as for (a) and (b); this is because $\tau_{1,1}^{(n)}$ is computed using $c_1^{(n-1)}$ before we replace one of the $c_1^{(n)}$ to simplify Step (III) of Algorithm 36. Also observe that for these new forms we do not have property (ii), $\widetilde{p}(0)=1$ does not necessarily hold. As we saw in Figure 8.3, however, this may not be an issue so long as the curvature of $\widetilde{f}$ matches that of f around $\boldsymbol{c}^\star$.

Table 8.1: Alternative functional forms and associated tuning parameters.

	Name	$\widetilde{p}_{1,1}(c_1^{(n)},\tau_{1,1}^{(n)})$	$\tau_{1,1}^{(n)}(\widehat{p}_{1,1}(c_1^{(n-1)}))$
(a)	Ratio	$\left(1+(\tau_{1,1}^{(n)}\,c_1^{(n)})^k\right)^{-1}$	$\frac{1}{c_1^{(n-1)}}\left(\frac{1-\widehat{p}_{1,1}(c_1^{(n-1)})}{\widehat{p}_{1,1}(c_1^{(n-1)})}\right)^{1/k}$
(b)	Weibull	$\exp\left(-(\tau_{1,1}^{(n)}\,c_1^{(n)})^k\right)$	$\frac{1}{c_1^{(n-1)}}\left(-\log(\widehat{p}_{1,1}(c_1^{(n-1)}))\right)^{1/k}$
(c)	Modified ratio	$-\int_0^{c_1^{(n)}} kc_1^{(n-1)^{k-1}}\tau_{1,1}^{(n)^k}\left(1+(\tau_{1,1}^{(n)}x)^k\right)^{-2}\mathrm{d}x$	$\frac{1}{c_1^{(n-1)}}\left(\frac{1-\widehat{p}_{1,1}(c_1^{(n-1)})}{\widehat{p}_{1,1}(c_1^{(n-1)})}\right)^{1/k}$
(d)	Modified Weibull	$-\int_0^{c_1^{(n)}} kc_1^{(n-1)^{k-1}}\tau_{1,1}^{(n)^k}\exp\left(-(\tau_{1,1}^{(n)}x)^k\right)\mathrm{d}x$	$\frac{1}{c_1^{(n-1)}}\left(-\log(\widehat{p}_{1,1}(c_1^{(n-1)}))\right)^{1/k}$

Based on (8.25) and (8.26) we can explicitly replace numerical optimisation with

$$c_1^{(n)}=\frac{1}{\tau_{1,1}^{(n)}}\left(-1+\sqrt{\frac{kc_1^{(n-1)^{k-1}}\tau_{1,1}^{(n)^k}\lambda_1\overline{\theta}_{1,1}}{\theta_1}}\right)^{1/k}\vee\overline{c}\,,$$

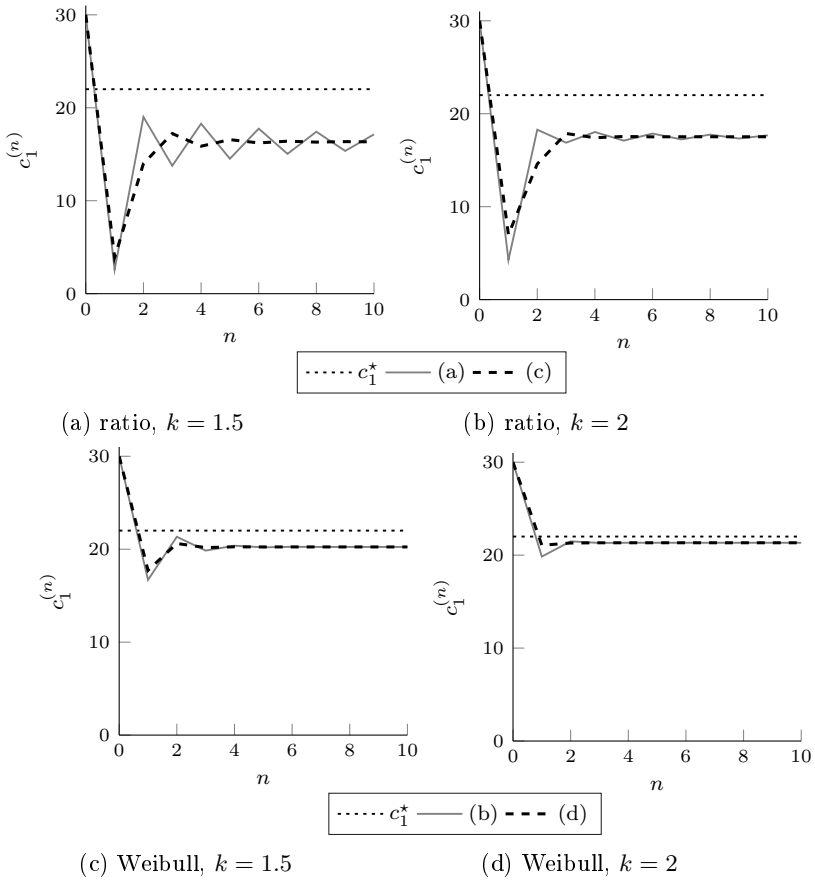
and

$$c_1^{(n)}=\frac{1}{\tau_{1,1}^{(n)}}\left(\log\left(\frac{kc_1^{(n-1)^{k-1}}\tau_{1,1}^{(n)^k}\lambda_1\overline{\theta}_{1,1}}{\theta_1}\right)\right)^{1/k}\vee\overline{c}\,,$$

in Step (III) of Algorithm 36 for the ratio and Weibull modified forms respectively. Here $\overline{c}$ is a boundary condition chosen sufficiently small to satisfy Assumption 35. In Figure 8.6 the progression of the algorithm using the modified forms (c) and (d) is compared to the case when (a) and (b) are used. In this case we have used an initial condition $c_1^{(0)}=30$. It can be seen that the forms (c) and (d) perform very similarly to (a) and (b), respectively.

8.4.3 Extension to tandem systems

In this subsection we expand our exploration to include the tandem-station single-class network outlined in Example 2 of Section 8.3.2 and depicted in Figure 8.2b. Recall from



(a) ratio, $k = 1.5$ (b) ratio, $k = 2$

(c) Weibull, $k = 1.5$ (d) Weibull, $k = 2$

Figure 8.6: Comparison of evolution of Algorithm 36 with different forms.

(8.14) that for this model the objective function is given by

$$f(\boldsymbol{c}) = -\theta_1\, c_1 - \theta_2\, c_2 + \lambda_1\, \overline{\theta}_{1,1}\, \big(1 - p_{1,1}(c_1)\big) + \lambda_1\, \overline{\theta}_{1,2}\, \big(1 - p_{1,1}(c_1)\big)\big(1 - p_{1,2}(\boldsymbol{c})\big)\,, \quad (8.27)$$

and we now aim to find the maximizing capacity allocation (c_1, c_2). Again, the costs and rewards θ_1, θ_2, $\overline{\theta}_{1,2}$, and $\overline{\theta}_{1,1}$ are known constants, and the arrival rate λ_1 is assumed known or can be found from simulation.

As in the single-station single-class case, Example 1, we need to approximate the blocking proportions $p_{1,1}$ and $p_{1,2}$ with functions $\widetilde{p}_{1,1}$ and $\widetilde{p}_{1,2}$ to find a good functional form

$$\widetilde{f}(\boldsymbol{c}, \boldsymbol{\tau}) = -\theta_1\, c_1 - \theta_2\, c_2 + \lambda_1\, \overline{\theta}_{1,1}\, \big(1 - \widetilde{p}_{1,1}(c_1, \tau_{1,1})\big) + \lambda_1\, \overline{\theta}_{1,2}\, \big(1 - \widetilde{p}_{1,1}(c_1, \tau_1)\big)\big(1 - \widetilde{p}_{1,2}(c_2, \tau_{1,2})\big)\,. \quad (8.28)$$

In this case we implement steps in (8.5)– (8.7) as follows.

Algorithm 37. *Tandem-Station Optimisation*
`Initialize: Set` $n = 1$ `and choose` $\boldsymbol{c}^{(0)} \in (0, \infty)^2$, $\varepsilon > 0$, $N \in \mathbb{N}$.

(I) `Evaluate` $\widehat{p}_{1,1}(c_1^{(n-1)})$ `and` $\widehat{p}_{1,2}(\boldsymbol{c}^{(n-1)})$ `using simulation.`

(II) For $i = 1, 2$ compute $\tau_{1,i}^{(n)}$ by solving

$$\widehat{p}_{1,i}(\boldsymbol{c}^{(n-1)}) = \widetilde{p}_{1,i}(\boldsymbol{c}^{(n-1)}, \tau_{1,i}^{(n)}) \,. \tag{8.29}$$

(III) Approximate the optimal capacity by finding

$$\boldsymbol{c}^{(n)} \in \underset{\boldsymbol{c}\in(0,\infty)^2}{\arg\max} \quad \widetilde{f}(\boldsymbol{c}, \boldsymbol{\tau}^{(n)}) \,,$$

where $\widetilde{f}$ is defined in (8.28).

(IV) If $||\boldsymbol{c}^{(n-1)} - \boldsymbol{c}^{(n)}|| < \varepsilon$ or $n > N$: output $\boldsymbol{c}^{(n)}$.
Else: set $n = n + 1$ and return to (I).

This algorithm could be implemented using any of the functional forms in Table 8.1 and solving Step (III) numerically, as in Section 8.4.1. For example, using the Weibull form results in

$$\widetilde{p}_{1,2}(c_2, \tau_{1,2}^{(n)}) = \exp\big(-(\tau_{1,2}^{(n)} c_2)^k\big) \,.$$

Seeking a closed form solution to Step (III) of the above algorithm, we evaluate (8.28) at $(c_1^{(n)}, c_2^{(n)})$ and differentiate. This results in the set of equations

$$\frac{\partial}{\partial c_1^{(n)}} \widetilde{f}(\boldsymbol{c}^{(n)}, \boldsymbol{\tau}^{(n)}) = -\theta_1 - \frac{\mathrm{d}\widetilde{p}_{1,1}(c_1^{(n)}, \tau_{1,1}^{(n)})}{\mathrm{d}c_1^{(n)}} \Big(\lambda_1\, \overline{\theta}_{1,1} + \lambda_1\, \overline{\theta}_{1,2}\, (1 - \widetilde{p}_{1,2}(c_2^{(n)}, \tau_{1,2}^{(n)})\Big) \,, \tag{8.30}$$

$$\frac{\partial}{\partial c_2^{(n)}} \widetilde{f}(\boldsymbol{c}^{(n)}, \boldsymbol{\tau}^{(n)}) = -\theta_2 - \frac{\mathrm{d}\widetilde{p}_{1,2}(c_2^{(n)}, \tau_{1,2}^{(n)})}{\mathrm{d}c_2^{(n)}} \Big(\lambda_1\, \overline{\theta}_{1,2}\, (1 - \widetilde{p}_{1,1}(c_1^{(n)}, \tau_{1,1}^{(n)}))\Big) \,. \tag{8.31}$$

Setting (8.30) and (8.31) equal to 0 results in a system of equations that we would like to solve in order to obtain a next approximation for the optimal capacity allocation.

In the previous subsection we saw that when $c_1^{(n)}$ appears multiple times in an equation that needs to be solved to find an optimal value (e.g., (8.24)), replacing all but one of the $c_1^{(n)}$ appearances with $c_1^{(n-1)}$ results in a highly simplified equation for which a closed form solution can be found. However, after doing this for (8.30) and (8.31), it turns out that each of these equations contain both $c_1^{(n)}$ and $c_2^{(n)}$ — so unfortunately solving these equations still relies on highly intricate manipulations. Instead, we propose to decouple solving for the $c_1^{(n)}$ and $c_2^{(n)}$: in (8.30) replace all appearances of $c_2^{(n)}$ with $c_2^{(n-1)}$, and in (8.31) replace all appearances of $c_1^{(n)}$ with $c_1^{(n-1)}$. To clearly indicate that we do this, when $\widetilde{f}$ is being used to optimise capacity at station l we denote it by $\widetilde{f}_l$. Consequently, we work with the set of functions $\{\widetilde{f}_1, \widetilde{f}_2\}$ in place of $\widetilde{f}$. We thus implicitly define $\widetilde{f}_1$ and $\widetilde{f}_2$ as

$$\frac{\partial}{\partial c_1^{(n)}} \widetilde{f}_1(c_1^{(n)}, \tau_1^{(n)}) = -\theta_1 - \frac{\mathrm{d}\widetilde{p}_{1,1}(c_1^{(n)}, \tau_1^{(n)})}{\mathrm{d}c_1^{(n)}} \Big(\lambda_1\, \overline{\theta}_{1,1} + \lambda_1\, \overline{\theta}_{2,1}\, (1 - \widehat{p}_{1,2}(\boldsymbol{c}^{(n-1)}))\Big) \,, \tag{8.32}$$

$$\frac{\partial}{\partial c_2^{(n)}} \widetilde{f}_2(c_2^{(n)}, \tau_2^{(n)}) = -\theta_2 - \frac{\mathrm{d}\widetilde{p}_{1,2}(c_2^{(n)}, \tau_2^{(n)})}{\mathrm{d}c_2^{(n)}} \Big(\lambda_1\, \overline{\theta}_{2,1}\, (1 - \widehat{p}_{1,1}(\boldsymbol{c}^{(n-1)}))\Big) \,. \tag{8.33}$$

This implies, for example, with a modified Weibull form with $k = 2$ (introduced in (8.26)) we have

$$\frac{\partial}{\partial c_1^{(n)}} \widetilde{f}_1(c_1^{(n)}, \tau_1^{(n)}) = -\theta_1 + 2c_1^{(n-1)} {\tau_1^{(n)}}^2 \exp\left(-(\tau_1^{(n)} c_1^{(n)})^2\right) \left(\lambda_1 \overline{\theta}_1 + \lambda_1 \overline{\theta}_2 \left(1 - \widehat{p}_{1,2}(\boldsymbol{c}^{(n-1)})\right)\right),$$

$$\frac{\partial}{\partial c_2^{(n)}} \widetilde{f}_2(c_2^{(n)}, \tau_2^{(n)}) = -\theta_2 + 2c_2^{(n-1)} {\tau_2^{(n)}}^2 \exp\left(-(\tau_2^{(n)} c_2^{(n)})^2\right) \left(\lambda_1 \overline{\theta}_2 \left(1 - \widehat{p}_{1,1}(\boldsymbol{c}^{(n-1)})\right)\right).$$

Since this is one of the modified forms, setting these to 0 and solving allows us to give the explicit expressions for $c_1^{(n)}$ and $c_2^{(n)}$:

$$c_1^{(n)} = \frac{1}{\tau_1^{(n)}} \sqrt{\log\left(\frac{2\lambda_1 c_1^{(n-1)} {\tau_1^{(n)}}^2 \left[\overline{\theta}_1 + \overline{\theta}_2 (1 - \widehat{p}_{1,2}(\boldsymbol{c}^{(n-1)}))\right]}{\theta_1}\right)} \vee \overline{c}, \tag{8.34}$$

$$c_2^{(n)} = \frac{1}{\tau_2^{(n)}} \sqrt{\log\left(\frac{2\lambda_2 c_2^{(n-1)} {\tau_2^{(n)}}^2 \overline{\theta}_2 (1 - \widehat{p}_{1,1}(\boldsymbol{c}^{(n-1)}))}{\theta_2}\right)} \vee \overline{c}, \tag{8.35}$$

where $\overline{c} > 0$ is a small constant to account for Assumption 35.

We now have four functional-form options, given in Table 8.1, each which can be varied according to a parameter k. The functional forms (a) and (b) in Table 8.1 are used in Algorithm 37 in combination with a numerical solver. The functional form (d) may be used to obtain (8.34) and (8.35) for use in Step (III) of Algorithm 37 in place of a numerical solver, and similarly for the form (c). We state these results more generally in Section 8.5.

For the simple tandem system we consider here it is possible to explicitly evaluate the objective function (8.27) using matrix analytic methods. We perform these computations in Section 8.7.1 to enable an exact evaluation of the various functional forms. These methods do not extend to more complex settings. In Table 8.2 we present the solution attained by the algorithm for each functional form with different choices of k. The arrival rate is $\lambda_1 = 16$, the cost of capacity is $\theta_1 = 0.2$ and $\theta_2 = 0.3$, revenue is $\overline{\theta}_{1,1} = \overline{\theta}_{1,2} = 1$. Using the matrix-analytic approach in Section 8.7.1, we can compute the real maximum over the objective function (8.27) to be $f(\boldsymbol{c}^\star) = 4.37$. We start the algorithm from each element of $\boldsymbol{c}^{(0)} \in \mathbb{Z}^2 \cap [1, 60]^2$ and record for each functional form and value of k the mean and variance of the objective function evaluated at the capacity allocation suggested by the algorithm over these 60 instances, as well as the mean number of iterations before the stopping condition $|c_1^{(n)} - c_1^{(n-1)}| + |c_2^{(n)} - c_2^{(n-1)}| < \varepsilon = 1$ or $n = N := 50$ is reached.

For this example, it can be seen that form (c) performs poorly compared to the others, as our modifications to obtain a closed-form solution for Step (III) of Algorithm 37 have substantially reduced accuracy. Form (b) with $k = 2$ and form (d) with $k = 3$ are the best of the remaining options. For the Weibull form, our modifications have not resulted in a reduction in accuracy, they may in fact be more accurate. Form (a) with $k = 4, 4.5, 5$ also performs well in terms of the objective, however it requires approximately 3 times as many simulations as form (b), and regardless it does not perform as well as form (d) with $k = 3$. Form (b) with $k = 2$ appears to be less accurate, but has faster convergence when compared to form (d) with $k = 3$.

Table 8.2: Objective function evaluated at output of Algorithm 37 as applied to Example 2 from Section 8.3.2. Each entry in the table summarises a mean or variance of the evaluation of the objective function at the capacity given by the final iteration from 3600 sample paths of the algorithm, each generated with a unique $\boldsymbol{c}^{(0)} \in \mathbb{Z}^2 \cap [1, 60]^2$. The actual optimal value is 4.37.

k	Mean obj. val.	Var obj. val.	Mean no. of iter.	Mean obj. val.	Var obj. val.	Mean no. of iter.
	Algorithm 37			Algorithm 37 using (8.32) and (8.33)		
	(a) Ratio			(c) Modified ratio		
1	-2.24	5.6	50	-0.3	0	3
1.5	-1.40	4.56	50	-0.04	0.04	6
2	-0.73	4.4	50	0.8	0.24	12
2.5	1.76	1.79	50	1.7	0.06	12
3	3.91	0.19	36	2.03	0.03	12
3.5	4.17	0.02	37	2.44	0.03	11
4	4.24	≈ 0	11	2.61	0.06	10
4.5	4.25	≈ 0	11	2.78	0.06	9
5	4.24	≈ 0	9	2.9	0.1	9
	(b) Weibull			(d) Modified Weibull		
1	3.81	0.01	22	-0.03	≈ 0	9
1.5	4.15	0.01	7	-0.58	0.02	31
2	4.18	≈ 0	4	4	≈ 0	12
2.5	4.11	≈ 0	5	4.2	≈ 0	8
3	4.04	≈ 0	5	4.27	≈ 0	7
3.5	3.96	≈ 0	6	4.26	≈ 0	7
4	3.84	≈ 0	6	4.26	≈ 0	7

8.4.4 Extension to two customer classes

This subsection considers Example 3 from Section 8.3.2; we return to the single station case but add an additional class of jobs. As seen previously in (8.15), the objective function in this case is

$$f(\boldsymbol{c}) = -\theta_1 c_1 + \lambda_1 \overline{\theta}_{1,1} \left(1 - p_{1,1}(c_1)\right) + \lambda_2 \overline{\theta}_{2,1} \left(1 - p_{2,1}(c_1)\right). \tag{8.36}$$

As before, this may be approximated by replacing the blocking proportions $p_{1,r}$ with any functional form $\widetilde{p}_{1,r}$ like those listed in Table 8.1:

$$\widetilde{f}(\boldsymbol{c}, \boldsymbol{\tau}) = -\theta_1 c_1 + \lambda_1 \overline{\theta}_{1,1} \left(1 - \widetilde{p}_{1,1}(c_1, \tau_{1,1})\right) + \lambda_2 \overline{\theta}_{2,1} \left(1 - \widetilde{p}_{2,1}(c_1, \tau_{2,1})\right). \tag{8.37}$$

Adapting the statement of Algorithm 37 to this model is straightforward: in Steps (I) and (II) replace $p_{1,2}$ and $\widetilde{p}_{1,2}$ with $p_{2,1}$ and $\widetilde{p}_{2,1}$ respectively, and in Step (III) use (8.37) instead of (8.28).

Seeking a closed-form solution to Step (III) of this algorithm, we differentiate $\widetilde{f}$ from

(8.37) with respect to $c_1^{(n)}$, and set this equal to 0. This results in the equation

$$\frac{\partial}{\partial c_1^{(n)}} \widetilde{f}(\boldsymbol{c}^{(n)}, \tau^{(n)}) = -\theta_1 - \frac{\mathrm{d}\widetilde{p}_{1,1}(c_1^{(n)}, \tau_{1,1}^{(n)})}{\mathrm{d}c_1^{(n)}} \lambda_1 \overline{\theta}_{1,1} - \frac{\mathrm{d}\widetilde{p}_{2,1}(c_1^{(n)}, \tau_{2,1}^{(n)})}{\mathrm{d}c_1^{(n)}} \lambda_2 \overline{\theta}_{2,1} \,. \tag{8.38}$$

We find ourselves with an equation that cannot be solved in closed form and has multiple instances of $c_1^{(n)}$ in it, for all functional forms (a)–(d). In Section 8.4.3 we overcame this difficulty by replacing all but one of the $c_1^{(n)}$ appearances with $c_1^{(n-1)}$. Because $c_1^{(n)}$ plays a fundamentally different role in the second and third terms of (8.38), it is not clear which appearance to replace with $c_1^{(n-1)}$.

Our approach is to combine $\widetilde{p}_{1,1}(c_1^{(n)}, \tau_{1,1}^{(n)})$ and $\widetilde{p}_{2,1}(c_1^{(n)}, \tau_{2,1}^{(n)})$ into a single function $\widetilde{p}_1(c_1^{(n)}, \tau_1^{(n)})$. We then replace $\widetilde{p}_{1,1}(c_1^{(n)}, \tau_{1,1}^{(n)})$ and $\widetilde{p}_{2,1}(c_1^{(n)}, \tau_{2,1}^{(n)})$ with $w_{1,1}\, \widetilde{p}_1(c_1^{(n)}, \tau_1)$ and $w_{2,1}\, \widetilde{p}_1(c_1^{(n)}, \tau_1)$, where $w_{1,1}$ and $w_{2,1}$ are appropriately chosen weights (as explained soon). This results in a functional form

$$f(\boldsymbol{c}, \tau_1^{(n)}) = -\theta_1 c_1 + \left[\lambda_1 \overline{\theta}_{1,1}(1 - w_{1,1}\widetilde{p}_1(c_1, \tau_1^{(n)})) + \lambda_2 \overline{\theta}_{2,1}(1 - w_{2,1}\widetilde{p}_1(c_1, \tau_1^{(n)}))\right]. \tag{8.39}$$

This equation is only fully specified given a procedure for finding the weights and a choice of $\widetilde{p}_1$. We now cover these two issues.

First, recall that we defined $A_{r,i}(t)$ as the total number of class r jobs to arrive in the interval $[0,t]$ at the i-th station utilised by the class, and $B_{r,i}(t)$ as the number of class r jobs to be blocked from entry to this station in $[0,t]$. As previously explained, observing sample paths of these processes allows us to estimate $p_{r,i}(\boldsymbol{c})$. Let $n_{l,r}$ denote the index of station $l_i^{(r)}$ (i.e., $n_{l,r} = i$ for station $l_i^{(r)}$). We now define

$$A_l(t) := \sum_{r:l\in r} A_{r,n_{l,r}}(t)\,,$$

as the total number of jobs arriving at station l in the interval $[0,t]$, and similarly

$$B_l(t) := \sum_{r:l\in r} B_{r,n_{l,r}}(t)\,,$$

as the total number of jobs blocked at station l in the interval $[0,t]$. As in (8.9), where we defined the quantity $p_{r,i}(\boldsymbol{c})$ to be used in our objective function (8.10), for each $l \in \mathcal{L}$ we now define $p_l(\boldsymbol{c})$ as the almost sure limit of the proportion of jobs blocked at station l:

$$\frac{B_l(t)}{A_l(t)} \to p_l(\boldsymbol{c})\,, \quad \text{as} \quad t \to \infty\,, \tag{8.40}$$

which is the long run proportion of jobs blocked at station l (we assume existence). We note that estimates $\widehat{p}_l(\boldsymbol{c})$ for $p_l(\boldsymbol{c})$ can be generated at the same time as $\widehat{p}_{r,i}(\boldsymbol{c})$ with almost no additional computational burden. The weight given to $\widetilde{p}_{r,i}$ when it is replaced with $\widetilde{p}_{l_i^{(r)}}$, which is denoted $w_{r,i}$, is given in terms of $\widehat{p}_{l_i^{(r)}}$ as follows

$$w_{r,i} := \frac{\widehat{p}_{r,i}(\boldsymbol{c})}{\widehat{p}_{l_i^{(r)}}(\boldsymbol{c})}\,.$$

Note that if the arrival processes to each class are independent Poisson processes, then this procedure is exact.

Now, we want to let $\widetilde{p}_1$ utilise any of the forms given in Table 8.1. For example, in the Weibull case with $k = 2$,

$$\widetilde{p}_1(c_1, \tau_1^{(n)}) = \exp\big(- (c_1 \tau_1^{(n)})^2 \big) ,$$

where τ_l is found by solving

$$\widehat{p}_l(c) = \widetilde{p}_l(c_1, \tau_l^{(n)}) .$$

We can now evaluate (8.39) at $c_1^{(n)}$ and differentiate to obtain

$$\frac{\partial}{\partial c_1^{(n)}} \widetilde{f}(c_1^{(n)}, \tau_1^{(n)}) = -\theta_1 - \frac{\mathrm{d}\widetilde{p}_1(c_1^{(n)}, \tau_1^{(n)})}{\mathrm{d}c_1^{(n)}} [\lambda_1 \overline{\theta}_{1,1} w_{1,1} + \lambda_2 \overline{\theta}_{2,1} w_{2,1}] .$$

For our modified functional forms (c) and (d) this can be explicitly evaluated. Upon letting $\gamma_1 = \lambda_1 \overline{\theta}_{1,1} w_{1,1} + \lambda_2 \overline{\theta}_{2,1} w_{2,1}$, for the modified Ratio form this is

$$c_1^{(n)} = \frac{1}{\tau_1^{(n)}} \left(-1 + \sqrt{\frac{\gamma_1 k {c_1^{(n-1)}}^{k-1} {\tau_1^{(n)}}^k}{\theta_1}} \right)^{1/k} \vee \overline{c} ,$$

and for the modified Weibull form this is

$$c_1^{(n)} = \frac{1}{\tau_1^{(n)}} \left(\log \left(\frac{\gamma_1 k {c_1^{(n-1)}}^{k-1} {\tau_1^{(n)}}^k}{\theta_1} \right) \right)^{1/k} \vee \overline{c} .$$

The constant $\overline{c} > 0$ is again a boundary condition to satisfy Assumption 35.

8.5 Algorithms for network setting

In this section the intuition developed in the previous section in the simple one and two station cases is synthesised into pseudocode for two algorithms that can be used to find approximate solutions to the maximisation problem

$$\boldsymbol{c}^\star \in \operatorname*{arg\,max}_{\boldsymbol{c} \in \mathbb{N}^L} f(\boldsymbol{c}) , \tag{8.41}$$

in the network setting where

$$f(\boldsymbol{c}) = -\langle \boldsymbol{c}, \boldsymbol{\theta} \rangle + \sum_{r \in \mathcal{R}} \lambda_r \sum_{i=1}^{N_r} \overline{\theta}_{r,i} \prod_{j=1}^{i} \big(1 - p_{r,j}(\boldsymbol{c})\big) ,$$

given earlier in (8.11) and (8.10). Recall that in this function:

- $\boldsymbol{c}$ is a vector of length L that contains the capacity allocated to each station;
- θ_l is the cost per unit time for maintaining capacity at station l;

- $\theta_{r,i}$ is the revenue received for processing a class-r job at the i-th station it potentially visits ($l_i^{(r)}$); and
- λ_r is the long-run arrival rate of class-r jobs (defined in (8.8));
- N_r is the maximum number of stations potentially visited by class-r jobs; and
- $p_{r,i}(\boldsymbol{c})$ is the long-run proportion of class-r jobs blocked from entry at the i-th station they potentially visit ($l_i^{(r)}$) as a function of $\boldsymbol{c}$ (defined in (8.9)).

As outlined in Section 8.2, we aim to replace f with a functional form $\widetilde{f}$ that allows (8.41) to be solved using an iterative procedure combining simulation runs and optimisation steps. In Section 8.4.1 this was achieved by replacing each $p_{r,i}$ with a function $\widetilde{p}_{r,i}$, which are summarised in Table 8.3 below. Given these forms, a method to find an approximate solution to (8.41) is given by Algorithm 38. Recall that $\varepsilon > 0$ is a target accuracy and $N \in \mathbb{N}$ is a computational budget.

Algorithm 38. *Basic functional-form optimisation*
`Initialize:` Set $n = 1$ and choose $\boldsymbol{c}^{(0)} \in (0,\infty)^L$, $\varepsilon > 0$, $N \in \mathbb{N}$.

(I) For $r \in \mathcal{R}$ for $i = 1, \ldots, N_r$ obtain $\widehat{p}_{r,i}(\boldsymbol{c}^{(n-1)})$.

(II) For $r \in \mathcal{R}$ for $i = 1, \ldots, N_r$ compute $\tau_{r,i}^{(n)}$ by solving

$$\widehat{p}_{r,i}(\boldsymbol{c}^{(n-1)}) = \widetilde{p}_{r,i}\left(c_{l_i^{(r)}}^{(n-1)}, \tau_{r,i}\right) .$$

(III) Solve to find the next guess for the optimal capacity:

$$\boldsymbol{c}^{(n)} \in \underset{\boldsymbol{c}\in(0,\infty)^L}{\arg\max} \quad -\langle \boldsymbol{c}, \boldsymbol{\theta}\rangle + \sum_{r\in\mathcal{R}} \lambda_r \sum_{i=1}^{N_r} \overline{\theta}_{r,i} \prod_{j=1}^{i} \left(1 - \widetilde{p}_{r,j}\left(c_{l_j^{(r)}}, \tau_{r,j}^{(n)}\right)\right) , \tag{8.42}$$

(IV) If $||\boldsymbol{c}^{(n-1)} - \boldsymbol{c}^{(n)}|| < \varepsilon$ or $n > N$: output $\boldsymbol{c}^{(n)}$.
Else: set $n = n + 1$ and return to (I).

Table 8.3: Functional forms and associated tuning parameters for Algorithm 38.

	Name	$\widetilde{p}_{r,i}(c_l, \tau_{r,i}^{(n)})$	$\tau_{r,i}^{(n)}(\widehat{p}_{r,i}(\boldsymbol{c}^{(n-1)}))$
(a)	Ratio	$\left(1 + (\tau_{r,i}^{(n)}\, c_l^{(n)})^k\right)^{-1}$	$\frac{1}{c_l^{(n-1)}}\left(\frac{1-\widehat{p}_{r,i}(\boldsymbol{c}^{(n-1)})}{\widehat{p}_{r,i}(\boldsymbol{c}^{(n-1)})}\right)^{1/k}$
(b)	Weibull	$\exp\left(-(\tau_{r,i}^{(n)}\, c_l^{(n)})^k\right)$	$\frac{1}{c_l^{(n-1)}}\left(-\log(\widehat{p}_{r,i}(\boldsymbol{c}^{(n-1)}))\right)^{1/k}$

Algorithm 38 requires access to a numerical solver in all but the simplest settings. Sections 8.4.2–8.4.4 developed further machinery and introduced new functional forms, for finding an approximate solution to (8.41) that enables closed-form solutions. To do this, the per-station total blocking proportion p_l was introduced in (8.40). The intuition of Section 8.4.3 points us towards using a set of functional-form approximations $\{\widetilde{f}_l\}_{l\in\mathcal{L}}$

where in iteration n of the algorithm for each $l \in \mathcal{L}$ we have a $(\widetilde{f}_l^{(n)}(c_l^{(n)}),\, c_l^{(n)} > 0)$ where all $c_{l'}^{(n)}$ with $l' \in \mathcal{L}\setminus\{l\}$ are replaced by $c_{l'}^{(n-1)}$. Note that since we are working with forms (c) or (d) we have already replaced many instances of $c_l^{(n)}$ with $c_l^{(n-1)}$ relative to using (a) or (b). Then, using the intuition of Section 8.4.4, replace each remaining $\widetilde{p}_{r,i}(c_l^{(n)}, \tau_{r,i}^{(n)})$ in $\widetilde{f}_l$ with $w_{r,i}\widetilde{p}_l(c_l^{(n)}, \tau_l^{(n)})$ where

$$w_{r,i} = \frac{\widehat{p}_{r,i}(\boldsymbol{c})}{\widehat{p}_{l_i^{(r)}}(\boldsymbol{c})}\,,$$

and $\widetilde{p}_l$ utilises a form from Table 8.4. Let $1_l[\cdot] : \mathcal{L} \to \{0,1\}$ be an indicator function which takes a station $l_i^{(r)}$ and returns 1 if $l_i^{(r)} = l$ and 0 otherwise. Similarly, let $\overline{1}_l[\cdot] : \mathcal{L} \to \{0,1\}$ be an indicator function which takes a station $l_i^{(r)}$ and returns 1 if $l_i^{(r)} \neq l$ and 0 otherwise. This results in a collection of functional forms $\{\widetilde{f}_l\}_{l\in\mathcal{L}}$ where

$$\widetilde{f}_l(\boldsymbol{c}^{(n)}, \tau_l^{(n)}) = -\langle \boldsymbol{c}, \boldsymbol{\theta}\rangle + \sum_{r\in\mathcal{R}} \lambda_r \sum_{i=1}^{N_r} \overline{\theta}_{r,i} \prod_{j=1}^{i} \Big(1 - \mathrm{I}_l[l_j^{(r)}] w_{r,j} \widetilde{p}_l\big(c_l^{(n)}, \tau_l^{(n)}\big) - \overline{\mathrm{I}}_l[l_j^{(r)}] \widehat{p}_{r,j}\big(\boldsymbol{c}^{(n-1)}\big)\Big)\,,$$

meaning

$$\frac{\partial \widetilde{f}_l}{\partial c_l^{(n)}}(\boldsymbol{c}^{(n)}, \tau_l) = -\theta_l - \frac{\mathrm{d}\widetilde{p}_l(c_l^{(n)}, \tau_l^{(n)})}{\mathrm{d}c_l^{(n)}} \sum_{r\in\mathcal{R}} \lambda_r \sum_{i=1}^{N_r} \overline{\theta}_{r,i} \prod_{j=1}^{i} \left(1_l[l_j^{(r)}] w_{r,j} + \overline{1}_l[l_j^{(r)}]\Big(1 - \widehat{p}_{r,j}\big(\boldsymbol{c}^{(n-1)}\big)\Big)\right).$$

Setting this to 0 gives an alternative to numerical optimisation that can be used in Step (III) of our algorithm. This 'per-station' approach is summarised in Algorithm 39.

Algorithm 39. *Per-station functional-form optimisation*
`Initialize:` `Set` $n = 1$ `and choose` $\boldsymbol{c}^{(0)} \in (0,\infty)^L$, $\varepsilon > 0$, $N \in \mathbb{N}$.

(I) `For` $r \in \mathcal{R}$ `for` $i = 1, \ldots, N_r$ `obtain` $\widehat{p}_{r,i}(\boldsymbol{c}^{(n-1)})$ `for` $i = 1, \ldots, N_r$. `For` $l \in \mathcal{L}$ `obtain` $\widehat{p}_l(\boldsymbol{c}^{(n-1)})$.

(II) `For` $l \in \mathcal{L}$ `compute` $\tau_l^{(n)}$ `by solving`

$$\widehat{p}_l(\boldsymbol{c}^{(n-1)}) = \widetilde{p}_l(\boldsymbol{c}^{(n-1)}, \tau_l)\,.$$

(III) `For` $l \in \mathcal{L}$ `solve for` $c_l^{(n)}$:

$$\frac{\mathrm{d}\widetilde{p}_l(c_l^{(n)}, \tau_l^{(n)})}{\mathrm{d}c_l} + \theta_l \left[\sum_{r\in\mathcal{R}} \lambda_r \sum_{i=1}^{N_r} \overline{\theta}_{r,i} \prod_{j=1}^{i} \left(1_l[l_j^{(r)}] w_{r,j} + \overline{1}_l[l_j^{(r)}]\Big(1 - \widehat{p}_{r,j}\big(\boldsymbol{c}^{(n-1)}\big)\Big)\right)\right]^{-1} = 0\,. \tag{8.43}$$

`If resulting` $c_l^{(n)} < \overline{c}_l$ `for any` l, `then set` $c_l^{(n)} = \overline{c}_l$.

(IV) `If` $||\boldsymbol{c}^{(n-1)} - \boldsymbol{c}^{(n)}|| < \varepsilon$ `or` $n > N$: `output` $\boldsymbol{c}^{(n)}$.
`Else:` `set` $n = n+1$ `and return to` (I).

Note that the second term in (8.43) is just a constant, and so as we did for the tandem network using the modified Weibull form with $k = 2$ in (8.34) and (8.35), it is possible to

Table 8.4: Functional forms and associated tuning parameters for Algorithm 39.

	Name	$\widetilde{p}_l(c_l^{(n)}, \tau_l^{(n)})$	$\tau_l^{(n)}\big(\widehat{p}_l(\boldsymbol{c}^{(n-1)})\big)$
(c)	Modified ratio	$\int_{c_l^{(n)}}^0 k{c_l^{(n-1)}}^{k-1}{\tau_l^{(n)}}^k \Big(1+(\tau_l^{(n)}x)^k\Big)^{-2}\,\mathrm{d}x$	$\frac{1}{c_l^{(n-1)}}\Big(\frac{1-\widehat{p}_l(\boldsymbol{c}^{(n-1)})}{\widehat{p}_l(\boldsymbol{c}^{(n-1)})}\Big)^{1/k}$
(d)	Modified Weibull	$\int_{c_l^{(n)}}^0 k{c_l^{(n-1)}}^{k-1}{\tau_l^{(n)}}^k \exp\Big(-(\tau_l^{(n)}x)^k\Big)\,\mathrm{d}x$	$\frac{1}{c_l^{(n-1)}}\Big(-\log(\widehat{p}_l(\boldsymbol{c}^{(n-1)}))\Big)^{1/k}$

solve (8.43) explicitly for the functional forms (c) and (d) as given in Table 8.4. For the modified ratio form this is

$$c_l^{(n)} = \frac{1}{\tau_l^{(n)}}\left(-1+\sqrt{\frac{\gamma_l k {c_l^{(n-1)}}^{k-1}{\tau_l^{(n)}}^k}{\theta_l}}\right)^{1/k} \vee \overline{c}_l\,,$$

and for the modified Weibull form this is

$$c_l^{(n)} = \frac{1}{\tau_l^{(n)}}\left(\log\left(\frac{\gamma_l k {c_l^{(n-1)}}^{k-1}{\tau_l^{(n)}}^k}{\theta_l}\right)\right)^{1/k} \vee \overline{c}_l\,,$$

where in both cases

$$\gamma_l := \sum_{r\in\mathcal{R}}\lambda_r\sum_{i=1}^{N_r}\overline{\theta}_{r,i}\prod_{j=1}^{i}\Big(1_l[l_j^{(r)}]w_{r,j}+\overline{1}_l[l_j^{(r)}]\Big(1-\widehat{p}_{r,j}\big(\boldsymbol{c}^{(n-1)}\big)\Big)\Big)\,.$$

8.6 Numerical experiments

In this section we evaluate the performance of Algorithm 38 and Algorithm 39 on Example 4 model given in Section 8.3.2. In all of the experiments the inter-arrival times and service times are generated from two-stage Coxian distributions with parameters set and scaled to match a collection of coefficients of variation (CoVs). We consider 100 randomly chosen (unique) scenarios where the inter-arrival times have a CoV randomly sampled from $\{0.75, 2, 3.25\}$, service times at stations 1 and 2 are randomly sampled from $\{2, 3.75, 5.5\}$, service times at stations 3 and 4 are randomly sampled from $\{1.5, 3, 4.5\}$, and service times at stations 5 and 6 are always equal to 1.5. The scenarios generated by this process are given in Table A.1 and Table A.2 in Section A.3 (in the appendix).

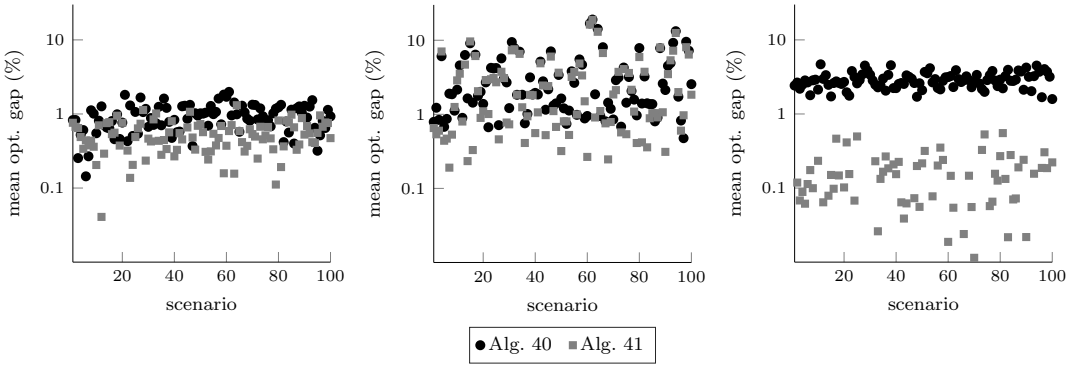
The inter-arrival times for jobs of class i are sampled as $\frac{1}{2\lambda_i}\big(E_1+\frac{1}{q}E_2\,B\big)$, where E_1 and E_2 are independent unit-mean exponentially distributed random variables, $q\in(0,1)$ is a real number, and B is a Bernoulli random variable with parameter q. The parameter $q=\frac{1}{2}v^{-2}$ is set such that CoV of the inter-arrival times matches the desired value v, specified in Table A.1 and Table A.2 for the scenario being studied. For classes $i=1,\ldots,6$ we take $\lambda_i=5$ and v equal to $A1$ as specified in the first column of Table A.1 and Table A.2. For classes $i=1,\ldots,6$ we take $\lambda_i=2.5$ and v equal to $A2$ as specified in the second column of Table A.1 and Table A.2. Service times are specified on a per-station basis. The service times of all jobs at station i are sampled as $\frac{3}{8}\big(E_1+\frac{1}{q}E_2\,B\big)$, where E_1,

E_2, B, and q are as before. For station i the CoV is set to match the value given in the $(2+i)$-th column of Table A.1 and Table A.2. Capacity costs θ_i are assumed unitary for all stations. The amount of revenue generated by a successful service $\overline{\theta}_{r,i}$ is equal for all classes of job at any particular station. We consider three different revenue settings, as displayed in Table 8.5.

Table 8.5: Three combinations of revenues assigned to stations.

Revenue setting	Station 1	Station 2	Station 3	Station 4	Station 5	Station 6
Equal	1.25	1.25	1.25	1.25	1.25	1.25
Increasing	0.8	0.8	1.1	1.1	1.5	1.5
Unordered	1.25	1.1	0.8	1.5	3	1.1

Our first goal is to understand how well the algorithms find an approximation to the optimal solution defined in (8.11). In order to find a benchmark we implemented a stochastic approximation algorithm (Algorithm 40 in Section 8.7.2) for each scenario and revenue setting. In Figure 8.7 we display the mean optimality gap from 10 sample paths of our Algorithm 38 and Algorithm 39 for the different scenarios and revenue settings. It can be seen that both algorithms are capable of reliably achieving an optimality gap of less than 10% in most scenario and revenue settings. In the equal revenue setting both algorithms achieve an optimality gap of approximately 1%. In the increasing revenue setting the performance is not quite as good and tends to be across the range 1–10% for both algorithms. In either of these revenue settings it is positive to see that our algorithm which does not require access to a numerical optimisation package (Algorithm 39) does not lose accuracy compared to the algorithm which does have access to a numerical optimisation package (Algorithm 38). In fact, for the unordered weight setting Algorithm 39 has distinctly superior performance to Algorithm 38 in terms of accuracy.

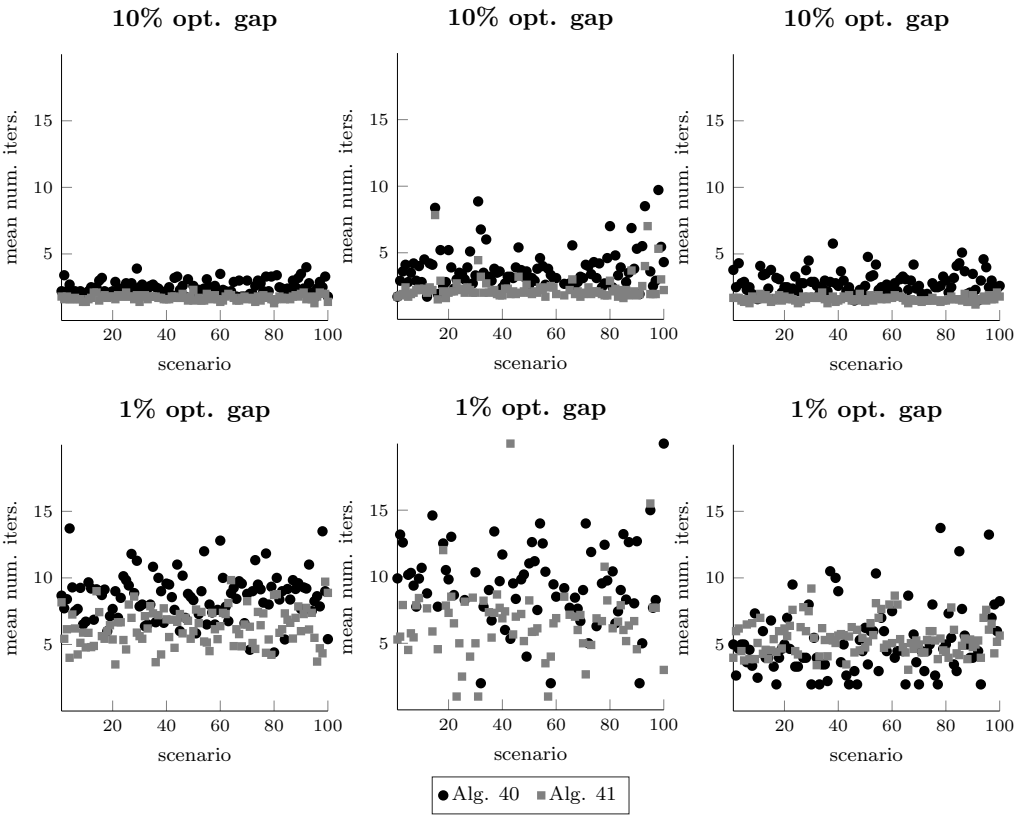


(a) equal revenues (b) increasing revenues (c) unordered revenues

Figure 8.7: Accuracy of Algorithm 38 using form (b) with $k = 2$ and Algorithm 39 using form (d) with $k = 2$ for Example 4 model given in Section 8.3.2 with the scenarios in Table A.1 and Table A.2.

Our second goal is to understand how efficient our algorithms are. Specifically, we would like to know how many iterations the algorithms require to reach a high level of

accuracy. The average optimality gap at our randomly chosen initial conditions over the 3000 experiments we conducted was at least 55% (this is the average optimality gap considering only those experiments where the initial objective function value was positive) and in many cases exceeded 100%. In Figure 8.8 it can be seen that for all scenarios and revenue settings where the optimality gap was reduced below 10%, this was always achieved in less than 5 iterations using Algorithm 39 and usually less than 5 iterations using Algorithm 38. To reduce the optimality gap below 1% the algorithms required approximately 10 iterations. Notably, since our method is derivative free it only requires a single sample of $\widehat{f}$ per iteration. In contrast, stochastic approximation with central finite-difference estimates (as in Algorithm 40) requires 12 samples of $\widehat{f}$ per iteration. The implication is that for many of the scenarios we considered in order to outperform our algorithm stochastic approximation would need to achieve an optimality gap of less than 1% in a single iteration.



(a) equal revenues (b) increasing revenues (c) unordered revenues

Figure 8.8: Efficiency of Algorithm 38 using form (b) with $k = 2$ and Algorithm 39 using form (d) with $k = 2$ for Example 4 model given in Section 8.3.2 with the scenarios in Table A.1 and Table A.2.

8.7 Supporting material

This section contains supporting material for the rest of the chapter which would be cumbersome to include elsewhere.

8.7.1 Matrix derivations

Although evaluation of the objective function in Example 1 of Section 8.3.2 has been possible for quite some time using known methods, the power of the approach described in the previous chapter, based on matrix analytic methods (MAMs), is that far more general systems can be considered. Example 2 given in Section 8.3.2 is such a system. This section uses these methods to explicitly compute our objective for this example, so that it can be used to exactly test our simulation method.

MAMs for Example 2. Here we explain how to obtain explicit expressions for each of the revenue terms in (8.14), i.e., for

$$\begin{aligned}\kappa_1 &:= \lambda_1\,\overline{\theta}_{1,1}\,(1-p_{1,1}(c_1))\,,\\ \kappa_2 &:= \lambda_1\,\overline{\theta}_{1,2}\,(1-p_{1,1}(c_1))(1-p_{1,2}(\boldsymbol{c}))\,,\end{aligned}$$

using matrix analytic methods.

Let $X_{1,1}(t)$ and $X_{1,2}(t)$ be the number of jobs being processed by the system at the first and second stations respectively. Following the previous chapter, we observe that this description leads to a marked MAP (see e.g. [118, Section 2.5] for details), $\big(N_1(t),\, N_2(t),\, t\in\mathbb{R}_0\big)$, that counts the number of jobs successfully entering the first and second stations as follows. Upon an arrival to the first station, as long as $X_{1,1} < c_1$, then $X_{1,1}$ jumps up by 1 and so does N_1. Upon a successful service completion at the first station, if $X_{1,2} < c_2$ then $X_{1,2}$ jumps up by 1 and so too does N_2. It can be seen that the process $(X_{1,2}, X_{1,2})$ is a background process for an encompassing MAP that experiences arrivals when jobs enter stations. In order to find κ_1 and κ_2 explicitly we need to carefully construct the structure of the Markov chain $(X_{1,1}, X_{1,2})$, specify its relationship to N_1 and N_2, and then apply Theorem 1 from [120].

In order to obtain expressions for κ_1 and κ_2 we need to define several matrices. Let:

- $\mathbf{1}_k$ to be a k-tuple containing all unit entries.
- I_k be a $k\times k$ identity matrix.
- $\overline{I}_k$ be a $k\times k$ matrix with upper diagonal containing unit entries and otherwise 0.
- $\underline{I}_k$ be a $k\times k$ matrix with lower diagonal containing unit entries and otherwise 0.
- Q_1 be a $(c_1+1)\times(c_1+1)$ matrix with non-zero entries

$$\begin{aligned}(Q_1)_{i,i+1} &= \lambda_1\,, && \text{for } i=1,\ldots,c_1\,,\\ (Q_1)_{i+1,i} &= (i+1)\mu_1\,, && \text{for } i=1,\ldots,c_1\,,\\ (Q_1)_{i,i} &= -\big(\lambda_1+(i-1)\mu_1\big)\,, && \text{for } i=1,\ldots,c_1\,,\\ (Q_1)_{c_1+1,c_1+1} &= -c_1\mu_1\,.\end{aligned}$$

- Q_2 be a $(c_2+1)\times(c_2+1)$ matrix with non-zero entries

$$\begin{aligned}
(Q_2)_{i,i+1} &= \lambda_2\,, && \text{for } i=1,\ldots,c_2\,,\\
(Q_2)_{i+1,i} &= (i+1)\mu_2\,, && \text{for } i=1,\ldots,c_2\,,\\
(Q_2)_{i,i} &= -\big(\lambda_2+(i-1)\mu_2\big)\,, && \text{for } i=1,\ldots,c_2\,,\\
(Q_2)_{c_2+1,c_2+1} &= -c_2\mu_2\,.
\end{aligned}$$

- $\overline{Q}_1$ be a $(c_1+1)\times(c_1+1)$ matrix with non-zero entries

$$(Q_1)_{i,i} = -(i-1)\mu_1\,, \qquad \text{for } i=1,\ldots,c_1\,.$$

- $\overline{Q}_2$ be a $(c_2+1)\times(c_2+1)$ matrix with non-zero entries

$$(Q_2)_{i,i} = -(i-1)\mu_2\,, \qquad \text{for } i=1,\ldots,c_2\,.$$

Then, use these to define

$$\begin{aligned}
D &= (I_{c_2+1}\otimes Q_1)+(Q_2\otimes I_{c_1+1})\,,\\
D_{1,1} &= \overline{I}_{c_2+1}\otimes\overline{Q}_1\,,\\
D_{1,2} &= \overline{Q}_2\otimes\underline{I}_{c_1+1}\,.
\end{aligned}$$

Take $\boldsymbol{\pi}$ to be the stationary distribution of the Markov chain generated by infinitesimal generator D, i.e. $\boldsymbol{\pi}D=0$ and $\boldsymbol{\pi}\mathbf{1}_{(c_1+1)(c_2+1)}=1$. We then obtain an expression for the expected number of successfully processed jobs at each station using [120, Theorem 1]:

$$\begin{aligned}
\mathbb{E}N_1(t) &= \boldsymbol{\pi}D_{1,1}\mathbf{1}_{(c_1+1)(c_2+1)}\,t+\mathrm{o}(t)\,,\\
\mathbb{E}N_2(t) &= \boldsymbol{\pi}D_{1,2}\mathbf{1}_{(c_1+1)(c_2+1)}\,t+\mathrm{o}(t)\,,
\end{aligned}$$

where $\mathrm{o}(t)\to 0$ is a function $h(t)$ such that $h(t)/t\to 0$ as $t\to\infty$. The quantities of interest κ_1 and κ_2 follow immediately from differentiation of this expression and multiplication by the appropriate θ term. Based on this we can explicitly compute (8.14) for our tandem network of loss systems.

8.7.2 Stochastic approximation implementation

For $\mathcal{C}\subset\mathbb{R}^d$ let $\Pi_{\mathcal{C}}$ be a function $\mathbb{R}^d\to\mathcal{C}$ which for $\boldsymbol{x}\in\mathbb{R}^d$ returns $\boldsymbol{c}\in\mathcal{C}$ which minimises $||\boldsymbol{c}-\boldsymbol{x}||$. Let $\mathbf{e}_l$ be a vector with all components 0 except for component l, which is unitary. Let $\{\delta^{(n)}\}_{n\in\mathbb{N}}$ and $\{\alpha^{(n)}\}_{n\in\mathbb{N}}$ be sequences satisfying

$$\sum_{n=1}^{\infty}\alpha^{(n)}=\infty\,,\quad \sum_{n=1}^{\infty}\alpha^{(n)}\delta^{(n)}<\infty\,,\quad \sum_{n=1}^{\infty}{\alpha^{(n)}}^2{\delta^{(n)}}^{-2}<\infty\,. \tag{8.44}$$

Then if $\widehat{f}$ are uniformly bounded random variables, the following is a convergent (in probability) stochastic approximation algorithm [174].

Algorithm 40. *Stochastic approximation*
`Initialize:` `Set` $n=1$ `and choose` $\boldsymbol{c}^{(0)}\in(0,\infty)^L$.

(I) For $l = 1, \ldots, L$ determine an estimate of the l-th component of the Jacobian:

$$\left(\nabla_{\boldsymbol{c}} \widehat{f}(\boldsymbol{c}^{(n-1)})\right)_l = \frac{\widehat{f}(\boldsymbol{c}^{(n-1)} + \delta^{(n)}\mathbf{e}_l) - \widehat{f}(\boldsymbol{c}^{(n-1)} - \delta^{(n)}\mathbf{e}_l)}{2\delta^{(n)}}.$$

(III) Choose a step size $\alpha^{(n)}$.

(IV) Set

$$\boldsymbol{c}^{(n)} = \Pi_{\mathcal{C}}\left(\boldsymbol{c}^{(n-1)} + \alpha^{(n)} \nabla_{\boldsymbol{c}} \widehat{f}(\boldsymbol{c}^{(n-1)})\right).$$

(V) If $||\widehat{f}(\boldsymbol{c}^{(n-1)}) - \widehat{f}(\boldsymbol{c}^{(n)})|| < \varepsilon$ or $n \geq N$: output $\boldsymbol{c}^{(n)}$.
Else: set $n = n + 1$ and return to (I).

In each iteration of this algorithm the step size $\alpha^{(n)}$ can be determined using the following backtracking line-search method. First choose $\beta > 0$, $\varrho_1 \in (0, 1)$, $\varrho_2 \in (0, 1)$, set $\alpha^{(n)} = \beta^{(n)}$ and $d = 1$, and then:

(I) Determine $F = \widehat{f}\left(\Pi_{\mathcal{C}}\left(\boldsymbol{c}^{(n-1)} + \alpha^{(n)} \nabla_{\boldsymbol{c}} \widehat{f}(\boldsymbol{c}^{(n-1)})\right)\right)$.

(II) If $F \geq \widehat{f}(\boldsymbol{c}^{(n-1)}) + \varrho_2 \alpha^{(n)} \nabla_{\boldsymbol{c}} \widehat{f}(\boldsymbol{c}^{(n-1)}$ or $d \geq D$: output $\alpha^{(n)}$.
Else: set $\alpha^{(n)} = \varrho_1 \alpha^{(n)}$ and $d = d + 1$, and return to (I).

Choosing $\beta^{(n)} = \beta/n$ where $\beta \in \mathbb{R}_+$ is a positive real number and $\delta^{(n)} = n^{-1/3}$ ensures that (8.44) holds and the algorithm therefore converges to the true optimiser with probability 1.

8.8 Outlook

The methodology used in this chapter could be used in a variety of other settings. Due to its low computational burden it could, for example, be extended to answer transient capacity allocation decisions as studied in the previous two chapters. It would be interesting to see if the functional forms that we found work well in an equilibrium setting carry over to transient settings.

Proving that our algorithm converges to a capacity allocation with a small optimality gap remains a challenge. Although it is not obvious to us how this could be done, development of an algorithm that uses a functional form (to be fast) and can provably converge to the actual optimiser (rather than an approximation) would be ideal.

Bibliography

[1] J. F. C. Kingman. *Poisson processes*, volume 3. Clarendon Press, 1992.

[2] F. Kelly. *Reversibility and stochastic networks*. Wiley, 1979.

[3] W. Feller. Die grundlagen der volterraschen theorie des kampfes ums dasein in wahrscheinlichkeitstheoretischer behandlung. *Acta Biotheoretica*, 5(1):11–40, 1939.

[4] J. D. C. Little. A proof for the queuing formula: $L = \lambda W$. *Operations Research*, 9(3):383–387, 1961.

[5] J. D. C. Little. Little's law as viewed on its 50th anniversary. *Operations Research*, 59(3):536–549, 2011.

[6] J. Kingman. Markov population processes. *Journal of Applied Probability*, 6(01):1–18, 1969.

[7] J. R. Jackson. Networks of waiting lines. *Operations Research*, 5(4):518–521, 1957.

[8] P. Whittle. Equilibrium distributions for an open migration process. *Journal of Applied Probability*, 5(03):567–571, 1968.

[9] L. Kleinrock. *Communication nets: stochastic message flow and delay*. Dover Publications, 1964.

[10] P. K. Pollett. Optimal capacity assignment in general queueing networks. In *Optimization: Structure and Applications*, 261–272. Springer, 2009.

[11] A. Dieker, S. Ghosh, and M. S. Squillante. Optimal resource capacity management for stochastic networks. *Operations Research*, 65(1):221–241, 2016.

[12] G. Grimmett and D. Stirzaker. *Probability and random processes*. Oxford University Press, 2001.

[13] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936–1948, 1992.

[14] R. Srikant and L. Ying. *Communication networks: an optimization, control, and stochastic networks perspective*. Cambridge University Press, 2013.

[15] A. L. Stolyar. Maximizing queueing network utility subject to stability: Greedy primal-dual algorithm. *Queueing Systems*, 50(4):401–457, 2005.

[16] B. Li and R. Srikant. Queue-proportional rate allocation with per-link information in multihop wireless networks. *Queueing Systems*, 83(3-4):329–359, 2016.

[17] A. Rybko and A. L. Stolyar. Ergodicity of stochastic processes describing the operation of open queueing networks. *Problemy Peredachi Informatsii*, 28(3):3–26, 1993.

[18] S. H. Lu and P. R. Kumar. Distributed scheduling based on due dates and buffer priorities. *Automatic Control, IEEE Transactions on*, 36(12):1406–1416, 1991.

[19] J. G. Dai and G. Weiss. Stability and instability of fluid models for reentrant lines. *Mathematics of Operations Research*, 21(1):115–134, 1996.

[20] P. Robert. *Stochastic networks and queues*. Springer Science & Business Media, 2003.

[21] J. Dai. On positive Harris recurrence of multiclass queueing networks: A unified approach via fluid limit models. *The Annals of Applied Probability*, 5(1):49–77, 1995.

[22] A. P. Kovalevskii, V. A. Topchii, and S. G. Foss. On the stability of a queueing system with uncountably branching fluid limits. *Problems of Information Transmission*, 41(3):254–279, 2005.

[23] M. Remerova. *Fluid limit approximations of stochastic networks*. PhD thesis, Vrije Universiteit Amsterdam, 2014.

[24] A. D. Barbour, P. Chigansky, and F. C. Klebaner. On the emergence of random initial conditions in fluid limits. *Journal of Applied Probability*, 53(4):1193–1205, 2016.

[25] J. G. Dai. A fluid limit model criterion for instability of multiclass queueing networks. *The Annals of Applied Probability*, 6(3):751–757, 1996.

[26] S. P. Meyn. Transience of multiclass queueing networks via fluid limit models. *The Annals of Applied Probability*, 5(4):946–957, 1995.

[27] H. C. Gromoll, P. Robert, and B. Zwart. Fluid limits for processor-sharing queues with impatience. *Mathematics of Operations Research*, 33(2):375–402, 2008.

[28] M. Remerova, J. Reed, and B. Zwart. Fluid limits for bandwidth-sharing networks with rate constraints. *Mathematics of Operations Research*, 39(3):746–774, 2014.

[29] J. Zhang, J. G. Dai, and B. Zwart. Law of large number limits of limited processor-sharing queues. *Mathematics of Operations Research*, 34(4):937–970, 2009.

[30] M. Frolkova, S. Foss, and B. Zwart. Fluid limits for an aloha-type model with impatient customers. *Queueing Systems*, 72(1-2):69–101, 2012.

[31] S. N. Ethier and T. G. Kurtz. *Markov processes: characterization and convergence*. John Wiley & Sons, 2009.

[32] D. Fiems, M. Mandjes, and B. Patch. Networks of infinite-server queues with multiplicative transitions. *Performance Evaluation*, 2018.

[33] M. Dobrzyński and F. Bruggeman. Elongation dynamics shape bursty transcription and translation. *Proceedings of the National Academy of Sciences*, 106(8):2583–2588, 2009.

[34] R. Serfozo. *Introduction to stochastic networks*, volume 44. Springer Science & Business Media, 2012.

[35] R. J. Boucherie and N. M. Van Dijk. Product forms for queueing networks with state-dependent multiple job transitions. *Advances in Applied Probability*, 23(1):152–187, 1991.

[36] A. Coyle, W. Henderson, C. E. Pearce, and P. G. Taylor. Mean-value analysis for a class of petri nets and batch-movement queueing networks with product-form equilibrium distributions. *Mathematical and Computer Modelling*, 22(10-12):27–34, 1995.

[37] W. Henderson, C. E. M. Pearce, P. G. Taylor, and N. M. van Dijk. Closed queueing networks with batch services. *Queueing Systems*, 6(1):59–70, 1990.

[38] W. Henderson and P. G. Taylor. Product form in networks of queues with batch arrivals and batch services. *Queueing Systems*, 6(1):71–87, 1990.

[39] Y. I. Mitrofanov, E. S. Rogachko, and E. P. Stankevich. Analysis of queueing networks with batch movements of customers and control of flows among clusters. *Automatic Control and Computer Sciences*, 49(4):221–230, 2015.

[40] H. Yamashita and M. Miyazawa. Geometric product form queueing networks with concurrent batch movements. *Advances in Applied Probability*, 30(4):1111–1129, 1998.

[41] A. Economou. Generalized product-form stationary distributions for Markov chains in random environments with queueing applications. *Advances in Applied Probability*, 37(1):185–211, 2005.

[42] G. Falin. The M/M/∞ queue in a random environment. *Queueing Systems*, 58(1):65–76, 2008.

[43] R. Krenzler, H. Daduna, and S. Otten. Jackson networks in nonautonomous random environments. *Advances in Applied Probability*, 48(2):315–331, 2016.

[44] S. Kapodistria, T. Phung-Duc, and J. Resing. Linear birth/immigration-death process with binomial catastrophes. *Probability in the Engineering and Informational Sciences*, 30(1):79–111, 2016.

[45] A. Economou and D. Fakinos. Alternative approaches for the transient analysis of Markov chains with catastrophes. *Journal of Statistical Theory and Practice*, 2(2):183–197, 2008.

[46] R. J. Swift. Transient probabilities for a simple birth-death-immigration process under the influence of total catastrophes. *International Journal of Mathematics and Mathematical Sciences*, 25(10):689–692, 2001.

[47] P. Holme and J. Saramäki. Temporal networks. *Physics Reports*, 519(3):97–125, 2012.

[48] P. Holme. Modern temporal network theory: a colloquium. *The European Physical Journal B*, 88(9):234, 2015.

[49] M. Mandjes, N. Starreveld, and R. Bekker. Queues on a dynamically evolving graph. *Journal of Statistical Physics*, 1–25, 2017.

[50] M. Mandjes, N. J. Starreveld, R. Bekker, and P. Spreij. Dynamic Erdős–Rényi graphs. *Lecture Notes in Computer Science*, 10000, 2018.

[51] X. Zhang, C. Moore, and M. E. Newman. Random graph models for dynamic networks. *The European Physical Journal B*, 90(10):200, 2017.

[52] M. Mandjes and K. De Turck. Markov-modulated infinite-server queues driven by a common background process. *Stochastic Models*, 32(2):206–232, 2016.

[53] S. Asmussen. *Applied probability and queues*. Springer–Verlag, 2nd edition, 2003.

[54] D. Bernstein. *Matrix mathematics*. Princeton University Press, 2009.

[55] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003.

[56] C. Van Loan. Computing integrals involving the matrix exponential. *IEEE transactions on automatic control*, 23(3):395–404, 1978.

[57] J. R. Artalejo and A. Gómez-Corral. *Retrial queueing systems: a computational Approach*. Springer, 1999.

[58] J. Kurose and K. Ross. *Computer networking*. Benjamin/Cummings, 3rd edition, 2004.

[59] J. Kurose and K. Ross. *Distributed storage: concepts, algorithms, and implementations*. CreateSpace Independent, 2013.

[60] J. Blom, K. De Turck, and M. Mandjes. Functional central limit theorems for Markov-modulated infinite-server systems. *Mathematical Methods of Operations Research*, 83(3):351–372, 2016.

[61] U. Franz, V. Liebscher, and S. Zeiser. Piecewise-deterministic Markov processes as limits of Markov jump processes. *Advances in Applied Probability*, 44(3):729–748, 2012.

[62] A. Shwartz and A. Weiss. *Large deviations for performance analysis: queues, communication and computing*, volume 5. CRC Press, 1995.

[63] N. Walton. Concave switching in single-hop and multihop networks. *Queueing Systems*, 81(2-3):265–299, 2015.

[64] M. Bramson, B. D'Auria, and N. S. Walton. Proportional switching in first-in, first-out networks. *Operations Research*, 65(2):496–513, 2016.

[65] B. Li and R. Srikant. Correction to "queue-proportional rate allocation with per-link information in multihop wireless networks". *Queueing Systems*, 85(3-4):383–385, 2017.

[66] A. L. Stolyar et al. Maxweight scheduling in a generalized switch: state space collapse and workload minimization in heavy traffic. *The Annals of Applied Probability*, 14(1):1–53, 2004.

[67] M. Andrews, K. Kumaran, K. Ramanan, A. Stolyar, R. Vijayakumar, and P. Whiting. Scheduling in a queuing system with asynchronously varying service rates. *Probability in the Engineering and Informational Sciences*, 18(2):191–217, 2004.

[68] J. Dai and B. Prabhakar. The throughput of data switches with and without speedup. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, 556–564. IEEE, 2000.

[69] X. Lin and S. B. Rasool. Constant-time distributed scheduling policies for ad hoc wireless networks. *IEEE Transactions on Automatic Control*, 54(2):231–242, 2009.

[70] M. Bramson. State space collapse with application to heavy traffic limits for multiclass queueing networks. *Queueing Systems*, 30(1-2):89–140, 1998.

[71] R. J. Williams. Diffusion approximations for open multiclass queueing networks: sufficient conditions involving state space collapse. *Queueing systems*, 30(1-2):27–88, 1998.

[72] P. Billingsley. *Convergence of probability measures*. John Wiley & Sons, 2 edition, 1999.

[73] Y. S. Chow. On a strong law of large numbers for martingales. *Ann. Math. Statist.*, 38(2):610, 1967.

[74] M. Mandjes, B. Patch, and N. Walton. Detecting Markov chain instability: a monte carlo approach. *Stochastic Systems*, 7(2):289–314, 2017.

[75] J. R. Jackson. Jobshoplike- queueing systems. *Management Science*, 10(1):131–142, 1963.

[76] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios. Open, closed, and mixed networks of queues with different classes of customers. *Journal of the ACM*, 22(2):248–260, 1975.

[77] F. P. Kelly. Networks of queues with customers of different types. *Journal of Applied Probability*, 12(3):542–554, 1975.

[78] P. R. Kumar and T. I. Seidman. Dynamic instabilities and stabilization methods in distributed real-time scheduling of manufacturing systems. *Automatic Control, IEEE Transactions on*, 35(3):289–298, 1990.

[79] M. Bramson. Instability of FIFO queueing networks. *The Annals of Applied Probability*, 4(2):414–431, 1994.

[80] C. Bordenave, D. McDonald, and A. Proutière. Asymptotic stability region of slotted ALOHA. *Information Theory, IEEE Transactions on*, 58(9):5841–5855, 2012.

[81] I. MacPhee, M. Menshikov, D. Petritis, and S. Popov. A Markov chain model of a polling system with parameter regeneration. *The Annals of Applied Probability*, 17(5/6):1447–1473, 2007.

[82] F. Baccelli and T. Bonald. Window flow control in FIFO networks with cross traffic. *Queueing Systems*, 32(1):195–231, 1999.

[83] Y. Nazarathy, T. Taimre, A. Asanjarani, J. Kuhn, B. Patch, and A. Vuorinen. The challenge of stabilizing control for queueing systems with unobservable server states. In *Control Conference (AUCC), 2015 5th Australian*, 342–347, Gold Coast, 2015. Engineers Australia.

[84] M. Bramson. *Stability of queueing networks*. Springer Berlin Heidelberg, New York, 2008.

[85] J. R. Wieland, R. Pasupathy, and B. W. Schmeiser. Queueing network simulation analysis: queueing-network stability: simulation-based checking. In *Proceedings of the 35th conference on Winter simulation: driving innovation*, 520–527. Winter Simulation Conference, 2003.

[86] H. Leahu and M. Mandjes. A numerical approach to stability of multiclass queueing networks. *(preprint)*, 2016.

[87] S. Kirkpatrick and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[88] L. Tassiulas and A. Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *Information Theory, IEEE Transactions on*, 39(2):466–478, 1993.

[89] M. Andrews and L. Zhang. Achieving stability in networks of input-queued switches. *Networking, IEEE/ACM Transactions on*, 11(5):848–857, 2003.

[90] J. Ghaderi, S. Borst, and P. Whiting. Queue-based random-access algorithms: fluid limits and stability issues. *Stochastic Systems*, 4(1):81–156, 2014.

[91] M. Hairer. Convergence of Markov processes. *Lecture notes*, 2010. Available at http://www.hairer.org/notes/Convergence.pdf.

[92] S. Meyn and R. Tweedie. *Markov chains and stochastic stability*. Communications and Control Engineering. Springer London, 2012.

[93] D. Williams. *Probability with martingales*. Cambridge University Press, Cambridge, 1991.

[94] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand. Achieving 100% throughput in an input-queued switch. *Communications, IEEE Transactions on*, 47(8):1260–1267, 1999.

[95] F. P. Kelly. Loss networks. *The Annals of Applied Probability*, 1(3):319–378, 1991.

[96] H. Khazaei, J. Mišić, and V. B. Mišić. Performance analysis of cloud computing centers using M/G/m/m+r queuing systems. *IEEE Transactions on Parallel and Distributed Systems*, 23(5):936–943, 2012.

[97] S. Rahimi-Ghahroodi, A. Al Hanbali, W. Zijm, J. van Ommeren, and A. Sleptchenko. Integrated planning of spare parts and service engineers with partial backlogging. *OR Spectrum*, 1–38, 2017.

[98] P. L. van den Berg, G. A. Legemaate, and R. D. van der Mei. Increasing the responsiveness of firefighter services by relocating base stations in Amsterdam. *Interfaces*, 2017.

[99] E. Brockmeyer, H. L. Halstrom, and A. Jensen. *The life and works of A. K. Erlang.* 1948.

[100] A. K. Erlang. Sandsynlighedsregning og telefonsamtaler. *Nyt tidsskrift for Matematik*, 20:33–39, 1909.

[101] A. K. Erlang. Løsning af nogle problemer fra sandsynlighedsregningen af betydning for de automatiske telefoncentraler. *Elektroteknikeren*, 13, 1917.

[102] R. W. Wolff. Poisson arrivals see time averages. *Operations Research*, 30(2):223–231, 1982.

[103] N. M. van Dijk. On the arrival theorem for communication networks. Computer Networks and ISDN Systems, 25(10):1135–1142, 1993.

[104] P. G. Taylor. Insensitivity in stochastic models. In *Queueing Networks*, 121–140. Springer, 2011.

[105] F. Kelly and E. Yudovina. *Stochastic networks.* Cambridge University Press, 2014.

[106] G. Louth, M. Mitzenmacher, and F. Kelly. Computational complexity of loss networks. *Theoretical Computer Science*, 125(1):45–59, 1994.

[107] F. P. Kelly. Blocking probabilities in large circuit-switched networks. *Advances in Applied Probability*, 18(2):473–505, 1986.

[108] F. P. Kelly. Fixed point models of loss networks. *The ANZIAM Journal*, 31(2):204–218, 1989.

[109] F. P. Kelly. Routing in circuit-switched networks: optimization, shadow prices and decentralization. *Advances in Applied Probability*, 20(1):112–144, 1988.

[110] F. P. Kelly. Routing and capacity allocation in networks with trunk reservation. *Mathematics of Operations Research*, 15(4):771–793, 1990.

[111] P. B. Key. Optimal control and trunk reservation in loss networks. *Probability in the Engineering and Informational Sciences*, 4(2):203–242, 1990.

[112] J. Fu, B. Moran, and P. G. Taylor. Restless bandits in action: resource allocation, competition and reservation. *arXiv preprint arXiv:1804.02100*, 2018.

[113] S. Zachary. Control of stochastic loss networks, with applications. *Journal of the Royal Statistical Society. Series B (Methodological)*, 61–73, 1988.

[114] J. Sanders, S. C. Borst, and J. S. van Leeuwaarden. Online network optimization using product-form Markov processes. *IEEE Transactions on Automatic Control*, 61(7):1838–1853, 2016.

[115] B. A. Chiera and P. G. Taylor. What is a unit of capacity worth? *Probability in the Engineering and Informational Sciences*, 16(4):513–522, 2002.

[116] B. A. Chiera, A. E. Krzesinski, and P. G. Taylor. Some properties of the capacity value function. *SIAM Journal on Applied Mathematics*, 65(4):1407–1419, 2005.

[117] B. Rasof. The initial-and final-value theorems in Laplace transform theory. *Journal of the Franklin Institute*, 274(3):165–177, 1962.

[118] Q. He. *Fundamentals of matrix-analytic methods*. Springer, 2014.

[119] G. Latouche and V. Ramaswami. *Introduction to matrix analytic methods in stochastic modeling*, volume 5. Siam, 1999.

[120] S. Narayana and M. F. Neuts. The first two moment matrices of the counts for the Markovian arrival process. *Communications in Statistics. Stochastic Models*, 8(3):459–477, 1992.

[121] B. Patch, T. Taimre, and Y. Nazarathy. Performance of faulty loss systems with persistent connections. *ACM SIGMETRICS Performance Evaluation Review*, 43(2):16–18, 2015.

[122] Queueing networks. volume 154 of *International Series in Operations Research & Management Science*. 2011.

[123] T. Chihara. *An Introduction to orthogonal polynomials*. Gordon and Breach, New York, 1978.

[124] B. Patch and T. Taimre. Transient provisioning and performance evaluation for cloud computing platforms: a capacity value approach. *Performance Evaluation*, 118:48–62, 2018.

[125] Z. Á. Mann. Allocation of virtual machines in cloud data centers — a survey of problem models and optimization algorithms. *ACM Computing Surveys*, 48(1):11, 2015.

[126] Q. Zhang, M. F. Zhani, S. Zhang, Q. Zhu, R. Boutaba, and J. L. Hellerstein. Dynamic energy-aware capacity provisioning for cloud computing environments. In *Proceedings of the 9th International Conference on Autonomic Computing*, 145–154. ACM, 2012.

[127] A. K. Erlang. Solutions of some problems in the theory of probabilities of significance in automatic telephone exchanges. *The Post Office Electrical Engineers' Journal*, 10:189–197, 1918.

[128] J. Abate and J. Whitt. Calculating transient characteristics of the Erlang loss model by numerical transform inversion. *Communications in Statistics. Stochastic Models.*, 14(3):663–680, 1998.

[129] C. Knessl and J. S. H. van Leeuwaarden. Transient analysis of the Erlang A model. *Mathematical Methods of Operations Research*, 82:143–173, 2015.

[130] M. Mandjes and A. Ridder. A large deviations approach to the transient of the Erlang loss model. *Performance Evaluation*, 43:181–198, 2015.

[131] P. Braunsteins, S. Hautphenne, and P. G. Taylor. The roles of coupling and the deviation matrix in determining the value of capacity in $M/M/1/C$ queues. *Queueing Systems*, 83(1):157–179, 2016.

[132] B. Patch, T. Taimre, and Y. Nazarathy. Performance of faulty loss systems with persistent connections. *ACM SIGMETRICS Performance Evaluation Review*, 43(2):16–18, 2015.

[133] H. Khazaei, J. Mišić, V. B. Mišić, and S. Rashwand. Analysis of a pool management scheme for cloud computing centers. *IEEE Transactions on parallel and distributed systems*, 24(5):849–861, 2013.

[134] L. Breuer. An EM algorithm for batch Markovian arrival processes and its comparison to a simpler estimation procedure. *Annals of Operations Research*, 112:123–138, 2002.

[135] P. Buchholz, P. Kemper, and J. Kriege. Multi-class Markovian arrival processes and their parameter fitting. *Performance Evaluation*, 67:1092–1106, 2010.

[136] P. Buchholz and J. Kriege. A heuristic approach for fitting MAPs to moments and joint moments. In *Quantitative Evaluation of Systems, Sixth International Conference on*, 53–62. IEEE Computer Society, 2009.

[137] P. Buchholz and J. Kriege. Fitting correlated arrival and service times and related queueing performance. *Queueing Systems*, 2017.

[138] G. Casale, E. Z. Zhang, and E. Smirni. KPC-toolbox: Simple yet effective trace fitting using Markovian arrival processes. In *Quantitative Evaluation of Systems, 2008. QEST'08. Fifth International Conference on*, 83–92. IEEE, 2008.

[139] Amazon auto-scaling. `https://aws.amazon.com/autoscaling`, 2017.

[140] R. B. Sidje. Expokit: a software package for computing matrix exponentials. *ACM Trans. Math. Softw.*, 24(1):130–156, Mar. 1998.

[141] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix. *SIAM Review*, 20(4):801–836, 1978.

[142] T. Atmaca, T. Begin, A. Brandwajn, and H. Castel-Taleb. Performance evaluation of cloud computing centers with general arrivals and service. *Parallel and Distributed Systems, IEEE Transactions on*, 2015.

[143] S. Zhang, Z. Qian, Z. Luo, J. Wu, and S. Lu. Burstiness-aware resource reservation for server consolidation in computing clouds. *IEEE Transactions on Parallel and Distributed Systems*, 27(4):964–977, 2016.

[144] H. Khazaei, J. Mišić, and V. B. Mišić. Performance analysis of cloud centers under burst arrivals and total rejection policy. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, 1–6. IEEE, 2011.

[145] H. Khazaei, J. Mišić, and V. B. Mišić. Performance of cloud centers with high degree of virtualization under batch task arrivals. *IEEE Transactions on Parallel and Distributed Systems*, 24(12):2429–2438, 2013.

[146] Y. Tan and C. H. Xia. An adaptive learning approach for efficient resource provisioning in cloud services. *ACM Sigmetrics Performance Evaluation Review*, 42(4):3–11, 2015.

[147] D. Bruneo. A stochastic model to investigate data center performance and QoS in IaaS cloud computing systems. *Parallel and Distributed Systems, IEEE Transactions on*, 25(3):560–569, 2014.

[148] V. J. Maccio and D. G. Down. On optimal policies for energy-aware servers. *Performance Evaluation*, 90:36–52, 2015.

[149] J. S. H. van Leeuwaarden, B. Mathijsen, and F. Sloothaak. Cloud provisioning in the QED regime. In *Proceedings of the 9th EAI International Conference on Performance Evaluation Methodologies and Tools*, 180–187. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2016.

[150] M. Harchol-Balter. *Performance modeling and design of computer systems: queueing theory in action*. Cambridge University Press, 2013.

[151] R. Ghosh, F. Longo, V. K. Naik, and K. S. Trivedi. Quantifying resiliency of IaaS cloud. In *Reliable Distributed Systems, 2010 29th IEEE Symposium on*, 343–347. IEEE, 2010.

[152] J. G. Kemeny and J. L. Snell. Finite continuous time markov chains. *Theory of Probability & Its Applications*, 6(1):101–105, 1961.

[153] P. Coolen-Schrijner and E. A. Van Doorn. The deviation matrix of a continuous-time markov chain. *Probability in the Engineering and informational Sciences*, 16(3):351–366, 2002.

[154] AberdeenGroup. Service parts management, unlocking value and profits in the service chain. *AberdeenGroup, Boston*, 2003.

[155] Z.-J. M. Shen, C. Coullard, and M. S. Daskin. A joint location-inventory model. *Transportation science*, 37(1):40–55, 2003.

[156] A. Kranenburg and G. Van Houtum. A new partial pooling structure for spare parts networks. *European Journal of Operational Research*, 199(3):908–921, 2009.

[157] L. Brotcorne, G. Laporte, and F. Semet. Ambulance location and relocation models. *European journal of operational research*, 147(3):451–463, 2003.

[158] P. L. van den Berg, J. T. van Essen, and E. J. Harderwijk. Comparison of static ambulance location models. In *Proceedings of Logistics Operations Management (GOL)*. IEEE, 2016.

[159] P. Agrawal, D. K. Anvekar, and B. Narendran. Channel management policies for handovers in cellular networks. *Bell Labs Technical Journal*, 1(2):97–110, 1996.

[160] M. Sidi and D. Starobinski. New call blocking versus handoff blocking in cellular networks. *Wireless networks*, 3(1):15–27, 1997.

[161] R. Kwan, R. Arnott, R. Paterson, R. Trivisonno, and M. Kubota. On mobility load balancing for lte systems. In *Proceedings of IEEE Vehicular Technology Conference (VTC)*, 2010.

[162] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief. A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys & Tutorials*, 19(4):2322–2358, 2017.

[163] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios. Open, closed, and mixed networks of queues with different classes of customers. *Journal of the ACM*, 22(2):248–260, 1975.

[164] J. M. Harrison and R. J. Williams. Brownian models of feedforward queueing networks: Quasireversibility and product form solutions. *The Annals of Applied Probability*, 263–293, 1992.

[165] S. Axsäter. Modelling emergency lateral transshipments in inventory systems. *Management Science*, 36(11):1329–1338, 1990.

[166] M. Restrepo, S. G. Henderson, and H. Topaloglu. Erlang loss models for the static deployment of ambulances. *Health care Management Science*, 12(1):67, 2009.

[167] J. M. Harrison and A. Zeevi. A method for staffing large call centers based on stochastic fluid models. *Manufacturing & Service Operations Management*, 7(1):20–36, 2005.

[168] S. C. Borst, A. Mandelbaum, and M. I. Reiman. Dimensioning large call centers. *Operations Research*, 52(1):17–34, 2004.

[169] R. Hassin, Y. Y. Shaki, and U. Yovel. Optimal service-capacity allocation in a loss system. *Naval Research Logistics (NRL)*, 62(2):81–97, 2015.

[170] L. M. Wein. Capacity allocation in generalized jackson networks. *Operations Research Letters*, 8(3):143–146, 1989.

[171] S. G. Henderson and B. L. Nelson. *Handbooks in operations research and management science: simulation*, volume 13. Elsevier, 2006.

[172] S. Asmussen and P. W. Glynn. *Stochastic simulation: algorithms and analysis*, volume 57. Springer Science & Business Media, 2007.

[173] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22(3):400–407, 1951.

[174] J. Kiefer, J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, 1952.

[175] R. H. Byrd, S. L. Hansen, J. Nocedal, and Y. Singer. A stochastic quasi-newton method for large-scale optimization. *SIAM Journal on Optimization*, 26(2):1008–1031, 2016.

[176] M. Brown and H. Solomon. A second-order approximation for the variance of a renewal reward process. *Stochastic Processes and their Applications*, 3:301–314, 1975.

[177] D. L. Jagerman. Some properties of the Erlang loss function. *Bell Labs Technical Journal*, 53(3):525–551, 1974.

[178] A. Harel. Convexity properties of the Erlang loss formula. *Operations Research*, 38(3):499–505, 1990.

[179] E. Pinsky. A simple approximation for the Erlang loss function. *Performance Evaluation*, 15(3):155–161, 1992.

[180] A. J. E. M. Janssen, J. S. H. Van Leeuwaarden, and B. Zwart. Gaussian expansions and bounds for the Poisson distribution applied to the Erlang B formula. *Advances in Applied Probability*, 40(1):122–143, 2008.

[181] H. Chen and A. Mandelbaum. Leontif systems, RBV's and RBM's. In H. A. Davis and R. J. Elliott, editors, *Applied Stochastic Analysis*, 1–43. Gordon and Breach, New York, 1991.

APPENDIX A

Mathematical miscellany

A.1 One-dimensional Skorohod problem

The following proposition describes standard properties of solutions of the one-dimensional Skorohod problem. The proof of this proposition can be found in [181].

Proposition 41. *Let $w = (w(t),\, t \geq 0)$ be a continuous function in $D([0,\infty),\mathbb{R})$ such that $w(0) \geq 0$. Then the following hold:*

(i) There exists a unique pair (z°, y°) of functions in $D([0,\infty),\overline{\mathbb{R}})$ such that:

(a) $z^\circ(t) = w(t) + y^\circ(t) \geq 0$, $t \geq 0$;

(b) y° is nondecreasing and nonnegative;

(c) $y^\circ(0) = 0$;

(d) for any $t \geq 0$, if $z^\circ(t) > 0$, then t is not a point of increase of y°.

This unique pair is (z, y), where

$$y(t) := -\left[0 \wedge \inf_{0\leq u\leq t} w(u)\right], \quad z(t) = w(t) + y(t), \quad t \geq 0\,.$$

(ii) For any pair of (z°, y°) of functions in $D([0,\infty),\overline{\mathbb{R}})$ satisfying (a) and (b) we have

$$y^\circ(t) \geq y(t)\,, \quad z^\circ(t) \geq z(t)\,, \quad t \geq 0\,.$$

A.2 Martingales and concentration

In Chapter 4 we develop a simulation based algorithm with sample paths that exhibit different random behaviour depending upon whether the system that is used as an input to the algorithm is stable or unstable. In the chapter we use a definition of stability that is closely related to Theorem 2, where we know that except for some finite set the process must have expected downward movement. Key to our analysis is utilising this expected

downward movement to provide an almost sure bound on the behaviour of sample paths generated by the algorithm. This type of bound is known as a *concentration inequality*. One of the simplest and most famous of such results is Markov's inequality, which states that for a non-negative random variable X and some constant $a > 0$ we have

$$\mathbb{P}(X \geq a) \leq \frac{\mathbb{E}X}{a}\,.$$

This result directly relates the deviations of a random variable to its expected value, thus providing information on the concentration of variability about the expected value.

Martingales are a class of random process where a non-trivial bound on the sample path behaviour of the process is readily available, so long as the process has bounded increments. A discrete time index random process $(X(t),\, t \in \mathbb{Z}_+)$ is called a martingale if for any time t

$$\mathbb{E}[|X(t)|] < \infty, \quad \text{and} \tag{A.1}$$
$$\mathbb{E}[X(t+1)\,|\,X_1, \ldots, X(t)] = X(t)\,, \tag{A.2}$$

almost surely. If the equality in (A.2) is replaced by $\leq$, then the process is called a *super-martingale*, which are a more common class of process that often share many of the desirable attributes of martingales. A key observation is if for a process $(Y(t),\, t \in \mathbb{Z}_+)$, we know that $\mathbb{E}[Y(t+1) - Y(t)] = \delta$ where $\delta < \infty$ is some constant, then the process defined by $X(t) - \delta\, t$ is a martingale. If we also know that $|Y(t) - Y(t-1)| < c_t$ almost surely for some constants $\{c_t\}_{t=1}^N$, then this allows us to directly connect the known (or assumed) expected movement of the process to the probability the process exceeds any pre-specified level by its t-th time step. We do this using the Azuma-Hoeffding inequality, which is given in the following theorem.

Theorem 10. *Suppose $(Y(t),\, t \in \mathbb{Z}_+)$ is a martingale (or super-martingale) and $|Y(t) - Y(t-1)| < c_t$ almost surely, then for all positive integers N and all real numbers $a > 0$,*

$$\mathbb{P}\big(Y(t) - Y(0) \geq a\big) \leq \exp\left(\frac{a^2}{2\sum_{t=1}^N c_t^2}\right)\,.$$

In Chapter 4 an assumption of stability allows us to create a supermartingale to which we can apply this theorem and consequently give the distribution of a random process which stochastically dominates stable processes. We then combine this with other martingale methods to obtain our key results. We use the optional stopping theorem, which says that after a random number of time steps (with distribution satisfying certain properties) the expected value of the process is equal to the expected value of the process at its starting location (see e.g. [93, Section 10.1]). We also use a strong law of large numbers for martingales, which says that a martingale $(Y(t),\, t \in \mathbb{Z}_+)$ with bounded second moment, that is $\mathbb{E}Y(t)^2 < \infty$ for all t, increases at a sub-linear rate, meaning $\mathbb{E}Y(t)/t \to 0$ as $t \to \infty$ (see e.g. [93, Section 12.14]).

A.3 Simulation scenarios

Table A.1: Coefficients of variation in the experiments in Section 8.6.

	$A1$	$A2$	$S1$	$S2$	$S3$	$S4$	$S5$	$S6$
1	4.5	4.5	2	2	0.75	0.75	1.5	1.5
2	3	4.5	2	2	3.25	0.75	1.5	1.5
3	4.5	3	2	2	3.25	0.75	1.5	1.5
4	4.5	4.5	2	2	3.25	0.75	1.5	1.5
5	4.5	1.5	2	2	0.75	2	1.5	1.5
6	3	1.5	2	2	2	2	1.5	1.5
7	1.5	3	2	2	0.75	3.25	1.5	1.5
8	1.5	4.5	2	2	0.75	3.25	1.5	1.5
9	4.5	3	2	2	0.75	3.25	1.5	1.5
10	4.5	3	2	2	2	3.25	1.5	1.5
11	3	4.5	3.75	2	0.75	0.75	1.5	1.5
12	3	3	3.75	2	2	0.75	1.5	1.5
13	1.5	4.5	3.75	2	3.25	0.75	1.5	1.5
14	3	3	3.75	2	3.25	0.75	1.5	1.5
15	1.5	1.5	3.75	2	0.75	2	1.5	1.5
16	3	1.5	3.75	2	0.75	2	1.5	1.5
17	4.5	1.5	3.75	2	0.75	2	1.5	1.5
18	1.5	3	3.75	2	2	2	1.5	1.5
19	3	1.5	3.75	2	2	2	1.5	1.5
20	4.5	3	3.75	2	2	2	1.5	1.5
21	1.5	3	3.75	2	0.75	3.25	1.5	1.5
22	4.5	4.5	3.75	2	0.75	3.25	1.5	1.5
23	1.5	3	5.5	2	0.75	0.75	1.5	1.5
24	3	3	5.5	2	0.75	0.75	1.5	1.5
25	4.5	1.5	5.5	2	2	0.75	1.5	1.5
26	3	1.5	5.5	2	3.25	0.75	1.5	1.5
27	3	4.5	5.5	2	3.25	0.75	1.5	1.5
28	4.5	3	5.5	2	3.25	0.75	1.5	1.5
29	1.5	4.5	5.5	2	0.75	2	1.5	1.5
30	3	1.5	5.5	2	2	2	1.5	1.5
31	1.5	3	5.5	2	3.25	2	1.5	1.5
32	1.5	3	5.5	2	2	3.25	1.5	1.5
33	4.5	1.5	5.5	2	2	3.25	1.5	1.5
34	1.5	4.5	2	3.75	0.75	0.75	1.5	1.5
35	3	4.5	2	3.75	0.75	0.75	1.5	1.5
36	3	1.5	2	3.75	2	0.75	1.5	1.5
37	3	3	2	3.75	2	0.75	1.5	1.5
38	4.5	3	2	3.75	0.75	2	1.5	1.5
39	4.5	4.5	2	3.75	0.75	2	1.5	1.5
40	4.5	1.5	2	3.75	2	2	1.5	1.5
41	1.5	3	2	3.75	2	3.25	1.5	1.5
42	3	3	2	3.75	3.25	3.25	1.5	1.5
43	1.5	1.5	3.75	3.75	0.75	0.75	1.5	1.5
44	3	3	3.75	3.75	0.75	0.75	1.5	1.5
45	1.5	1.5	3.75	3.75	3.25	0.75	1.5	1.5
46	3	1.5	3.75	3.75	3.25	0.75	1.5	1.5
47	1.5	4.5	3.75	3.75	0.75	2	1.5	1.5
48	4.5	1.5	3.75	3.75	0.75	2	1.5	1.5
49	3	1.5	3.75	3.75	2	2	1.5	1.5
50	4.5	4.5	3.75	3.75	2	2	1.5	1.5

Table A.2: Coefficients of variation in the experiments in Section 8.6.

	$A1$	$A2$	$S1$	$S2$	$S3$	$S4$	$S5$	$S6$
51	0.75	3.25	3.75	3.75	4.5	3	1.5	1.5
52	0.75	3.25	3.75	3.75	4.5	4.5	1.5	1.5
53	2	3.25	3.75	3.75	3	4.5	1.5	1.5
54	3.25	3.25	3.75	3.75	1.5	4.5	1.5	1.5
55	3.25	3.25	3.75	3.75	3	3	1.5	1.5
56	0.75	0.75	5.5	3.75	4.5	4.5	1.5	1.5
57	2	0.75	5.5	3.75	1.5	1.5	1.5	1.5
58	3.25	0.75	5.5	3.75	3	1.5	1.5	1.5
59	0.75	2	5.5	3.75	4.5	1.5	1.5	1.5
60	2	2	5.5	3.75	4.5	4.5	1.5	1.5
61	2	3.25	5.5	3.75	3	1.5	1.5	1.5
62	2	3.25	5.5	3.75	4.5	4.5	1.5	1.5
63	3.25	3.25	5.5	3.75	1.5	1.5	1.5	1.5
64	3.25	3.25	5.5	3.75	1.5	3	1.5	1.5
65	0.75	0.75	2	5.5	4.5	4.5	1.5	1.5
66	2	0.75	2	5.5	1.5	1.5	1.5	1.5
67	2	0.75	2	5.5	4.5	3	1.5	1.5
68	3.25	0.75	2	5.5	1.5	4.5	1.5	1.5
69	3.25	0.75	2	5.5	3	4.5	1.5	1.5
70	0.75	2	2	5.5	1.5	3	1.5	1.5
71	0.75	2	2	5.5	3	3	1.5	1.5
72	0.75	3.25	2	5.5	1.5	3	1.5	1.5
73	0.75	3.25	2	5.5	3	4.5	1.5	1.5
74	0.75	3.25	2	5.5	4.5	3	1.5	1.5
75	2	3.25	2	5.5	1.5	1.5	1.5	1.5
76	2	3.25	2	5.5	1.5	3	1.5	1.5
77	2	3.25	2	5.5	1.5	4.5	1.5	1.5
78	2	3.25	2	5.5	3	1.5	1.5	1.5
79	3.25	3.25	2	5.5	1.5	1.5	1.5	1.5
80	0.75	0.75	3.75	5.5	1.5	3	1.5	1.5
81	2	0.75	3.75	5.5	1.5	3	1.5	1.5
82	2	0.75	3.75	5.5	3	1.5	1.5	1.5
83	2	0.75	3.75	5.5	4.5	3	1.5	1.5
84	3.25	0.75	3.75	5.5	4.5	4.5	1.5	1.5
85	0.75	2	3.75	5.5	4.5	1.5	1.5	1.5
86	0.75	2	3.75	5.5	4.5	4.5	1.5	1.5
87	2	2	3.75	5.5	1.5	4.5	1.5	1.5
88	3.25	2	3.75	5.5	1.5	1.5	1.5	1.5
89	3.25	2	3.75	5.5	4.5	3	1.5	1.5
90	0.75	0.75	5.5	5.5	1.5	3	1.5	1.5
91	2	0.75	5.5	5.5	4.5	4.5	1.5	1.5
92	3.25	0.75	5.5	5.5	1.5	3	1.5	1.5
93	3.25	0.75	5.5	5.5	1.5	4.5	1.5	1.5
94	0.75	2	5.5	5.5	1.5	4.5	1.5	1.5
95	0.75	2	5.5	5.5	3	4.5	1.5	1.5
96	2	2	5.5	5.5	4.5	4.5	1.5	1.5
97	3.25	2	5.5	5.5	3	3	1.5	1.5
98	3.25	2	5.5	5.5	4.5	1.5	1.5	1.5
99	0.75	3.25	5.5	5.5	3	4.5	1.5	1.5
100	3.25	3.25	5.5	5.5	4.5	1.5	1.5	1.5