THE UNIVERSITY OF CHICAGO

DESIGN AND ANALYSIS OF FLEXIBLE SERVER SYSTEMS

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE UNIVERSITY OF CHICAGO
BOOTH SCHOOL OF BUSINESS
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

BY
GORKEM UNLU

CHICAGO, ILLINOIS
DECEMBER 2023

To Eren.

TABLE OF CONTENTS

LIST OF FIGURES vi

LIST OF TABLES vii

ACKNOWLEDGMENTS viii

ABSTRACT ix

1 INTRODUCTION 1

2 STABILITY AND INSTABILITY OF PARAMETER AGNOSTIC POLICIES IN PARALLEL SERVER SYSTEMS 3
- 2.1 Introduction 3
  - 2.1.1 Discussion of MaxWeight and Other Related Works 6
  - 2.1.2 Definitions and Notation 10
- 2.2 Intersecting Switching Curves 12
  - 2.2.1 Lack of Stationary Distribution for a Markov Chain with One Dimensional State Space 13
  - 2.2.2 Coupling of the X-Model with the One Dimensional Markov Chain . 16
- 2.3 Non-Intersecting Switching Curves 22
  - 2.3.1 Instability Result 22
  - 2.3.2 Stability Result 32
- 2.4 One Switching Curve for Both Servers 38
- 2.5 Discussion and Conclusion 40

3 PROCESS FLEXIBILITY DESIGN FOR PARALLEL SERVER SYSTEMS WITH GENERAL PROCESSING RATES 42
- 3.1 Introduction 42
  - 3.1.1 Literature Review 48
- 3.2 Model, Definitions and Notations 52
- 3.3 Static Planning Problem and Its Dual 55
- 3.4 Sparse Design of the Flexibility Graph 69
- 3.5 Numerical Experiments 81
- 3.6 Discussion and Conclusion 88

4 EXTENSIONS AND RELATED TOPICS 92
- 4.1 Instability Results for Non-Switching Curve Parameter Agnostic Policies . . 92
- 4.2 Connection between the Static Planning Problem and the Transportation Problems 97
  - 4.2.1 Application of Algorithm 2 in Product Form Systems 102
  - 4.2.2 Uniqueness of Solutions in SPP($\mathcal{F}$) 107

REFERENCES 112

LIST OF FIGURES

2.1 Graph of the X-model 5
2.2 Capacity Region of the X-model where $\mu_{11} > \mu_{12}$ and $\mu_{22} > \mu_{21}$ 12
2.3 Constructed One Dimensional CTMC 14
2.4 Partitioning of the State Space of $(\mathbf{Q}(t))_{t\geq 0}$ 17
2.5 The Instability Region for Non-Intersecting Switching Curves According to Proposition 2 27
2.6 Comparison of Sufficient Conditions for Instability from Corollary 2 and Proposition 2 32

3.1 Examples of Flexibility Structures 43
3.2 Processing Rates and Demand Rates on Fully Flexible Graph 46
3.3 Greedy Max-Flow Solutions on Fully Flexible Graph 46
3.4 Graphs for simulated system 1 84
3.5 Long-run Average Costs for Simulated System 1 85
3.6 Relative Difference of Costs of $\mathcal{S}$ and $\mathcal{F}$ for Simulated System 1 86
3.7 Graphs for simulated system 2 87
3.8 Long-run Average Costs for Simulated System 2 88
3.9 Relative Difference of Costs of $\mathcal{S}$ and $\mathcal{F}$ for Simulated System 2 89

4.1 Partitioning of the State Space of $(\mathbf{Q}(t))_{t\geq 0}$ 95

LIST OF TABLES

3.1 Service and heavy traffic arrival rates for simulated system 1 83
3.2 Service and heavy traffic arrival rates for simulated system 2 87

ACKNOWLEDGMENTS

I am deeply grateful to my advisor, Yuan Zhong, for being a profound researcher and teaching me the fundamentals of high-quality research: identifying important questions, valuing simplicity, intuition, and rigor. It goes without saying that this thesis would not have been possible without his enormous support and guidance.

I wish to express my sincere thanks to Rene Caldentey, Amy Ward, and Yehua Wei for graciously agreeing to serve on my committee and for providing astute insights and constructive feedback. They helped me remain mindful of the rooted philosophical questions on top of which this work was built.

I am grateful to Dan Adelman, for teaching me how empirical and theoretical work can complement each other and for equipping me with multidisciplinary skills.

To my cherished friends - Berkin, Mehmet, Gokcem, Aysu, Erdem, Naz, Cagla, and Yueyang - many of whom have been by my side since high school, I offer my gratitude for being the steadfast pillars of sanity throughout this journey.

I am indebted forever to my beloved family - my parents Hatice and Mehmet, as well as my aunts, cousins and grandparents. Their consistent support, spanning eight-hour time zone differences, has carried me to where I am today. From lifting my spirits during gloomy moments through video calls to wholeheartedly celebrating every small triumph, their presence has been my constant motivation.

My deepest gratitude is to my husband, Eren. His role as my sounding board, earnest supporter, and empathetic listener has been indispensable. From being a source of strength during challenging times to joining me in every minor celebration, I owe this thesis to his unwavering presence.

ABSTRACT

We consider problems related to design and analysis of flexible server systems. On the analysis side, we study the stability properties of the X-model under parameter agnostic policies. The X-model is a special case of flexible server systems that carries insights into larger flexible server systems. It consists of two servers and two queues where both servers are capable of serving either queue. It is the smallest flexible server system that contains a cycle, for which the stability question has not been fully answered. We consider this model under parameter agnostic policies, which dictate what queue an idle server will serve. These policies are appealing in real-world scenarios as they solely rely on queue size information, and eliminate the need for knowledge about system parameters. We show that, despite being desirable, such policies can result in instability. Our analysis focuses on parameter agnostic switching curve policies, wherein each server's service decision is determined by a non-decreasing function of queue sizes. We demonstrate that even at relatively low system loads, these policies can lead to instability. We explore various classes of parameter agnostic policies and characterize sufficient conditions for instability.

We then focus on the design problem on flexible server systems, where the goal is to construct a sparse server-buffer compatibility graph without compromising performance. Researchers have proposed various flexibility structures to tackle the design problem, and validated the proposed designs by theoretical justifications. However, these structures typically rely on a restrictive flow conservation assumption, where exactly one unit of processing capacity is required for one unit of a job. We, on the other hand, relax the flow conservation assumption and allow that arbitrary processing capacities may be required to process one unit of a job. We show that in systems with general processing rates, there exists a sparse flexibility structure with $O(m + n)$ arcs that can achieve good performance in heavy traffic, where m is the number of servers and n is the number of buffers; and we introduce an algorithm that constructs the said flexibility structure. We justify the performance of

our proposed design via numerical experiments. We highlight the critical differences in the analysis of such systems compared to systems where flow conservation assumption holds, and we discuss the connection between the flexibility structure design problem and the transportation problems.

CHAPTER 1

INTRODUCTION

Flexible server systems, also known as parallel server systems, are used to model settings where the servers have varying degrees of flexibility in processing jobs. They consist of multiple non-identical servers working in parallel to process different classes of jobs. Even though these systems are central to the operations of many companies, they are not fully understood. Both the design and analysis problems on flexible server systems have drawn considerable attention over the years. Design problem is typically concerned with constructing an effective compatibility graph between the servers and and buffers. The goal in this stream of research is to achieve system performance akin to a scenario where all servers can cater to all buffers, while employing considerably fewer connections within the compatibility graph. Analysis problem, on the other hand, is concerned with assessing the performance of flexible server systems, where the compatibility graph is given. Assessing the performance is considered under numerous policies and many different system designs. In this thesis, we address the design and analysis problems in certain settings.

In Chapter 2, we focus on the analysis of parallel server systems. Given a special design, the so-called X-model, we determine whether this model is maximally stable under parameter agnostic policies. Parameter agnostic policies designate which queue an idle server will serve without depending on any of the system parameters like arrival and service rates, which make them desirable in practice. If they are to be used in practice, their performance measures like queue lengths, wait times or certain holding costs become of interest. However, stability is a strict prerequisite for all these performance measures. If the system is not stable, all these performance measures diverge to infinity. Therefore, it is worthwhile and necessary to understand stability properties of any policy prior to conducting performance analysis. Here, we address the stability question for parameter agnostic policies and show that they are not maximally stable, i.e., they cannot stabilize all underloaded parallel server systems.

Because parameter agnostic policies are particularly attractive when system parameters are unknown, our instability results suggest that even though they are practically desirable, parameter agnostic policies might bring about unsatisfactory results. Insights generated from our analysis carry on to bigger parallel server systems and helps understand parameter agnostic policies in broader systems. In Chapter 4, we extend some of the instability results to less strictly defined policies.

In Chapter 3, we focus on the design of parallel server systems. Given system parameters, we show that a sparse compatibility graph can be constructed that achieves good performance under certain conditions. This problem has previously been addressed under a restrictive assumption: the system has flow conservation. That is, one unit of processing capacity is required for one unit of job. We relax this assumption and allow that arbitrary units of processing capacities may be required to process one unit of a job. This relaxation is of practical interest, as flow conservation may not hold in reality. For instance, in logistical systems, due to physical constraints, various units may be delivered to different destinations from the same origin. Similarly, in service systems, different classes of customers may be served by different servers with different service rates. In production systems, jobs may be competing for machines that can serve their purposes more efficiently. Therefore, addressing the question of what type of compatibility graphs should be built for systems with general processing rates has substantial practical implications. We take the first step in understanding the design principles in such systems. Additionally, in Chapter 4, we present connections between the design problem and the transportation problems, which provides further noteworthy insights into process flexibility design problem.

CHAPTER 2

STABILITY AND INSTABILITY OF PARAMETER AGNOSTIC POLICIES IN PARALLEL SERVER SYSTEMS

2.1 Introduction

In many practical systems that can be modeled as stochastic processing networks, like data centers, hospitals, call centers or manufacturing systems, obtaining accurate information about system parameters such as job arrival and service rates is a non-trivial task. This information is typically deemed necessary because certain control decisions, such as the allocation of resources to jobs (also known as the scheduling policy), must be made. Unfortunately, obtaining precise system parameter information is not always feasible, and the estimates can quickly become outdated if the parameters change.

For instance, in data centers where human involvement is minimal, system parameter estimates can be obtained by investing time and resources into the learning process. Nevertheless, any alterations in the parameters could render those estimates obsolete, necessitating constant monitoring and adjustment.

Similarly, service systems such as call centers also face challenges in acquiring accurate system parameter information. Conducting detailed time studies may be necessary to estimate the service rates of agents, but even then, the estimates might lose relevance if new agents are hired or other factors change.

These examples highlight the need for simple scheduling policies that do not rely on explicit knowledge of the system parameters. We refer to such policies as *parameter agnostic policies*. Some well known scheduling policies fall into this class of policies: longest queue first (LQF) only compares the queue sizes without considering specific parameters. Similarly, static priority policy uses only queue size information. Unlike estimating system parameters, obtaining queue size information is generally easier, faster and more accurate. If a parameter

agnostic policy is adopted, even when system parameters are unknown, imprecise or non-stationary over time, the system would continue to rely on only the queue length information.

In this chapter, we consider parameter agnostic policies in the context of parallel server systems. Parallel server systems consist of servers working in parallel to serve different classes of jobs that are generated externally. The applications described above- data centers and call centers, can be modeled as parallel server systems. We attempt to answer the first order question about parameter agnostic policies in parallel server systems: is the policy maximally stable? In other words, can a parameter agnostic policy stabilize all parallel server systems that can be stabilized?

Stability property is usually a prerequisite for analyzing the performance of a scheduling policy. There is no universal result that establishes or disproves stability for a given scheduling policy, and the characteristics of the underlying system need to be exploited to answer this question. For instance, Gamarnik and Katz (2009) show that characterizing stable queueing networks is an algorithmically undecidable problem for the case of non-preemptive static buffer priority scheduling policies.

Even though parameter agnostic policies are desirable for practical purposes, we show that they can be unstable for even low system loads. We focus on the X-model, a special parallel server system that consists of two servers working in parallel, and two queues with external arrivals. Both servers can serve both queues, but only one at a time. An illustration of this model is given in Figure 2.1, detailed description of the model is given in Section 3.2.

Previously, Baharian and Tezcan (2011) had shown that when the underlying graph of a parallel server system is cycle-free, the LQF policy is maximally stable. Therefore, the X-model is a natural model to consider for the stability question, because it is the smallest parallel server system that contains a cycle. The instability results for the X-model naturally generalize to bigger parallel server systems that contain cycles. This system is also quite intuitive, because a parameter agnostic policy is essentially a function on the two dimensional

graph of queue lengths.

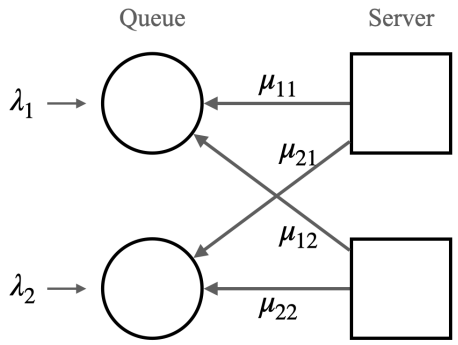


Figure 2.1: Graph of the X-model

A parameter agnostic policy can be an arbitrary function on the two dimensional graph of queue lengths for the X-model. However, in practical systems, there is a preference for more structured policies that prioritize longer queues. This preference stems from the desire to simplify the decision-making process and reduce the length of queues. Accordingly, we focus on parameter agnostic switching curve policies, where each server works according to their own function on the graph of queue lengths, called the switching curve. A formal definition of parameter agnostic switching curve policies for the X-model is given below.

Definition 1. *Let* $\mathbf{Q}(t) = (Q_1(t), Q_2(t)) \in \mathbb{Z}_+^2$ *denote the queue sizes at time* $t \geq 0$*. For* $j \in \{1, 2\}$*, let* $\varphi_j : \mathbb{Z}_+ \to \mathbb{R}_+$ *be non-decreasing functions. A policy is called a switching curve policy if the following conditions hold.*

- *If* $\mathbf{Q}(t)$ *is such that* $\varphi_j(Q_1(t)) > Q_2(t)$*, server* j *serves queue 1.*
- *If* $\varphi_j(Q_1(t)) < Q_2(t)$*, server* j *serves queue 2.*
- *If* $\varphi_j(Q_1(t)) = Q_2(t)$*, the tie is broken arbitrarily, server* j *can serve either queue.*

As a shorthand, we will refer to parameter agnostic switching curve policies as switching curve policies. If the switching curve policy in consideration is not parameter agnostic, we will make the distinction clear.

Informally, there are four different feasible service configurations at any time in the X-model: server j serving queue i where $i, j \in \{1, 2\}$. A switching curve policy determines which service configuration will be used depending on the system state and the functions φ_j, $j \in \{1, 2\}$.

Even though we show our results for non-decreasing φ_j, many of them generalize to less structured parameter agnostic policies. For instance, the monotonicity assumption can be relaxed for the majority of our results. For some of our results, the switching curve policy can be relaxed to any parameter agnostic policy that uses the same service configurations as the discussed switching curve policy. We defer the discussion of such generalizations to Chapter 4, where we consider broader classes of parameter agnostic policies.

2.1.1 Discussion of MaxWeight and Other Related Works

Stability of queueing systems has been studied extensively over several decades. Here, we discuss the most relevant works. Perhaps the most relevant result of all, the well known MaxWeight policy is proven to be maximally stable for parallel server systems (Tassiulas and Ephremides 1990, Stolyar et al. 2004, Dai and Lin 2005). However, MaxWeight assigns a queue to server j according to

$$\arg\max_{i \in \mathcal{I}} \mu_{ij} Q_i(t),$$

where $\mathcal{I}$ is the set of queues, $Q_i(t)$ is the length of queue i at time $t \geq 0$, and μ_{ij} is the service rate of server j to queue i. Since MaxWeight relies on service rate information, it cannot be considered parameter agnostic. Consequently, it becomes susceptible to potential issues that may arise from inaccuracies in the service rate information.

Stability of MaxWeight for parallel server systems is remarkably relevant to the work we

present in this chapter. Though not parameter agnostic, MaxWeight consists of switching curves. In the X-model, MaxWeight assigns a queue to server 1 according to

$$\arg\max\{\mu_{11}Q_1(t), \mu_{21}Q_2(t)\}.$$

Therefore, the switching curve of server 1 under MaxWeight is

$$\varphi_1^{MW}(q_1) = \frac{\mu_{11}}{\mu_{21}}q_1.$$

Similarly, the switching curve of server 2 under MaxWeight is

$$\varphi_2^{MW}(q_1) = \frac{\mu_{12}}{\mu_{22}}q_1.$$

This also implies that MaxWeight uses only three of the four feasible service configurations. Both servers serve queue 1 if $\mathbf{Q}(t)$ is below both of the switching curves. Both servers serve queue 2 if $\mathbf{Q}(t)$ is above both of the switching curves. When $\mathbf{Q}(t)$ is between the switching curves, one of the other two service configurations are used. For example, if

$$\frac{\mu_{11}}{\mu_{21}} > \frac{\mu_{12}}{\mu_{22}},$$

then $\varphi_1^{MW} > \varphi_2^{MW}$. In that case, when $\varphi_1^{MW}(Q_1(t)) > Q_2(t) > \varphi_2^{MW}(Q_1(t))$, server 1 serves queue 1 and server 2 serves queue 2.

Even though MaxWeight is maximally stable for parallel server systems, it is shown to result in instability for switched networks with subcritical system loads. Andrews and Zhang (2003) show that the underlying fluid model of a variant of MaxWeight is not maximally stable for a network of input-queued switches. Bramson et al. (2021) disprove maximal stability of MaxWeight in multihop single class switched networks.

In practice, the system designer could use MaxWeight if she wants to ensure maximal

stability. However, knowledge of service rates would be required in such a case, which is not always easy to obtain. For instance, in data centers, the conditions of processing resources (and accordingly the service rates) can vary over time (Kandula et al. 2009). Then the system designer can either consider adopting parameter agnostic policies, or she can dedicate resources to learn the system parameters.

There are several papers in the literature that consider parameter agnostic policies in parallel server systems. Dimakis and Walrand (2006) study the longest queue first (LQF) policy, and establish sufficient conditions for maximal stability of LQF. For the X-model, Baharian and Tezcan (2011) prove necessary and sufficient conditions for the stability of LQF, and they show that LQF can result in instability for subcritical system loads. Pedarsani and Walrand (2016) study LQF and a related scheduling policy and show maximal stability for a multiclass queueing network with two groups of two queues. Our definition of switching curve policies cover LQF, since LQF essentially asserts in the X-model that both servers have their switching curves $\varphi_1(q_1) = \varphi_2(q_1) = q_1$. When we present instability results for switching curve policies, they would naturally hold for LQF.

Tezcan (2013) shows that static priority policy is not maximally stable in the N-model, which is another simple two-by-two parallel server system. While static priority policy is not a switching curve policy, it has synchronized servers in the context of X-model; that is, both servers serve the same queue at the same time. Generalization of one of our instability results cover synchronized server policies. In other words, our result can be extended to show that static priority policy is not maximally stable in the X-model, which has more overlapping server capabilities than the N-model.

Pedarsani et al. (2017) focus on designing parameter agnostic scheduling policies. They illustrate with the X-model that their proposed policies are not maximally stable. This is in line with our results, we show that parameter agnostic policies are in general not maximally stable.

A more recent branch of the operations research literature focuses on learning the system parameters or policies under unknown system parameters. Krishnasamy et al. (2021), Stahlbuhk et al. (2021) and Choudhury et al. (2021) study the learning problem in a multi armed bandit (MAB) framework. These papers propose algorithms that balance stabilizing the system (exploitation) and learning the service rates (exploration) while minimizing the so-called queue length regret. They all assume that the service rates are stationary over time.

Zhong et al. (2022) consider the learning problem in a multi-class queueing system with abandonments. Although time-varying arrivals are permitted in their model, the customer abandonment assumptions provide natural stabilizing properties. This allows them to alleviate concerns about system stability, as opposed to previous papers whose algorithms need to stabilize the system.

Hsu et al. (2018) study a matching problem using MAB framework, which can be interpreted as the allocation of resources to jobs. They assume arrival rates, service rates and payoff from job completions are all unknown, and show bounds on the payoff gap between their proposed algorithm and the payoff of an oracle who knew all the unknowns.

Adler et al. (2022) consider an $M/M/k/k$ system, also known as Erlang-B, where the service rates are unknown, and unobserved by the dispatcher. They propose a dispatching algorithm that maximizes the long run average reward upon job completion using a maximum likelihood estimation approach.

All the works related to learning accentuate the importance of understanding parameter agnostic policies. Learning is a tedious task, and not lenient in the face of non-stationary system parameters; whereas parameter agnostic policies are robust to parameter changes by design. In this chapter, we attempt to understand parameter agnostic policies in their most fundamental property of stability.

2.1.2 Definitions and Notation

We consider the X-model shown in Figure 2.1. The system consists of two queues with infinite capacity, and two servers that can serve both queues. The external arrival rate to queue i is $\lambda_i \geq 0$ for $i \in \{1, 2\}$. Server j serves queue i with rate $\mu_{ij} > 0$, where $i, j \in \{1, 2\}$. We denote $\boldsymbol{\lambda} = (\lambda_1, \lambda_2)$ and $\boldsymbol{\mu} = (\mu_{11}, \mu_{12}, \mu_{21}, \mu_{22})$.

It is important to note that even though the system designer does not know the system parameters, we assume that she knows that both servers are capable of serving both queues, i.e., $\mu_{ij} > 0$.

We assume that the inter-arrival times to queues and the service times are independent and identically distributed and they follow exponential distribution. Then the system evolution can be represented by a continuous time Markov Chain $(\mathbf{Q}(t))_{t\geq 0}$. The state of the Markov Chain at time $t \geq 0$ is a two dimensional vector $\mathbf{Q}(t) = (Q_1(t), Q_2(t)) \in \mathbb{Z}_+^2$, where $Q_i(t)$ is the size of queue i at time t.

A scheduling policy dictates which queue a server will serve when it becomes idle. We focus on switching curve policies (see Definition 1), where the scheduling policy relies solely on a function of system state $\mathbf{Q}(t)$ and not on the system parameters $\boldsymbol{\lambda}, \boldsymbol{\mu}$. Such a policy is said to stabilize the system if the Markov Chain $(\mathbf{Q}(t))_{t\geq 0}$ is positive recurrent under the policy. The system is unstable if it is not stable.

The system load ρ is a solution to the static planning problem (SPP):

$$\begin{aligned} \text{minimize} \quad & \rho \\ \text{subject to} \quad & \mu_{11}x_{11} + \mu_{12}x_{12} = \lambda_1 \\ & \mu_{21}x_{21} + \mu_{22}x_{22} = \lambda_2 \\ & x_{11} + x_{21} \leq \rho \\ & x_{12} + x_{22} \leq \rho \\ & x_{ij}, \rho \geq 0 \quad \forall i, j \in \{1, 2\}. \end{aligned}$$

Intuitively, x_{ij} represents the allocation of server j's time to queue i. The system is subcritically loaded or underloaded when $\rho < 1$. Then for given $\boldsymbol{\mu}$, the set of all $\boldsymbol{\lambda}$ where a solution to SPP with $\rho < 1$ exists can be written as the *capacity region*.

Definition 2. *For given $\boldsymbol{\mu}$, capacity region for the X-model is defined as*

$$\begin{aligned} \Lambda(\boldsymbol{\mu}) = \big\{ \boldsymbol{\lambda} \in \mathbb{R}_+^2 : \exists p, q \in (0,1) \text{ such that} \\ \lambda_1 < p\mu_{11} + q\mu_{12}, \\ \lambda_2 < (1-p)\mu_{21} + (1-q)\mu_{22} \big\}. \end{aligned}$$

One of the perks of the X-model is that capacity region can easily be plotted on $\mathbb{R}_+^2$. For instance, if $\mu_{11} > \mu_{12}$ and $\mu_{22} > \mu_{21}$, the capacity region would be as in Figure 2.2. For given $\boldsymbol{\mu}$, $\boldsymbol{\lambda}$ being in the capacity region is a well known necessary condition for stability, but it is not always a sufficient condition. We will show that for certain switching curve policies, there exist $\boldsymbol{\lambda}, \boldsymbol{\mu}$ with $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$ such that the system is unstable. In other words, we show that switching curve policies are not maximally stable.

We tackle the problem of stability for switching curve policies case by case. First, in

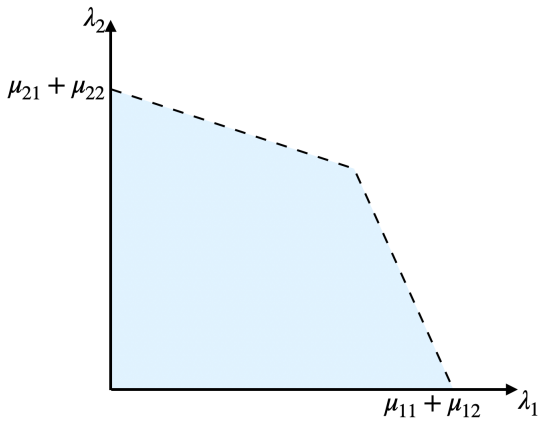


Figure 2.2: Capacity Region of the X-model where $\mu_{11} > \mu_{12}$ and $\mu_{22} > \mu_{21}$

section 2.2, we consider the most general setting, where the switching curves intersect infinitely often as the queue sizes grow. We show the existence of parameters in the capacity region under which the system is unstable. Then in Section 2.3, we impose more structure on the switching curves and assume that the curves do not intersect after a finite q_1^*. For this case, we show an instability and a stability result. In section 2.4, we further specialize the switching curves and assume that the curves are identical for both servers. We show a sufficient condition for instability, and discuss how the capacity region is restricted under this condition. In section 2.5, we discuss some implications of our results and provide our concluding remarks.

2.2 Intersecting Switching Curves

Let $\varphi_1(q_1)$ and $\varphi_2(q_1)$ be the switching curves of server 1 and 2, respectively. In this section, we assume that there does not exist $q_1^* \in \mathbb{Z}_+$ such that $\varphi_i(q_1) > \varphi_j(q_1)$ for all $q_1 > q_1^*$, where $i, j \in \{1, 2\}$ with $i \neq j$. That is, the switching curves $\varphi_1(q_1)$ and $\varphi_2(q_1)$ intersect infinitely often as q_1 grows.

Here, we introduce our most general result where the switching curves are intersecting infinitely often. This is the most complicated result of the paper, and uses a coupling

argument to show instability in such switching curve policies. The methodology that is introduced in this section partially recovers the instability results we obtain in sections 2.3 and 2.4. We will show an application of this methodology to non-intersecting switching curves in section 2.3, and highlight the differences in the sufficient conditions obtained in both cases.

When the intersecting switching curves intersect infinitely often, we show that for some $\boldsymbol{\mu}$, there exists $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$ such that the system is unstable. The course of the argument is as follows. We first show a sufficient condition for a single dimensional continuous time Markov Chain to be either null recurrent or transient (i.e. not positive recurrent). We then show that the evolution of $(\mathbf{Q}(t))_{t\geq 0}$ couples with the single dimensional continuous time Markov Chain, and the state of the single dimensional Markov Chain is a lower bound on $Q_1(t) + Q_2(t)$ for all $t \geq 0$.

In section 2.2.1, we show the lack of stationary distribution for a single dimensional Markov Chain. In section 2.2.2, we show that $(\mathbf{Q}(t))_{t\geq 0}$ couples with $(X(t))_{t\geq 0}$, and conclude that it cannot have a stationary distribution.

2.2.1 Lack of Stationary Distribution for a Markov Chain with One Dimensional State Space

Let $(X(t))_{t\geq 0}$ be a continuous time Markov Chain with state space $\mathbb{Z}_+$. Let $(c_k)_{k\in\mathbb{Z}_+}$ be an increasing sequence in the state space $\mathbb{Z}_+$, with $c_0 = 0$. The state space can then be partitioned by the sets

$$I_k = \{s \in \mathbb{Z}_+ : c_{2k-2} \leq s < c_{2k-1}\}$$
$$J_k = \{s \in \mathbb{Z}_+ : c_{2k-1} \leq s < c_{2k}\}$$

for all $k \in \mathbb{Z}_{++}$. For $s \in \mathbb{Z}_+$, let the transition rates of the Markov Chain $(X(t))_{t\geq 0}$ be

$$\begin{aligned} q_{s,s+1} &= \alpha, \\ q_{s,s-1} &= \beta_1 \quad \text{if} \quad s \in I_k, \\ q_{s,s-1} &= \beta_2 \quad \text{if} \quad s \in J_k \end{aligned}$$

for any $k \in \mathbb{Z}_{++}$, where $\alpha, \beta_1, \beta_2 \in \mathbb{R}_+$. Let $\beta_1 < \alpha < \beta_2$.

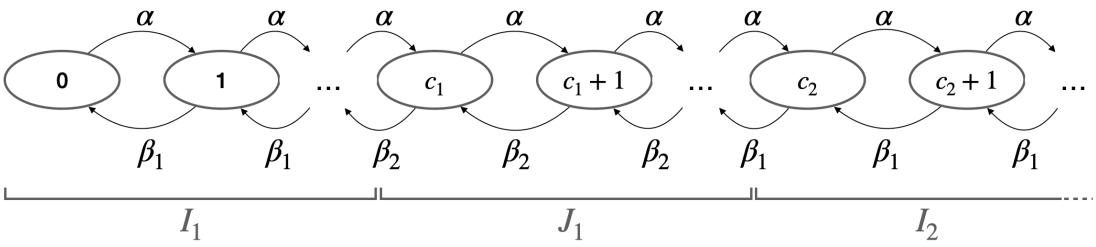


Figure 2.3: Constructed One Dimensional CTMC

Note that $(X(t))_{t\geq 0}$ is a generalized birth and death process (representation given in Figure 2.3). If a stationary distribution exists, the process is time reversible, then the detailed balance equations,

$$\pi_s q_{s,r} = \pi_r q_{r,s},$$

must be satisfied for all states $s, r \in \mathbb{Z}_+$, where π_s is the steady state probability of being in state s. We will use this property to find a sufficient condition for $(X(t))_{t\geq 0}$ to not be positive recurrent in the following proposition.

Proposition 1. *$(X(t))_{t\geq 0}$ is not positive recurrent if*

$$\lim_{n\to\infty}\left(\frac{\alpha}{\beta_1}\right)^{\sum_{k=1}^{n}|I_k|/n}\left(\frac{\alpha}{\beta_2}\right)^{\sum_{k=1}^{n}|J_k|/n}>1.$$

Proof. Assume to the contrary that $(X_t)_{t\geq 0}$ is positive recurrent. Then a stationary distribution $\boldsymbol{\pi}$ exists and the detailed balance equations are satisfied.

$$\begin{aligned}
\pi_0\alpha &= \pi_1\beta_1\\
\pi_1\alpha &= \pi_2\beta_1\\
&\vdots\\
\pi_{c_1-1}\alpha &= \pi_{c_1}\beta_2\\
\pi_{c_1}\alpha &= \pi_{c_1+1}\beta_2\\
&\vdots\\
\pi_{c_2-1}\alpha &= \pi_{c_2}\beta_1\\
\pi_{c_2}\alpha &= \pi_{c_2+1}\beta_1\\
&\vdots
\end{aligned}$$

Because $\boldsymbol{\pi}$ is a probability distribution, $\sum_{s=0}^{\infty}\pi_s=1$. For ease of notation, let $a:=\frac{\alpha}{\beta_1}$ and $b:=\frac{\alpha}{\beta_2}$. Using $c_{2k-1}-c_{2k-2}=|I_k|$ and $c_{2k}-c_{2k-1}=|J_k|$, where $|\cdot|$ denotes the cardinality of the set. Expanding the summation,

$$\pi_0\Bigg[-\frac{1}{a-1}+\Big(\frac{1}{a-1}+\frac{1}{1-b}\Big)\Big(a^{|I_1|}+a^{|I_1|}b^{|J_1|}(a^{|I_2|}-1)\\ +a^{|I_1|+|I_2|}b^{|J_1|+|J_2|}(a^{|I_3|}-1)+\dots\Big)\Bigg]=1.$$

Note that $a > 1$ and $b < 1$. We had assumed

$$\lim_{n\to\infty} a^{\sum_{k=1}^{n} |I_k|/n} b^{\sum_{k=1}^{n} |J_k|/n} > 1,$$

which is a sufficient condition for the infinite sum above to diverge. Then stationary distribution $\boldsymbol{\pi}$ cannot exist, contradicting positive recurrence of $(X(t))_{t\geq 0}$. □

2.2.2 Coupling of the X-Model with the One Dimensional Markov Chain

Let the intersection points of φ_1 and φ_2 have coordinates $(x_k, y_k)_{k\in\mathbb{Z}_+}$. Then state space of $(\mathbf{Q}(t))_{t\geq 0}$ can be partitioned into subsets

$$U_k = \{(q_1, q_2) \in \mathbb{Z}_+^2 : x_{2k-2} + y_{2k-2} \leq q_1 + q_2 < x_{2k-1} + y_{2k-1},\ \varphi_1(q_1) \leq \varphi_2(q_1)\},$$
$$V_k = \{(q_1, q_2) \in \mathbb{Z}_+^2 : x_{2k-1} + y_{2k-1} \leq q_1 + q_2 < x_{2k} + y_{2k},\ \varphi_2(q_1) \leq \varphi_1(q_1)\}$$

where $(x_0, y_0) = (0, 0)$. This partitioning of the state space distinguishes where φ_1 lies above φ_2 and visa versa. An illustration of this partitioning is given in Figure 2.4. When $\boldsymbol{\lambda}, \boldsymbol{\mu}$ and the sequence $(c_k)_{k\in\mathbb{Z}_+}$ are appropriately chosen, the process $Q_1(t) + Q_2(t)$ couples with $X(t)$, implying the instability of $(\mathbf{Q}(t))_{t\geq 0}$.

More specifically, let $c_k = \lceil x_k + y_k \rceil$ for all $k \in \mathbb{Z}_+$. Then for any k, $(q_1, q_2) \in U_k$ if and only if $q_1 + q_2 \in I_k$. Similarly, $(q_1, q_2) \in V_k$ if and only if $q_1 + q_2 \in J_k$.

Theorem 1. *If $\varphi_1(q_1)$ and $\varphi_2(q_1)$ are non-decreasing switching curves with intersection points $(x_k, y_k)_{k\in\mathbb{Z}_+}$, and*

$$\lim_{n\to\infty} \frac{\sum_{k=1}^{n} |I_k|}{\sum_{k=1}^{n} |J_k|} \in [0, \infty],$$

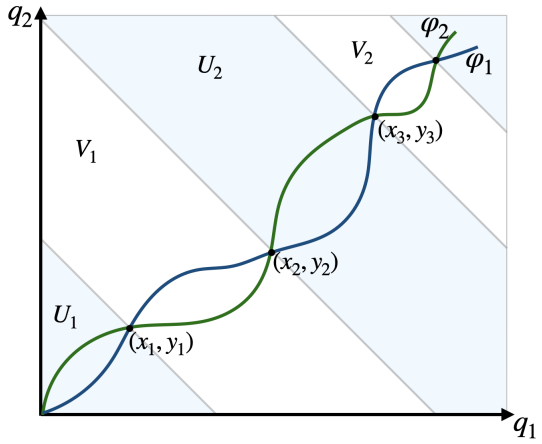


Figure 2.4: Partitioning of the State Space of $(\mathbf{Q}(t))_{t\geq 0}$

where $c_k = \lceil x_k + y_k \rceil$ *for all* $k \in \mathbb{Z}_+$*, then there exists* $\boldsymbol{\mu}$ *and* $\boldsymbol{\lambda}$ *with* $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$ *such that the system is unstable.*

Proof. We will prove the theorem for $\lim_{n\to\infty} \frac{\sum_{k=1}^{n} |I_k|}{\sum_{k=1}^{n} |J_k|} \geq 1$, the other case can be proven in a similar way with symmetrical modifications.

Let

$$\lim_{n\to\infty} \frac{\sum_{k=1}^{n} |I_k|}{\sum_{k=1}^{n} |J_k|} \geq 1,$$

and $\lambda_1 = \lambda_2 := \lambda$, $\mu_{11} = \mu_{22} := \mu$, $\mu_{12} = \mu_{21} := \mu'$, where $\mu > \lambda > \mu'$. It can be seen that $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$, and the system is underloaded.

We consider the uniformized discrete time Markov Chain underlying the continuous time Markov Chain $(\mathbf{Q}(t))_{t\geq 0}$ with the uniformization constant $P := 2\lambda + 2\mu + 2\mu'$. With a slight abuse of notation, we denote the uniformized Markov Chain again with $\mathbf{Q}(t)$ for $t \in \{0, 1, \ldots\}$.

For each $t \geq 0$, let $Z(t)$ be independent and identically distributed uniform random variables between 0 and 1. Then for $Q_1(t) > 0$ and $Q_2(t) > 0$, one step transition of the

system state $(Q_1(t), Q_2(t))$ would result in

$$\mathbf{Q}(t+1) - \mathbf{Q}(t) = \begin{cases} (1,0) & \text{if } Z(t) \in \left[0, \frac{\lambda}{P}\right) \\ (0,1) & \text{if } Z(t) \in \left[\frac{\lambda}{P}, \frac{2\lambda}{P}\right) \\ (-1,0) & \text{if } Z(t) \in \left[\frac{2\lambda}{P}, \frac{2\lambda+s_1}{P}\right) \\ (0,-1) & \text{if } Z(t) \in \left[\frac{2\lambda+s_1}{P}, \frac{2\lambda+s_1+s_2}{P}\right) \\ (0,0) & \text{if } Z(t) \in \left[\frac{2\lambda+s_1+s_2}{P}, 1\right) \end{cases}$$

where s_1 and s_2 are the service rates that queue 1 and queue 2 receive at time t, respectively. According to the partitioning of the state space to subsets U_k and V_k, the scheduling policy dictates that in sets U_k, either $s_1 + s_2 = \mu + \mu'$ or $s_1 + s_2 = 2\mu'$. Similarly, in sets V_k, either $s_1 + s_2 = 2\mu$ or $s_1 + s_2 = \mu + \mu'$. Note that $2\mu > \mu + \mu' > 2\mu'$.

Now consider the continuous time Markov Chain $(X(t))_{t\geq 0}$ described in Section 2.2.1, with the newly assigned transition rates

$$\begin{aligned} q_{s,s+1} &= 2\lambda, \\ q_{s,s-1} &= \mu + \mu' \quad \text{if} \quad s \in I_k, \\ q_{s,s-1} &= 2\mu \quad \text{if} \quad s \in J_k. \end{aligned}$$

$(X(t))_{t\geq 0}$ can be similarly uniformized with the uniformization constant $P = 2\lambda + 2\mu + 2\mu'$. Then, if $X(t) \in I_k$ for any k, one step transition of $X(t)$ would result in

$$X(t+1) - X(t) = \begin{cases} 1, & \text{if } Z(t) \in \left[0, \frac{2\lambda}{P}\right) \\ -1, & \text{if } Z(t) \in \left[\frac{2\lambda}{P}, \frac{2\lambda+\mu+\mu'}{P}\right) \\ 0, & \text{if } Z(t) \in \left[\frac{2\lambda+\mu+\mu'}{P}, 1\right) \end{cases} .$$

If $X(t) \in J_k$ for any k, one step transition of $X(t)$ would result in

$$X(t+1) - X(t) = \begin{cases} 1, & \text{if } Z(t) \in \left[0, \frac{2\lambda}{P}\right) \\ -1, & \text{if } Z(t) \in \left[\frac{2\lambda}{P}, \frac{2\lambda+2\mu}{P}\right) \\ 0, & \text{if } Z(t) \in \left[\frac{2\lambda+2\mu}{P}, 1\right) \end{cases} .$$

We will show that if $X(0) = Q_1(0) + Q_2(0)$, then we have $X(t) \leq Q_1(t) + Q_2(t)$ for all $t \geq 0$.

To show this by induction, we assume that $X(t) \leq Q_1(t) + Q_2(t)$ and we will show $X(t+1) \leq Q_1(t+1) + Q_2(t+1)$. We need to consider this case by case according to the service configuration that $(Q_1(t), Q_2(t))$ will trigger depending on the scheduling policy.

- If $\mathbf{Q}(t) \in U_k$ and $s_1 + s_2 = \mu + \mu'$, then $Q_1(t) + Q_2(t)$ will evolve according to $Z(t)$ exactly as $X(t)$ does, ensuring that $X(t+1) \leq Q_1(t+1) + Q_2(t+1)$.

- If $\mathbf{Q}(t) \in U_k$ and $s_1 + s_2 = 2\mu'$,

  - If
  $$Z(t) \in \left[\frac{2\lambda + 2\mu'}{P}, \frac{2\lambda + \mu + \mu'}{P}\right),$$
  then
  $$Q_1(t+1) + Q_2(t+1) - Q_1(t) - Q_2(t) = 0,$$
  $$X(t+1) - X(t) = -1,$$
  ensuring $X(t+1) \leq Q_1(t+1) + Q_2(t+1)$.

  - If
  $$Z(t) \notin \left[\frac{2\lambda + 2\mu'}{P}, \frac{2\lambda + \mu + \mu'}{P}\right),$$

then $Q_1(t)+Q_2(t)$ will evolve according to $Z(t)$ exactly as $X(t)$, so that $X(t+1) \leq Q_1(t+1) + Q_2(t+1)$.

- If $(Q_1(t), Q_2(t)) \in V_k$ and $s_1 + s_2 = 2\mu$, then $Q_1(t) + Q_2(t)$ will evolve according to $Z(t)$ exactly as $X(t)$, so that $X(t+1) \leq Q_1(t+1) + Q_2(t+1)$.

- If $(Q_1(t), Q_2(t)) \in V_k$ and $s_1 + s_2 = \mu + \mu'$,

  - If
$$Z(t) \in \left[\frac{2\lambda + \mu + \mu'}{P}, \frac{2\lambda + 2\mu}{P}\right),$$
then
$$Q_1(t+1) + Q_2(t+1) - Q_1(t) - Q_2(t) = 0,$$
$$X(t+1) - X(t) = -1,$$
ensuring $X(t+1) \leq Q_1(t+1) + Q_2(t+1)$.

  - If
$$Z(t) \notin \left[\frac{2\lambda + \mu + \mu'}{P}, \frac{2\lambda + 2\mu}{P}\right),$$
then $Q_1(t) + Q_2(t)$ will evolve according to $Z(t)$ exactly as $X(t)$ does, ensuring $X(t+1) \leq Q_1(t+1) + Q_2(t+1)$.

So far, we have assumed $Q_1(t) > 0$ and $Q_2(t) > 0$. The boundary cases where either $Q_1(t) = 0$ or $Q_2(t) = 0$ should also be considered. When $Q_1(t) > 0$ and $Q_2(t) = 0$, both servers serve the first queue, coinciding with the cases where $s_1 + s_2 = \mu + \mu'$. Similarly, when $Q_1(t) = 0$ and $Q_2(t) > 0$, both servers serve the second queue, coinciding with the cases where $s_1 + s_2 = \mu + \mu'$. When $Q_1(t) = 0$ and $Q_2(t) = 0$, the analysis remains very similar, except the transition probabilities from $\mathbf{Q}(t)$ to $\mathbf{Q}(t+1)$ are different. In that case,

one step transition would result in

$$\mathbf{Q}(t+1)-\mathbf{Q}(t)=\begin{cases}(1,0) & \text{if } Z(t)\in\left[0,\frac{\lambda}{P}\right),\\ (0,1) & \text{if } Z(t)\in\left[\frac{\lambda}{P},\frac{2\lambda}{P}\right)\\ (0,0) & \text{if } Z(t)\in\left[\frac{2\lambda}{P},1\right)\end{cases}.$$

In this case, $X(t)\leq Q_1(t)+Q_2(t)$ still holds. Therefore, $(\mathbf{Q}(t))_{t\geq 0}$ couples with the one dimensional Markov Chain $(X(t))_{t\geq 0}$.

The assumption

$$\lim_{n\to\infty}\frac{\sum_{k=1}^{n}|I_k|}{\sum_{k=1}^{n}|J_k|}\geq 1$$

implies that

$$\lim_{n\to\infty}\left(\frac{2\lambda}{\mu+\mu'}\right)^{\sum_{k=1}^{n}|I_k|/n}\left(\frac{2\lambda}{2\mu}\right)^{\sum_{k=1}^{n}|J_k|/n}>1.$$

In Proposition 1, this coincides with $\alpha=2\lambda$, $\beta_1=\mu+\mu'$, $\beta_2=2\mu$. Therefore, by Proposition 1, $X(t)$ is not positive reccurrent. Since for all t, $X(t)\leq Q_1(t)+Q_2(t)$, then $Q_1(t)+Q_2(t)$ cannot have a stationary distribution, implying instability of the system. □

This is our only result where the monotonicity assumption on the switching curves φ_j cannot be naturally relaxed. This is because the coupling result relies on the fact that only three of the four feasible service configurations can be used in sets U_k and V_k. We can relax the monotonicity assumption if it is still true that only three of the four feasible service configurations can be used in sets U_k and V_k. Then the instability result would still be valid.

Note that MaxWeight also employs three of the four feasible service configurations. Both servers serving queue 1 and both servers serving queue 2 are the configurations that are always included in MaxWeight, similar to the switching curves. In MaxWeight, depending on the ratios of service rates μ_{11}/μ_{21} and μ_{12}/μ_{22}, either server j serving queue $i=j$ or

server j serving queue $i \neq j$ is employed. Let us informally call this "the service configuration that result in stability". Then intuitively, the limit condition in Theorem 1 suggests that on average, the service configuration that results in stability according to MaxWeight is used in a smaller subset of the state space compared to the service configuration that results in instability.

2.3 Non-Intersecting Switching Curves

Imposing more structure on the policy, we now assume that there are two switching curves φ_1 and φ_2 such that there exists $q_1^* \in \mathbb{Z}_+$ such that for all $q_1 > q_1^*$, $\varphi_2(q_1) > \varphi_1(q_1)$. That is, after a finite point in the state space $\mathbb{Z}_+^2$, the switching curves φ_1 and φ_2 do not intersect. The assumption $\varphi_2(q_1) > \varphi_1(q_1)$ is without loss of generality, the results would hold with minor and symmetrical modifications if the direction of the inequality is reversed.

In this section, we first present an instability result, and then a stability result. Even though the stability result is proven for switching lines instead of switching curves, our intuition tells us the result would hold under mild conditions, such as the existence of a fluid model limit, with switching curves.

2.3.1 Instability Result

Proposition 2. *If there exists $q_1^* \in \mathbb{Z}_+$ such that for all $q_1 > q_1^*$, $\varphi_2(q_1) > \varphi_1(q_1)$, then there exist $\boldsymbol{\lambda}, \boldsymbol{\mu}$ with $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$ such that the system is unstable.*

Proof of Proposition 2. By Theorem 4.1 in Dai (1995), a fluid limit for the parallel server system exists and satisfies the fluid model equations. Let $\bar{Q}_i(t) \geq 0$ be the amount of fluid in queue i at time t, $\bar{T}_{ij}(t) \in [0, t]$ be the total amount of time up to t that server j serves queue i. Let $\bar{I}_j(t) \in [0, t]$ be the total idle time of server j up to time t. The fluid model

equations are

$$\bar{Q}_i(t) = \bar{Q}_i(0) + \lambda_i t - \mu_{i1}\bar{T}_{i1}(t) - \mu_{i2}\bar{T}_{i2}(t), \text{ for } i \in \{1,2\}, \tag{2.1}$$

$$\bar{T}_{ij}(t) \text{ starts from zero and is non-decreasing in } t, \text{ for } i,j \in \{1,2\}, \tag{2.2}$$

$$\bar{I}_j(t) = t - \bar{T}_{1j}(t) - \bar{T}_{2j} \text{ is non-decreasing, for } j \in \{1,2\}, \tag{2.3}$$

$$\int_0^\infty (\bar{Q}_1(t) + \bar{Q}_2(t)) d\bar{I}_j(t) = 0, \text{ for } j \in \{1,2\}. \tag{2.4}$$

In addition, the following policy specific fluid model equations hold:

$$\dot{\bar{T}}_{11}(t) = 1 \text{ and } \dot{\bar{T}}_{21}(t) = 0 \text{ if } \varphi_1(\bar{Q}_1(t)) > \bar{Q}_2(t)), \tag{2.5}$$

$$\dot{\bar{T}}_{21}(t) = 1 \text{ and } \dot{\bar{T}}_{11}(t) = 0 \text{ if } \varphi_1(\bar{Q}_1(t)) < \bar{Q}_2(t)),$$

$$\dot{\bar{T}}_{12}(t) = 1 \text{ and } \dot{\bar{T}}_{22}(t) = 0 \text{ if } \varphi_2(\bar{Q}_1(t)) > \bar{Q}_2(t)),$$

$$\dot{\bar{T}}_{22}(t) = 1 \text{ and } \dot{\bar{T}}_{12}(t) = 0 \text{ if } \varphi_2(\bar{Q}_1(t)) < \bar{Q}_2(t)), \tag{2.6}$$

where $\dot{\bar{T}}_{ij}(t) = \frac{d}{dt}\bar{T}_{ij}(t)$. The ties are broken arbitrarily, meaning that if $\varphi_1(\bar{Q}_1(t)) = \bar{Q}_2(t))$, server 1 can serve either queue and if $\varphi_2(\bar{Q}_1(t)) = \bar{Q}_2(t))$, server 2 can serve either queue. We note that equations 2.5 and 2.6 cannot hold at the same time, because we assume $\varphi_2(q_1) > \varphi_1(q_1)$.

Let $||\bar{\mathbf{Q}}(t)|| = w_1\bar{Q}_1(t) + w_2\bar{Q}_2(t)$ where $w_1, w_2 \in \mathbb{R}_{++}$. Then we can decompose the defined norm using the fluid model equations,

$$\begin{aligned} ||\bar{\mathbf{Q}}(t)|| =& w_1\bar{Q}_1(0) + w_2\bar{Q}_2(0) + (w_1\lambda_1 + w_2\lambda_2)t \\ &- [w_1(\mu_{11} + \mu_{12})t_A(t) + (w_1\mu_{12} + w_2\mu_{21})t_B(t) + w_2(\mu_{21} + \mu_{22})t_C(t)], \end{aligned} \tag{2.7}$$

where $t_A(t), t_B(t)$ and $t_C(t)$ are the total amount of time until t spent with service configurations where both servers serve queue 1; server 1 serves queue 2, server 2 serves queue 1;

and both servers serve queue 2, respectively. That is,

$$t_A(t) := \int_0^t \dot{\bar{T}}_{11}(t)\dot{\bar{T}}_{12}(t)dt,$$
$$t_B(t) := \int_0^t \dot{\bar{T}}_{12}(t)\dot{\bar{T}}_{21}(t)dt,$$
$$t_C(t) := \int_0^t \dot{\bar{T}}_{21}(t)\dot{\bar{T}}_{22}(t)dt.$$

In addition, we assume that the initial condition $\bar{\mathbf{Q}}(0)$ is large enough such that $\bar{I}_j(t) = 0$ for $j \in \{1, 2\}$. Then we have $t = t_A(t) + t_B(t) + t_C(t)$. According to the service regimes that contribute to $t_A(t)$, $t_B(t)$ and $t_C(t)$, expression 2.7 can be decomposed as

$$D_A(t_A(t)) := (w_1\lambda_1 + w_2\lambda_2)t_A(t) - w_1(\mu_{11} + \mu_{12})t_A(t),$$
$$D_B(t_B(t)) := (w_1\lambda_1 + w_2\lambda_2)t_B(t) - (w_1\mu_{12} + w_2\mu_{21})t_B(t),$$
$$D_C(t_C(t)) := (w_1\lambda_1 + w_2\lambda_2)t_C(t) - w_2(\mu_{21} + \mu_{22})t_C(t),$$

and we can write

$$||\bar{\mathbf{Q}}(t)|| = w_1\bar{Q}_1(0) + w_2\bar{Q}_2(0) + D_A(t_A(t)) + D_B(t_B(t)) + D_C(t_C(t)).$$

We will now establish that there exist $w_1, w_2 > 0$ such that

$$D_A(t_A(t)) > \delta_A t_A(t),$$
$$D_B(t_B(t)) > \delta_B t_B(t),$$
$$D_C(t_C(t)) > \delta_C t_C(t)$$

for some $\boldsymbol{\lambda}, \boldsymbol{\mu}$ with $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$, where $\delta_A, \delta_B, \delta_C > 0$. That would imply by Theorem 3.2 in Meyn et al. (1995) that the Markov Chain $(\mathbf{Q}(t))_{t\geq 0}$ is transient.

Such w_1 and w_2 would lie in the intersection of the conditions derived separately for regions A, B and C. We suppress the dependence on t and denote $t_A(t) = t_A$, $t_B(t) = t_B$, $t_C(t) = t_C$ for convenience.

- Region A: both servers serve queue 1. We have $D_A(t_A) > \delta_A t_A$ if

$$\frac{w_1}{w_2} < \frac{\lambda_2}{\mu_{11} + \mu_{12} - \lambda_1}. \tag{2.8}$$

  To see this, we directly use the definition of $D_A(t_A)$.

$$\begin{aligned} D_A(t_A) &= (w_1\lambda_1 + w_2\lambda_2)t_A - w_1(\mu_{11} + \mu_{12})t_A \\ &= [w_1(\lambda_1 - \mu_{11} - \mu_{12}) + w_2\lambda_2]t_A \\ &= \delta_A t_A, \end{aligned}$$

  where $\delta_A > 0$ if $w_2\lambda_2 > w_1(\mu_{11} + \mu_{12} - \lambda_1)$.

- Region B: server 1 serves queue 2 and server 2 serves queue 1. We have $D_B(t_B) > \delta_B t_B$ if

$$w_1(\lambda_1 - \mu_{12}) + w_2(\lambda_2 - \mu_{21}) > 0. \tag{2.9}$$

  To see this, we use the definition of $D_B(t_B)$.

$$\begin{aligned} D_B(t_B) &= (w_1\lambda_1 + w_2\lambda_2)t_B - (w_1\mu_{12} + w_2\mu_{21})t_B \\ &= [w_1(\lambda_1 - \mu_{12}) + w_2(\lambda_2 - \mu_{21})]t_B \\ &= \delta_B t_B, \end{aligned}$$

  where $\delta_B > 0$ if $w_1(\lambda_1 - \mu_{12}) + w_2(\lambda_2 - \mu_{21}) > 0$.

- Region C: both servers serve queue 2. We have $D_C(t_C) > \delta_C t_C$ if

$$\frac{w_1}{w_2} > \frac{\mu_{21} + \mu_{22} - \lambda_2}{\lambda_1}. \tag{2.10}$$

To see this, we use the definition of $D_C(t_C)$.

$$\begin{aligned} D_C(t_C) &= (w_1\lambda_1 + w_2\lambda_2)t_C - w_2(\mu_{21} + \mu_{22})t_C \\ &= [w_1\lambda_1 + w_2(\lambda_2 - \mu_{21} - \mu_{22})]t_C \\ &= \delta_C t_C, \end{aligned}$$

where $\delta_C > 0$ if $w_1\lambda_1 > w_2(\mu_{21} + \mu_{22} - \lambda_2)$.

System parameters in the capacity region that also satisfy conditions 2.8, 2.9 and 2.10 can be found. For example, let $\lambda_1 = 6.65, \lambda_2 = 8, \mu_{11} = 10, \mu_{12} = 1, \mu_{21} = 3, \mu_{22} = 7$. These parameters are in the stability region and satisfy 2.8, 2.9 and 2.10 with $w_1 = 0.1376$, $w_2 = 0.1839$. Therefore, there exist $\boldsymbol{\lambda}, \boldsymbol{\mu}$ with $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$ such that the system is unstable. A parameter agnostic policy with non-intersecting switching curves is not maximally stable.

□

A complete characterization of the conditions on $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$ can also be derived based on inequalities 2.8, 2.9 and 2.10. For example, let $\mu_{11} > \mu_{12}$ and $\mu_{22} > \mu_{21}$. Then $w_1, w_2 > 0$ exist if $\lambda_1 > \mu_{12}$, $\lambda_2 > \mu_{21}$ and $(\mu_{11} + \mu_{12})(\mu_{21} + \mu_{22}) - \lambda_1(\mu_{21} + \mu_{22}) - \lambda_2(\mu_{11} + \mu_{12}) < 0$. Repeating the same exercise for cases where $\lambda_1 > \mu_{12}$, $\lambda_2 < \mu_{21}$ and $\lambda_1 < \mu_{12}$, $\lambda_2 > \mu_{21}$, we can see that the shaded part of the capacity region in Figure 2.5 results in instability.

A sufficient condition for instability can also be derived using the methodology introduced in Section 2.2 for intersecting switching curves. The sufficient condition is different than Proposition 2, because the methodology in Section 2.2 tracks the evolution of $Q_1(t) + Q_2(t)$

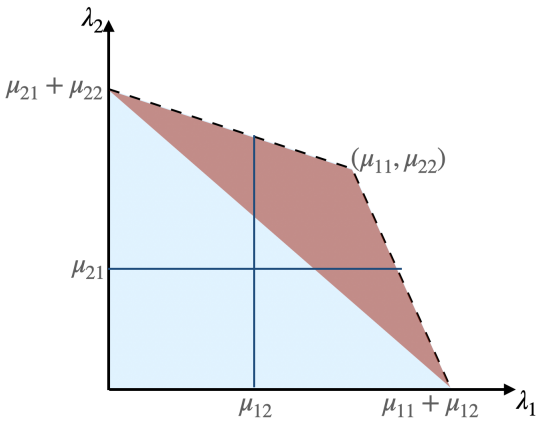


Figure 2.5: The Instability Region for Non-Intersecting Switching Curves According to Proposition 2

whereas the proof of Proposition 2 tracks $w_1\bar{Q}_1(t) + w_2\bar{Q}_2(t)$. In Corollaries 1 and 2, we apply the methodologies in Proposition 1 and Theorem 1 to the case of non intersecting switching curves.

Corollary 1 (of Proposition 1)**.** *Let $(X(t))_{t\geq 0}$ be a special case of the constructed continuous time Markov Chain with $\beta_1 = \beta_2 = \beta$. $(X(t))_{t\geq 0}$ is not positive recurrent if*

$$\frac{\alpha}{\beta} \geq 1.$$

Proof. $(X(t))_{t\geq 0}$ is a birth-death process with forward rates α and backward rates β. Assume to the contrary that $(X(t))_{t\geq 0}$ is positive recurrent. Then the process is time reversible and the detailed balance equations must be satisfied. That is,

$$\pi_0\alpha = \pi_1\beta$$

$$\pi_1\alpha = \pi_2\beta$$

$$\vdots$$

Because $\boldsymbol{\pi}$ is a probability distribution, $\sum_{s=0}^{\infty} \pi_s = 1$. Then we must have

$$\pi_0 \sum_{n=0}^{\infty} \left(\frac{\alpha}{\beta}\right)^n = 1.$$

But we assumed that

$$\frac{\alpha}{\beta} \geq 1.$$

Then the infinite sum diverges, and $(X(t))_{t\geq 0}$ cannot be positive recurrent. □

Then following the steps in the proof of Theorem 1, we can find the instability condition for the non-intersecting switching curves.

Corollary 2 (of Theorem 1)**.** *Let $\varphi_2(q_1) > \varphi_1(q_1)$ for all $q_1 \in \mathbb{Z}_+$. Then the X-model is unstable for $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$ if*

$$\frac{\lambda_1 + \lambda_2}{\max\{\mu_{11} + \mu_{12}, \mu_{12} + \mu_{21}, \mu_{21} + \mu_{22}\}} \geq 1.$$

Proof. We consider the uniformized discrete time Markov Chain underlying the continuous time Markov Chain $(\mathbf{Q}(t))_{t\geq 0}$ with the uniformization constant $P := \lambda_1 + \lambda_2 + \mu_{11} + \mu_{21} + \mu_{12} + \mu_{22}$. With a slight abuse of notation, we denote the uniformized Markov Chain again with $\mathbf{Q}(t)$ for $t \in \{0, 1, \ldots\}$.

For each $t \geq 0$, let $Z(t)$ be i.i.d. uniform random variables between 0 and 1. Then for $Q_1(t) > 0$ and $Q_2(t) > 0$, one step transition of the system state $(Q_1(t), Q_2(t))$ would result

in

$$\mathbf{Q}(t+1)-\mathbf{Q}(t)=\begin{cases}(1,0) & \text{if } Z(t)\in\left[0,\frac{\lambda_1}{P}\right)\\ (0,1) & \text{if } Z(t)\in\left[\frac{\lambda_1}{P},\frac{\lambda_1+\lambda_2}{P}\right)\\ (-1,0) & \text{if } Z(t)\in\left[\frac{\lambda_1+\lambda_2}{P},\frac{\lambda_1+\lambda_2+s_1}{P}\right)\\ (0,-1) & \text{if } Z(t)\in\left[\frac{\lambda_1+\lambda_2+s_1}{P},\frac{\lambda_1+\lambda_2+s_1+s_2}{P}\right)\\ (0,0) & \text{if } Z(t)\in\left[\frac{\lambda_1+\lambda_2+s_1+s_2}{P},1\right)\end{cases}$$

where s_1 and s_2 are the service rates that queue 1 and queue 2 receive at time t, respectively. When $\varphi_2(q_1)>\varphi_1(q_1)$, s_1+s_2 can take three values. If both servers serve queue 1, $s_1+s_2=\mu_{11}+\mu_{12}$; if both servers serve queue 2, $s_1+s_2=\mu_{21}+\mu_{22}$; if server 1 serves queue 2 and server 2 serves queue 1, $s_1+s_2=\mu_{12}+\mu_{21}$.

Now consider the birth-death process $(X(t))_{t\geq 0}$ described in Corollary 1, with the transition rates

$$\begin{aligned}q_{s,s+1}&=\lambda_1+\lambda_2,\\ q_{s,s-1}&=\max\{\mu_{11}+\mu_{12},\mu_{12}+\mu_{21},\mu_{21}+\mu_{22}\}:=M.\end{aligned}$$

$(X(t))_{t\geq 0}$ can be similarly uniformized with the uniformization constant

$$P:=\lambda_1+\lambda_2+\mu_{11}+\mu_{21}+\mu_{12}+\mu_{22}.$$

Then one step transition of $X(t)$ would result in

$$X(t+1)-X(t)=\begin{cases}1, & \text{if } Z(t)\in\left[0,\frac{\lambda_1+\lambda_2}{P}\right)\\ -1, & \text{if } Z(t)\in\left[\frac{\lambda_1+\lambda_2}{P},\frac{\lambda_1+\lambda_2+M}{P}\right)\\ 0, & \text{if } Z(t)\in\left[\frac{\lambda_1+\lambda_2+M}{P},1\right)\end{cases}.$$

If $X(0)=Q_1(0)+Q_2(0)$, we have $X(t)\leq Q_1(t)+Q_2(t)$ for all $t\geq 0$. To see this, assume that $X(t)\leq Q_1(t)+Q_2(t)$ and we will show that $X(t+1)\leq Q_1(t+1)+Q_2(t+1)$.

In all available service options, we have $s_1+s_2\leq M$ by the definition of M. Then at time $t\geq 0$,

- If
$$Z(t)\in\left[\frac{\lambda_1+\lambda_2+s_1+s_2}{P},\frac{\lambda_1+\lambda_2+M}{P}\right),$$
then
$$\begin{aligned}&Q_1(t+1)+Q_2(t+1)-Q_1(t)-Q_2(t)=0\\ &X(t+1)-X(t)=-1 \quad \text{if } s_1+s_2<M,\\ &X(t+1)-X(t)=0 \quad \text{if } s_1+s_2=M,\end{aligned}$$
ensuring that $X(t+1)\leq Q_1(t+1)+Q_2(t+1)$.

- If
$$Z(t)\notin\left[\frac{\lambda_1+\lambda_2+s_1+s_2}{P},\frac{\lambda_1+\lambda_2+M}{P}\right),$$
then $Q_1(t)+Q_2(t)$ evolves according to $Z(t)$ exactly as $X(t)$, so that $X(t+1)\leq Q_1(t+1)+Q_2(t+1)$.

Since $X(t)$ is not positive recurrent, $Q_1(t)+Q_2(t)$ is transient.

Then, the sufficient condition for instability is

$$\frac{\lambda_1 + \lambda_2}{\max\{\mu_{11} + \mu_{12}, \mu_{12} + \mu_{21}, \mu_{21} + \mu_{22}\}} \geq 1.$$

To see that there exists $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}$ with $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$ that satisfy this condition, assume that $\mu_{11} = \lambda_1 + \epsilon_1$ and $\mu_{22} = \lambda_2 + \epsilon_2$ for $\epsilon_1, \epsilon_2 > 0$. Then clearly $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$. If, in addition, $\mu_{22} \geq \mu_{12} + \epsilon_1 + \epsilon_2$ and $\mu_{11} \geq \mu_{21} + \epsilon_1 + \epsilon_2$, we have

$$\begin{aligned}
\mu_{11} + \mu_{22} &> \lambda_1 + \lambda_2, \\
\lambda_1 + \lambda_2 &= \mu_{11} + \mu_{22} - \epsilon_1 - \epsilon_2 \\
&\geq \max\{\mu_{11} + \mu_{12}, \mu_{12} + \mu_{21}, \mu_{21} + \mu_{22}\}.
\end{aligned}$$

The boundary cases where $Q_1(t)$ or $Q_2(t)$ is equal to zero can be handled similar to the proof of Theorem 1. □

The sufficient conditions in Proposition 2 and Corollary 2 being different indicates that these conditions may not be tight and there might exist other parameters for which the system is unstable. The characterization for the instability region according to Corollary 2 compared to Proposition 2 is given in Figure 2.6 in the case where $\mu_{11} + \mu_{12} > \mu_{21} + \mu_{22}$. The dark shaded region corresponds to the instability condition from Corollary 2 when $\mu_{11} + \mu_{12} > \mu_{21} + \mu_{22}$, and the red shaded region corresponds to the instability condition from Proposition 2. Depending on what function of the system trajectory we track (here, either $w_1\bar{Q}_1(t) + w_2\bar{Q}_2(t)$ or $Q_1(t) + Q_2(t)$), we might get different sufficient conditions for instability.

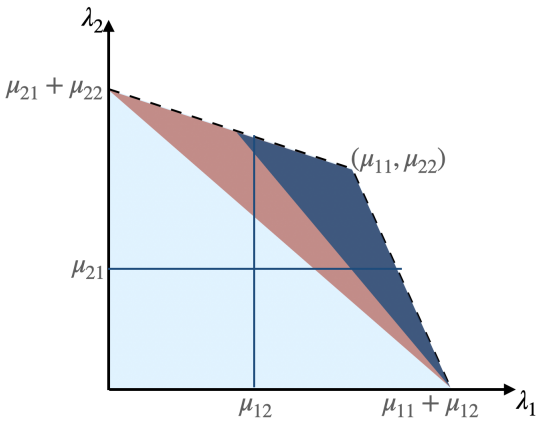


Figure 2.6: Comparison of Sufficient Conditions for Instability from Corollary 2 and Proposition 2

2.3.2 Stability Result

When the switching curves are not intersecting, one of them necessarily lies above the other after some $q_1^* \in \mathbb{Z}_+$. This is similar with MaxWeight, whose switching curves are actually switching lines of the form

$$\varphi_1^{MW}(q_1) = \frac{\mu_{11}}{\mu_{21}} q_1$$

and

$$\varphi_2^{MW}(q_1) = \frac{\mu_{21}}{\mu_{22}} q_1.$$

If the service rates are such that $\mu_{11}/\mu_{21} > \mu_{21}/\mu_{22}$, then $\varphi_1^{MW} > \varphi_2^{MW}$. Our stability result shows that if $\varphi_1 > \varphi_2$ where MaxWeight also suggests the same direction of the inequality, $\varphi_1^{MW} > \varphi_2^{MW}$, then the system is stable.

Theorem 2. *Let $\varphi_1(q_1) = m_1 q_1$ and $\varphi_2(q_1) = m_2 q_1$ for some $m_1, m_2 > 0$. If either*

1. *$m_1 > m_2$ and $\frac{\mu_{11}}{\mu_{21}} > \frac{\mu_{12}}{\mu_{22}}$, or*
2. *$m_1 < m_2$ and $\frac{\mu_{11}}{\mu_{21}} < \frac{\mu_{12}}{\mu_{22}}$,*

the system is stable for all $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$.

Proof. We will prove case (1), as the proof for case (2) will follow by replicating the steps symmetrically.

Similar to Proposition 2, a fluid limit exists and satisfies the fluid model equations 2.1-2.4. In addition, the policy specific fluid model equations hold.

$$\dot{\bar{T}}_{11}(t) = 1 \text{ and } \dot{\bar{T}}_{21}(t) = 0 \text{ if } m_1\bar{Q}_1(t) > \bar{Q}_2(t),$$

$$\dot{\bar{T}}_{21}(t) = 1 \text{ and } \dot{\bar{T}}_{11}(t) = 0 \text{ if } m_1\bar{Q}_1(t) < \bar{Q}_2(t), \tag{2.11}$$

$$\dot{\bar{T}}_{12}(t) = 1 \text{ and } \dot{\bar{T}}_{22}(t) = 0 \text{ if } m_2\bar{Q}_1(t) > \bar{Q}_2(t), \tag{2.12}$$

$$\dot{\bar{T}}_{22}(t) = 1 \text{ and } \dot{\bar{T}}_{12}(t) = 0 \text{ if } m_2\bar{Q}_1(t) < \bar{Q}_2(t).$$

The ties are broken arbitrarily. Note that 2.11 and 2.12 cannot hold at the same time because we assume $m_1 > m_2$.

We will utilize the fluid model to prove state space collapse in Lemma 1, then we'll use a Lyapunov function to show negative drift in the smaller state space in Lemma 2.

For the rest of the proof, assume $\lambda_1 < \mu_{11}$ and $\lambda_2 > \mu_{22}$. We will demonstrate how Lemmas 1 and 2 can be modified when this assumption is relaxed.

Lemma 1. *Let the distance of* $\bar{\mathbf{Q}}(t)$ *to the switching line* $\varphi_1(q_1) = m_1 q_1$ *be*

$$f(\bar{\mathbf{Q}}(t)) = \frac{|m_1\bar{Q}_1(t) - \bar{Q}_2(t)|}{\sqrt{m_1^2 + 1}}.$$

For any $\bar{\mathbf{Q}}(0)$, $f(\bar{\mathbf{Q}}(t))$ *decreases monotonically to zero in a finite time* $\tau_0 = \inf\{t \geq 0 : f(\bar{\mathbf{Q}}(t)) = 0\}$ *and* $f(\bar{\mathbf{Q}}(T)) = 0$ *for any* $T \geq \tau_0$.

Proof of Lemma 1. For any fluid model solution, $\bar{Q}_i(t)$ are Lipschitz continuous in t, then the distance function $f(\bar{\mathbf{Q}}(t))$ is also Lipschitz continuous in t.

Therefore, $f(\bar{\mathbf{Q}}(t))$ is absolutely continuous, then differentiable almost everywhere. For

any regular point t, we have

$$\frac{d}{dt}f(\bar{\mathbf{Q}}(t)) = \frac{1}{\sqrt{m_1^2+1}}\frac{m_1\bar{Q}_1(t)-\bar{Q}_2(t)}{|m_1\bar{Q}_1(t)-\bar{Q}_2(t)|}\left(m_1\dot{\bar{Q}}_1(t)-\dot{\bar{Q}}_2(t)\right). \tag{2.13}$$

According to the policy specific fluid model equations, at any time t, one of the following three cases hold:

$$\begin{aligned}
(i) \quad & \dot{\bar{T}}_{11}(t)=1,\ \dot{\bar{T}}_{21}(t)=0,\ \dot{\bar{T}}_{12}(t)=1,\ \dot{\bar{T}}_{22}(t)=0,\\
(ii) \quad & \dot{\bar{T}}_{11}(t)=1,\ \dot{\bar{T}}_{21}(t)=0,\ \dot{\bar{T}}_{22}(t)=1,\ \dot{\bar{T}}_{12}(t)=0,\\
(iii) \quad & \dot{\bar{T}}_{21}(t)=1,\ \dot{\bar{T}}_{11}(t)=0,\ \dot{\bar{T}}_{22}(t)=1,\ \dot{\bar{T}}_{12}(t)=0.
\end{aligned}$$

Then for each of the conditions above, equation 2.13 becomes

$$\begin{aligned}
(i) \quad & \frac{d}{dt}f(\bar{\mathbf{Q}}(t)) = \frac{1}{\sqrt{m_1^2+1}}(m_1(\lambda_1-\mu_{11}-\mu_{12})-\lambda_2)<0,\\
(ii) \quad & \frac{d}{dt}f(\bar{\mathbf{Q}}(t)) = \frac{1}{\sqrt{m_1^2+1}}(m_1(\lambda_1-\mu_{11})-(\lambda_2-\mu_{22}))<0,\\
(iii) \quad & \frac{d}{dt}f(\bar{\mathbf{Q}}(t)) = \frac{-1}{\sqrt{m_1^2+1}}(m_1\lambda_1-(\lambda_2-\mu_{21}-\mu_{22}))<0.
\end{aligned}$$

Therefore, the time derivative of the distance to the switching line is always a negative constant. Then the fluid trajectory collapses onto the switching line $\varphi_1(q_1) = m_1 q_1$ in a finite time $\tau_0 = \{t \geq 0 : f(\bar{\mathbf{Q}}(t)) = 0\}$. Since $f(\bar{\mathbf{Q}}(t))$ is absolutely continuous, for any $T \geq \tau_0$,

$$f(\bar{\mathbf{Q}}(T)) - f(\bar{\mathbf{Q}}(\tau_0)) = \int_{\tau_0}^{T}\frac{d}{dt}f(\bar{\mathbf{Q}}(t))dt.$$

Because distance is always non-negative, we have

$$0 \leq f(\bar{\mathbf{Q}}(T)) \leq f(\bar{\mathbf{Q}}(\tau_0)) + \int_{\tau_0}^{T} \frac{d}{dt} f(\bar{\mathbf{Q}}(t))dt \leq 0,$$

implying that for any $T \geq \tau_0$, $f(\bar{\mathbf{Q}}(T)) = 0$. □

Lemma 2. *The fluid limit model is stable when* $f(\bar{\mathbf{Q}}(t)) = 0$ *for* $t \geq \tau_0$.

Proof of Lemma 2. For all $t \geq 0$, let the candidate Lyapunov function be

$$\begin{aligned} V(\bar{\mathbf{Q}}(t)) &:= \bar{Q}_1(t) + w_2 \bar{Q}_2(t), \text{ where} \\ w_2 \in \mathcal{I} &:= \left(\frac{\lambda_1}{\mu_{21} + \mu_{22} - \lambda_2}, \frac{\mu_{11} - \lambda_1}{\lambda_2 - \mu_{22}} \right). \end{aligned}$$

The interval $\mathcal{I}$ is non-empty. To see this, assume to the contrary that $\lambda_1/(\mu_{21} + \mu_{22} - \lambda_2) \geq (\mu_{11} - \lambda_1)/(\lambda_2 - \mu_{22})$. Then

$$\begin{aligned} \frac{\lambda_1}{\mu_{21} + \mu_{22} - \lambda_2} &\geq \frac{\mu_{11} - \lambda_1}{\lambda_2 - \mu_{22}}, \\ \lambda_1 &\geq (\mu_{21} + \mu_{22}) \frac{\mu_{11}}{\mu_{21}} - \lambda_2 \frac{\mu_{11}}{\mu_{21}}. \end{aligned}$$

By the capacity region definition, for some $p, q \in [0, 1]$, the inequalities $\lambda_1 < p\mu_{11} + q\mu_{12}$ and $\lambda_2 < (1-p)\mu_{21} + (1-q)\mu_{22}$ hold. Using the latter inequality, the expression becomes

$$\begin{aligned} \lambda_1 &> (\mu_{21} + \mu_{22}) \frac{\mu_{11}}{\mu_{21}} - \left[(1-p)\mu_{21} + (1-q)\mu_{22} \right] \frac{\mu_{11}}{\mu_{21}}, \\ &= p\mu_{11} + q \frac{\mu_{11}\mu_{22}}{\mu_{21}}, \\ &> p\mu_{11} + q\mu_{12}, \end{aligned}$$

which is a contradiction with $\lambda_1 < p\mu_{11} + q\mu_{12}$. Here, the last inequality follows from the assumption $\mu_{11}/\mu_{21} > \mu_{12}/\mu_{22}$. Therefore, we must have $\lambda_1/(\mu_{21} + \mu_{22} - \lambda_2) <$

$(\mu_{11} - \lambda_1)(\lambda_2 - \mu_{22})$, implying that $\mathcal{I}$ is non-empty.

It is clear that $V(\bar{\mathbf{Q}}(t)) \neq 0$ when $\bar{\mathbf{Q}}(t) \neq 0$. We will show $\frac{d}{dt}V(\bar{\mathbf{Q}}(t)) < 0$ for $t \geq \tau_0$.

$$\frac{d}{dt}V(\bar{\mathbf{Q}}(t)) = \lambda_1 - \mu_{11}\dot{\bar{T}}_{11}(t) - \mu_{12}\dot{\bar{T}}_{12}(t) + w_2\left(\lambda_2 - \mu_{21}\dot{\bar{T}}_{21}(t) - \mu_{22}\dot{\bar{T}}_{22}(t)\right).$$

For $t \geq \tau_0$, the trajectory is on the switching line $\varphi_1(\bar{Q}_1(t)) = \bar{Q}_2(t)$. Then $\dot{\bar{T}}_{12}(t) = 0$ and $\dot{\bar{T}}_{22}(t) = 1$. Thus, we have

$$\frac{d}{dt}V(\bar{\mathbf{Q}}(t)) = \lambda_1 + w_2(\lambda_2 - \mu_{22}) - \left(\mu_{11}\dot{\bar{T}}_{11}(t) + w_2\mu_{21}\dot{\bar{T}}_{21}(t)\right).$$

Note that $\dot{\bar{T}}_{11}(t) + \dot{\bar{T}}_{21}(t) = 1$ due to the non-idling assumption of the fluid model. Then $(\mu_{11}\dot{\bar{T}}_{11}(t) + w_2\mu_{21}\dot{\bar{T}}_{21}(t))$ is a convex combination of μ_{11} and $w_2\mu_{21}$, therefore takes a value between these terms. For $w_2 \in \mathcal{I}$,

$$\lambda_1 + w_2(\lambda_2 - \mu_{22}) - \mu_{11} < 0,$$
$$\lambda_1 + w_2(\lambda_2 - \mu_{22}) - w_2\mu_{21} < 0,$$

implying that we have $\frac{d}{dt}V(\bar{\mathbf{Q}}(t)) < 0$. Hence, the fluid limit model is stable. □

We assumed for Lemmas 1 and 2 that $\lambda_1 < \mu_{11}$ and $\lambda_2 > \mu_{22}$. Note that the analysis still valid if either $\lambda_1 = \mu_{11}$ and $\lambda_2 > \mu_{22}$, or $\lambda_1 < \mu_{11}$ and $\lambda_2 = \mu_{22}$. Only one of the inequalities must be strict.

For $\lambda_1 > \mu_{11}$ and $\lambda_2 < \mu_{22}$, the trajectory $\bar{\mathbf{Q}}(t)$ approaches the switching line $\varphi_2(q_1) = m_2 q_1$ and stays there. The same Lyapunov function $V(\bar{\mathbf{Q}}(t)) = \bar{Q}_1(t) + w_2\bar{Q}_2(t)$ with appropriately chosen w_2 has negative drift on the line $\varphi_2(\bar{Q}_1(t)) = \bar{Q}_2(t)$, proving the stability of the fluid limit model.

The case $\lambda_1 \geq \mu_{11}$ and $\lambda_2 \geq \mu_{22}$ does not need to be considered, because there does not exist such $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$ that also satisfy the assumption $\mu_{11}/\mu_{21} > \mu_{12}/\mu_{22}$. Indeed, the

capacity region inequalities and $\lambda_1 \geq \mu_{11}$ and $\lambda_2 \geq \mu_{22}$ imply that

$$\mu_{11} \leq \lambda_1 < p\mu_{11} + q\mu_{12} \implies \frac{\mu_{11}}{\mu_{12}} < \frac{q}{1-p},$$
$$\mu_{22} \leq \lambda_2 < (1-p)\mu_{21} + (1-q)\mu_{22} \implies \frac{q}{1-p} < \frac{\mu_{21}}{\mu_{22}},$$

contradicting $\mu_{11}/\mu_{21} > \mu_{12}/\mu_{22}$.

The only other case that needs to be considered is $\lambda_1 < \mu_{11}$ and $\lambda_2 < \mu_{22}$. It can be shown that the Lyapunov function $V(\bar{\mathbf{Q}}(t)) = \bar{Q}_1(t) + w_2\bar{Q}_2(t)$ with

$$w_2 \in \mathcal{J} := (\lambda_1/(\mu_{21} - \mu_{22} - \lambda_2),\ (\mu_{11} + \mu_{12} - \lambda_1)/\lambda_2)$$

has negative drift at every regular point, following the same steps in Lemma 2.

By Theorem 4.2 in Dai (1995), the stability of the fluid limit model implies the positive recurrence of the Markov chain describing the dynamics of the X-model. □

Theorem 2 eliminates the need for precision when learning service rates μ_{ij}. If it can be reasonably estimated whether $\mu_{11}/\mu_{21} > \mu_{12}/\mu_{22}$, then a switching line policy can be designed with $m_1 > m_2$ to achieve maximal stability.

This theorem has an interesting practical implication as well. Imagine that the system designer has the intention of building a dedicated system with a little flexibility. That is, the intention is to have server 1 serve queue 1 efficiently, with the ability to serve queue 2 less efficiently. Similarly, server 2 can be designed to serve queue 2 efficiently and queue 1 less efficiently. Theorem 2 then implies that a parameter agnostic switching line policy with $m_1 > m_2$ is maximally stable.

2.4 One Switching Curve for Both Servers

If $\varphi_1(q_1) = \varphi_2(q_1)$ for all $q_1 \in \mathbb{Z}_+$, then there is essentially one switching curve that dictates the scheduling decision. Such a policy implies that the servers are synchronized, they serve the same queue at the same time.

Proposition 3. *Let $\varphi_1(q_1) = \varphi_2(q_1)$ for all $q_1 \in \mathbb{Z}_+$. For $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$, if*

$$\frac{\lambda_1}{\mu_{11} + \mu_{12}} + \frac{\lambda_2}{\mu_{21} + \mu_{22}} > 1,$$

then the system is unstable.

Proof of Proposition 3. Similar to Proposition 2, a fluid limit exists and satisfies the fluid model equations 2.1-2.4. In addition, the following policy specific fluid model equations hold.

$$\bar{T}_{11}(t) = \bar{T}_{12}(t),$$
$$\bar{T}_{21}(t) = \bar{T}_{22}(t).$$

Then, according to the fluid model equations, we have

$$\bar{T}_{11}(t) + \bar{T}_{21} \leq t, \tag{2.14}$$
$$\bar{T}_{12}(t) + \bar{T}_{22} \leq t. \tag{2.15}$$

Let $\bar{p}(t)$ denote the total fraction of time up to t that the servers serve queue 1. That is,

$$\bar{p}(t) := \frac{\bar{T}_{11}(t)}{t} = \frac{\bar{T}_{12}(t)}{t},$$

which implies by equations 2.14 and 2.15 that

$$\frac{\bar{T}_{21}(t)}{t} = \frac{\bar{T}_{22}(t)}{t} \leq 1 - \bar{p}(t).$$

Note that $\bar{p}(t) \in [0, 1]$. Then we can write the first fluid model equation, Equation 2.1 as

$$\bar{Q}_1(t) = \bar{Q}_1(0) + \lambda_1 t - \bar{p}(t)t\mu_{11} - \bar{p}(t)t\mu_{12},$$
$$\bar{Q}_2(t) \geq \bar{Q}_2(0) + \lambda_2 t - (1 - \bar{p}(t))t\mu_{21} - (1 - \bar{p}(t))t\mu_{22}.$$

To use Theorem 3.2 in Meyn et al. (1995), we define the norm $||\bar{\mathbf{Q}}(t)|| = \bar{Q}_1(t) + \bar{Q}_2(t)$. For any t, if $\bar{p}(t) < \frac{\lambda_1}{\mu_{11}+\mu_{12}}$, we have

$$||\bar{\mathbf{Q}}(t)|| = \bar{Q}_1(0) + t(\lambda_1 - \bar{p}(t)(\mu_{11} + \mu_{12})) + \bar{Q}_2(t) \geq \delta_1 t,$$

where $\delta_1 = \lambda_1 - \bar{p}(t)(\mu_{11} + \mu_{12}) > 0$.

If $\bar{p}(t) > 1 - \frac{\lambda_2}{\mu_{21}+\mu_2}$, we have

$$||\bar{\mathbf{Q}}(t)|| \geq \bar{Q}_1(t) + \bar{Q}_2(0) + t(\lambda_2 - (1 - \bar{p}(t))(\mu_{21} + \mu_{22})) \geq \delta_2 t,$$

where $\delta_2 = \lambda_2 - (1 - \bar{p}(t))(\mu_{21} + \mu_{22}) > 0$.

We had assumed

$$\frac{\lambda_1}{\mu_{11} + \mu_{12}} + \frac{\lambda_2}{\mu_{21} + \mu_{22}} > 1,$$

implying that for all $\bar{p}(t) \in [0, 1]$, the norm can be lower bounded as $||\bar{\mathbf{Q}}(t)|| \geq \min\{\delta_1, \delta_2\}t$.

Then by Theorem 3.2 in Meyn et al. (1995), the Markov Chain $(\mathbf{Q}(t))_{t\geq 0}$ is transient. □

2.5 Discussion and Conclusion

We have considered parameter agnostic policies in the context of parallel server systems. We have shown that in general, parameter agnostic policies are not maximally stable.

When the policy consists of intersecting switching curves, we have shown in Theorem 1 that the X-model couples with a single dimensional Markov Chain. Under the condition that the service rate smaller than the arrival rate is used more often than the service rate greater than the arrival rate on average, this single dimensional Markov Chain cannot be positive recurrent. Because the X-model couples with the single dimensional Markov Chain, it cannot have a stationary distribution either. This result indicates that the three service regimes that MaxWeight uses are critical. If the fourth service regime is introduced and used more frequently than the "correct one" according to MaxWeight, the system becomes unstable. In a sense, MaxWeight is tight in achieving maximal stability while requiring minimal information on system parameters.

Even though MaxWeight is maximally stable itself, using the slopes implied by it, $\frac{\mu_{11}}{\mu_{21}}$ and $\frac{\mu_{12}}{\mu_{22}}$, is not strictly necessary for maximal stability. If the system designer only knows the service configuration implied by the MaxWeight slopes, she can devise a switching line policy with slopes m_1, m_2 that implies the same service configuration with MaxWeight slopes, and achieve maximal stability. We show this in Theorem 2 by proving state space collapse to one of the switching lines, then showing negative drift of the fluid model. This result is especially insightful for systems where each server is trained specifically to serve a single class of jobs with the capability of handling the other when necessary. In such cases, if the system designer is reasonably confident that the servers serve their intended class more efficiently, $\mu_{11}/\mu_{21} > \mu_{12}/\mu_{22}$ naturally holds. Then a parameter agnostic policy with $m_1 > m_2$ is maximally stable. Our intuition tells us that the stability result would hold for switching curves under mild conditions such as the existence of a fluid limit.

Imposing more conditions on the switching curves and assuming that the curves do not

intersect, we are able to obtain instability with easier arguments. We show in Proposition 2 that $w_1\bar{Q}_1(t)+w_2\bar{Q}_2(t)$ increases linearly in time for appropriately chosen weights $w_1, w_2 > 0$.

If both servers have the same switching curve, such as LQF where the switching curves are $\varphi_1(q_1) = \varphi_2(q_1) = q_1$, the system is again not maximally stable. We show in Proposition 3 that $\bar{Q}_1(t) + \bar{Q}_2(t)$ grows linearly in t.

In general, even though parameter agnostic policies are attractive in practice, a system designer should not implement them without any knowledge of the system itself, qualitative or quantitative. This is because these policies can become unstable if the system parameters turn out to satisfy certain sufficient conditions. Even a little knowledge of the system, such as whether the servers are specifically trained to handle one job class with the capability to help the other when necessary, closes this gap significantly and enables the system designer to adopt a maximally stable parameter agnostic policy.

There are natural future research directions that can follow our results. For instance, generalizing the stability result to bigger parallel server systems would provide insights into what kinds of knowledge on system parameters would suffice in ensuring stability. Another more challenging direction would be to remove all assumptions on structure of the policy, and disproving maximal stability for arbitrary parameter agnostic policies.

We believe wholly understanding parameter agnostic policies, not only in terms of stability but also in other performance metrics, will allow system designers to make informed decisions on whether they want to dedicate resources to learning the system parameters.

CHAPTER 3

PROCESS FLEXIBILITY DESIGN FOR PARALLEL SERVER SYSTEMS WITH GENERAL PROCESSING RATES

3.1 Introduction

Companies nowadays are increasingly expected to expand their product portfolios. This entails offering customers a wider range of choices in order to maintain a competitive edge. Efficient management of a wider product range requires meticulous operational planning, including strategies to mitigate the impact of unforeseen disruptions to the supply chain.

The need to respond adeptly to external shocks is not limited to physical goods alone; it extends to service-oriented systems as well. Service providers must maintain high-quality service even in the face of unexpected fluctuations in labor availability or service demand. Similarly, computing environments, such as cloud platforms, are expected to handle surges in job requests efficiently and without delays.

To tackle this challenge, one effective approach is to integrate process flexibility into the underlying networks of these operations. By embracing process flexibility, companies can pool resources and swiftly adapt to unforeseen changes without compromising service levels.

Typically, the most desired form of process flexibility is *full flexibility*, where every server, agent or plant in the network can handle any job, customer, or product type. However, embedding such a high degree of process flexibility is usually costly. Additionally, majority of the resource pooling benefits of full flexibility have been observed in systems with significantly less degree of process flexibility, implying that constructing full flexibility may not be necessary at all. Jordan and Graves (1995) led research in this area by observing that a simple sparse design called *the long chain* achieves most of the benefits of an equivalent fully flexible system. This indicates that by adopting a systematic approach, we can almost attain the pooling benefits of a fully flexible system with a sparser flexibility design.

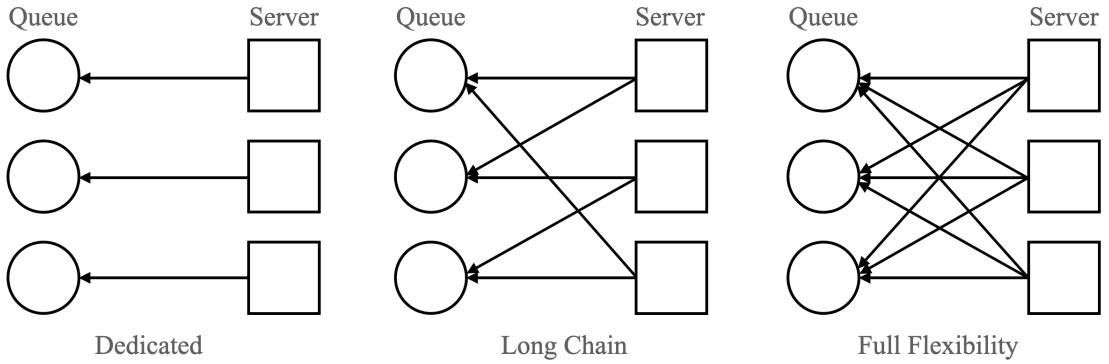


Figure 3.1: Examples of Flexibility Structures

Indeed, extensive research, both theoretical and empirical, has delved into studying various flexibility designs (e.g., Chou et al. (2011), Simchi-Levi and Wei (2012), Wang and Zhang (2015), Désir et al. (2016), Shi et al. (2019)). Figure 3.1 shows graph representations of some well known flexibility structures, including the long chain. The central goal of the research in this area has mostly been to design flexibility structures with performance close to that of the full flexibility structure. Similarly, analysis of given flexibility structures focuses on comparing the performance of a given flexibility structure to that of the full flexibility structure. In this chapter, we address the problem of designing a sparse flexibility structure that has performance close to that of the full flexibility structure. Our model allows for jobs to wait in their queues until they get served, so that we consider multi-period settings, which distinguishes this work from most of the papers in the literature that focus on a single-period setting (Chou et al. 2010, 2011, Simchi-Levi and Wei 2012, Chen et al. 2015, Wang and Zhang 2015, Désir et al. 2016). There are recent papers in the literature that explore the problem in a multi-period setting (e.g., Shi et al. (2019), Asadpour et al. (2020), Varma and Maguluri (2021)). However, our work follows a fundamentally different approach than these. In particular, this is the first attempt to address the design problem while relaxing the widely-adopted flow conservation assumption put forth by Jordan and Graves (1995).

The critical flow conservation assumption that all previous papers in this area rely on was introduced in the context of the long chain, suggested first by Jordan and Graves (1995). All

theoretical and empirical analysis of the long chain has been conducted under the assumption that one unit of server capacity is required to process one unit of a compatible job. This assumption is not unique to the analysis of long chain. Any flexibility structure that has been studied thereafter was studied under the assumption that the servers have a fixed capacity, one unit of which can process one unit of a compatible job.

Relaxing this crucial assumption, we consider servers with general processing rates. In other words, one unit of server capacity may process more or less than one unit of a compatible job. The exact rate at which one unit of capacity processes a compatible job depends on the server-buffer pair. This is an important relaxation of the longstanding assumption, because flow conservation might not be observed in practice. In production systems, certain plants may deliver fewer units of a certain product per unit time, potentially due to physical or logistical constraints. Similar limitations can arise in service systems. For example, call center agents may possess the capability to assist various customer types, but at varying speeds. It might be the case in a service system that service agents primarily focus on one type of service request, but they can handle other types if necessary, albeit at reduced processing rates. Due to all these possibilities, study of flexibility designs in systems with general processing rates holds significant importance.

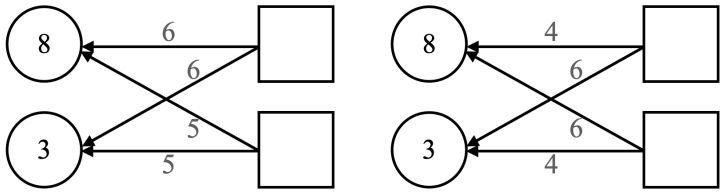
Relaxation of the flow conservation assumption results in substantial difficulties in the design and analysis of flexibility graphs. While it is algorithmically easy to obtain a feasible processing plan in systems with flow conservation, obtaining a feasible solution becomes a non-trivial concern in systems with general processing rates. In particular, greedily solving a maximum flow problem on the fully flexible system always yields a feasible sparse subgraph of the fully flexible system when the system has flow conservation (see, e.g., Shi et al. (2019)). On the other hand, when processing rates are general, each server is associated with multiple service rates, so that greedily solving a maximum flow problem on the fully flexible system would not always result in a feasible sparse subgraph. We would encounter supply or demand

deficiencies as the greedy algorithm moves towards the ending nodes of the graph.

For example, consider the system given in Figure 3.2. In Figure 3.2a, the first plant serves both products with a rate of 6, and the second plant serves both products with a rate of 5. The first product has a demand rate of 8, and the second product has a total demand rate of 3. Total service rate of the plants is 11, which is equal to the total demand rate. A greedy solution to the max-flow problem on the fully flexible model is feasible, as given in Figure 3.3a. The first plant sends 6 units of product to the first demand node. The second plant dedicates 0.4 of its capacity to the first demand node, i.e., sends 2 units to the first demand node. It dedicates the remaining 0.6 of its capacity to the second plant node, i.e., sends 3 units to the second plant node. This results in a remaining demand (or imbalance) of zero units in both demand nodes. Because there is no imbalance at any of the nodes, the design implied by the greedy solution is feasible.

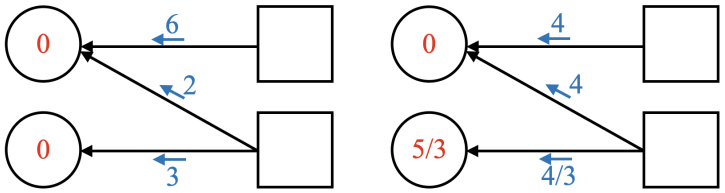
On the other hand, Figure 3.2b demonstrates almost the same problem, but with general processing rates. Now, the first plant serves the first product with a rate of 4, and serves the second product with a rate of 6. Similarly, the second plant serves the first product with a rate of 6, and the second product with a rate of 4. The demand rates are the same as the previous case. It can be seen that there exists a feasible solution that results in no imbalance at any of the nodes. If the first server dedicates 0.5 of its capacity to the first product and the remaining 0.5 of its capacity to the second product; and the second server dedicates all of its capacity to the first product, there would be no imbalance at any of the nodes and the system would be feasible. However, if we wanted to greedily solve a max-flow problem, the second demand node would have a deficiency of 5/3 units, as Figure 3.3b demonstrates.

This simple example showcases the aforementioned difficulty in systems with general processing rates: feasibility is not straightforward to attain. Another difficulty we encounter with general processing rates stems from the inherent limitations of these systems. For a given flexibility graph, a flexible server system that operates across multiple periods can



(a) Server Dependent Rates (b) General Processing Rates

Figure 3.2: Processing Rates and Demand Rates on Fully Flexible Graph



(a) Server Dependent Rates (b) General Processing Rates

Figure 3.3: Greedy Max-Flow Solutions on Fully Flexible Graph

be modeled as a discrete-time parallel server system (see, e.g., Harrison and López (1999), Mandelbaum and Stolyar (2004), Stolyar et al. (2004), Gurvich and Whitt (2009)). If the so-called complete resource pooling (CRP) condition holds for parallel server systems operating under heavy traffic, it is expected that numerous scheduling policies should achieve good performance. CRP is interpreted as the system behaving as if the servers form a single resource pool in heavy traffic.

While CRP condition always holds in fully flexible systems with flow conservation, it may not always hold in fully flexible systems with general processing rates. In the context of parallel server systems, the flow conservation assumption can be interpreted as servers having server dependent service rates. That is, each server can serve all compatible queues with the same processing rate. When service rates are server dependent, parallel server systems always achieve a single resource pool in heavy traffic, i.e., CRP. Then the goal becomes achieving CRP with a sparser design. On the other hand, in the case of general processing rates, a fully flexible system may not achieve CRP for certain system parameters under

heavy traffic. Consequently, any sparse design cannot achieve a single resource pool. This limitation is intrinsic to systems with general processing rates. Recognizing this distinction is crucial when designing a sparse flexibility system with good performance, as heavy traffic analyses of control policies in queueing systems heavily rely on it.

In this chapter, we show the existence of a sparse flexibility structure with $O(m+n)$ connections that achieves a single resource pool in heavy traffic, where a single resource pool is achievable in the fully flexible system. Here, m is the number of servers and n is the number of buffers. The difficulties described above results in our methodology being significantly different than the existing literature. In particular, we heavily utilize a well-known linear program (LP), the static planning problem (SPP). We show that if a single resource pool is attainable in the fully flexible system, then the SPP has a non-denegerate optimal solution. We start with an optimal basic solution of the SPP, and augment it with alternative solutions of the SPP until a non-degenerate (possibly non-basic) optimal solution is attained. The resulting solution translates to a connected feasible subgraph of the fully flexible system, which achieves a single resource pool in heavy traffic and has $O(m+n)$ arcs. This approach not only provides valuable insights into the design and analysis of flexibility structures, but also marks the first step in addressing the challenges posed by general processing rates.

The rest of the chapter is organized as follows. In Section 3.1.1, we provide a brief literature review on process flexibility, and performance of different control policies in parallel server systems. In Section 3.2, we outline our model, notation and relevant definitions. In Section 3.3, we introduce the SPP and its dual, and discuss important distinctions regarding uniqueness and multiplicity of the solutions. We also derive useful properties of the two problems. In Section 3.4, we introduce our main result, that there exists a sparse design that achieves CRP in heavy traffic. We present an algorithm that achieves that design. In Section 3.5, we numerically demonstrate the performance of our proposed design compared

to various flexibility designs under MaxWeight. Finally, we discuss future research directions and conclude in Section 3.6.

3.1.1 Literature Review

3.1.1.1 Process Flexibility

Design and analysis of process flexibility structures started with the seminal paper of Jordan and Graves (1995), in which they introduced the long chain. Since then, the performance of the long chain has been theoretically justified. Chou et al. (2010) derived the ratio of the performance of the long chain and the performance of the fully flexible system as the system size grows. They showed that when the demand follows certain distributions like the uniform and the normal distributions, this ratio approaches 1. Simchi-Levi and Wei (2012) demonstrated that when building a long chain from a dedicated system, the biggest benefit is achieved by the arc that closes the chain. They also showed that the performance of the long chain relative to the fully flexible system decreases with the size of the system. Wang and Zhang (2015) developed a model to analyze the performance of any bipartite network compared with the full flexibility system under a distributionally robust setting. They obtained a bound on the performance of the long chain that depends only on the first two moments of the demand distribution, implying that the performance of the long chain becomes worse when the demand has more variability. Désir et al. (2016) proved that the long chain is optimal among all designs that use $2n$ arcs, where $2n$ is the total number of nodes on the graph.

Flexibility designs other than the long chain have been proposed and studied as well. Chou et al. (2011) showed that under the assumption that the demand distribution has a bounded variation of 1, the class of graph expanders (a class of sparse but highly connected graphs) perform well compared to the fully flexible system. They show that there exists a class of expanders with $O(n/\epsilon)$ arcs that achieves $(1-\epsilon)$-optimality in the worst case.

Chen et al. (2015) improved this using probabilistic graph expanders, achieving the same performance with a sparser graph with $O(1/\epsilon)$ arcs in balanced and symmetrical systems. Later on, the authors generalized this result to unbalanced and asymmetric systems (Chen et al. 2019). Simchi-Levi and Wei (2015) study the design problem under a worst-case metric, and propose a simple and implementable class of heuristic algorithms to design flexibility graphs.

Most of these works were studied under the model of Jordan and Graves (1995), which considers a single period setting. The problem remains relatively unexplored for multi-period systems. Shi et al. (2019) studied a multi-period system, and showed that $m+n$ are necessary to ensure that a sparse design can asymptotically attain the performance of a fully flexible system. They introduced the concept of generalized chaining gap (GCG), which can be thought of as the minimum excess capacity in the system. GCG being positive is equivalent to the CRP condition. Following up from Shi et al. (2019), Varma and Maguluri (2021) characterized exactly when $m+n$ arcs are necessary and when $m+n-1$ arcs are sufficient to ensure satisfactory performance. They utilized the connection between parallel server systems and the transportation problem, and explored the design problem when CRP does not necessarily hold. Independently, Asadpour et al. (2020) derived what they call the ξ-hall condition for balanced systems, which is equivalent to CRP condition when the system is under heavy traffic. Their model is such that resource capacities are depleted over time. Xu et al. (2020) generalized their policy and improved the upper bound on expected total lost sales.

On the analysis side of multi-period systems, Iravani et al. (2007) propose a new metric of evaluating and comparing the performance of two arbitrarty flexible server systems and show through extensive simulations that this method can be used to distinguish which of the two designs is better. Tsitsiklis and Xu (2017) prove that a family of expander-graph-based flexibility designs achive diminishing queue delays as the system size grows to infinity.

A more in-depth treatment of these topics is presented in Wang et al. (2021), where the authors conduct an extensive recent review of the literature on process flexibility design and analysis.

Here, similar to Shi et al. (2019), Asadpour et al. (2020), Varma and Maguluri (2021), we consider a multi-period setting, where the system can be modeled as a discrete time parallel server system. Our approach is distinguished from the literature because we are the first to address the flexibility design problem while relaxing the flow conservation assumption that is adopted in the previous works. This approach results in our design method being fundamentally different than the literature, which contributes to the theoretical understanding of flexibility designs, the structural properties of the SPP for parallel server systems, and the relationship between parallel server systems and generalized transportation problems.

3.1.1.2 Performance of Control Policies

Given a flexibility design, the performance of a discrete time parallel server system is determined by the scheduling policy, which dictates which compatible queue a server will serve when it becomes idle. Given the flexibility design and the scheduling policy, the performance of the system is usually measured by a weighted sum of queue lengths, called the cost function. This type of analysis is typically done in heavy traffic and the asymptotic optimality of such policies has been the main interest, since the pre-limit formulation is often intractable. An additional assumption that greatly simplifies the analysis is CRP, which asserts that the system achieves a single resource pool in heavy traffic. That is, servers behave like a single super-server in serving the jobs in heavy traffic.

Traditionally, the heavy traffic assumption asserts that the SPP associated with the given system has a unique solution. Under the traditional heavy traffic assumption and the CRP condition, Stolyar et al. (2004) showed that the MaxWeight policy is asymptotically optimal in minimizing the long run average of a specifically weighted sum of queue lengths. Under

the same assumptions, Mandelbaum and Stolyar (2004) showed that a generalized $c\mu$-rule is asymptotically optimal in minimizing a strictly convex cost function of queue lengths. Again, under traditional heavy traffic and CRP assumptions, Bell and Williams (2005) showed that an intricate threshold policy is asymptotically optimal in minimizing the long run discounted sum of arbitrarily weighted sum of queue lengths.

In general, there is no reason to assume that the SPP has a unique solution. However, not much has been theoretically shown when the SPP has multiple solutions, and the technical difficulties for this case have been acknowledged (Harrison and López 1999, Mandelbaum and Stolyar 2004). Eryilmaz and Srikant (2012) has shown that MaxWeight is asymptotically optimal under heavy traffic and CRP, without imposing the condition that the SPP has a unique solution. However, the optimality of MaxWeight holds in that case under the condition that the service times of the servers are deterministic. Atar et al. (2022) has started exploring the problem of asymptotic optimality of scheduling policies when the SPP has multiple solutions.

The design problem is tightly connected to performance analysis problem. When the SPP for the fully flexible system has a unique solution, results of Harrison and López (1999) has implications for a reasonable design. Specifically, if the unique solution to the SPP for the fully flexible system is non-degenerate, then the dual of the SPP for the implied sparse design has a unique solution. Then the CRP condition holds for the implied design, and the traditional performance analysis results hold. That is, the design implied by the unique solution would be a reasonable design.

On the other hand, if the SPP for the fully flexible system has multiple solutions, the design implied by a solution to that SPP might not necessarily result in a unique solution to the dual of the SPP for the implied design. In such a case, CRP fails for the implied design. Then the system cannot achieve a single resource pool, even if a single resource pool was achievable with the fully flexible system. As a result, the implied design, which has multiple

resource pools, would have significantly worse performance than the full flexibility design, which achieves a single resource pool. The central goal of the flexibility design literature, finding a sparse design with performance close to that of a fully flexible system, would fail. Precisely in this case, we want to find a solution to the SPP for the fully flexible system, such that the design implied by that solution is sparse and has a unique solution for its associated dual SPP. In that case, our sparse design will satisfy CRP and we expect good performance from various scheduling policies.

Even though we know that MaxWeight is asymptotically optimal under deterministic service times when the SPP has multiple solutions and CRP holds, the performance analysis of alternative scheduling policies for alternative cost functions remains an open problem. Here, we address the problem of effectively designing a sparse flexibility graph with good properties. Even though we numerically justify the performance of our design compared to the full flexibility system under MaxWeight, we leave the theoretical performance analysis of alternative scheduling policies as future research directions.

3.2 Model, Definitions and Notations

The convention we adopt in this paper is to denote real numbers by $\mathbb{R}$, and non-negative real numbers by $\mathbb{R}_+$. Integers and non-negative integers are denoted by $\mathbb{Z}$ and $\mathbb{Z}_+$, respectively. The vectors of all ones and all zeros are denoted by $\mathbf{1}$ and $\mathbf{0}$, respectively. Boldface letters denote vectors and unbold letters denote scalars.

We consider a discrete time parallel server system with $m \geq 1$ servers and $n \geq 1$ queues where time periods are denoted by $t \in \mathbb{Z}_+$. At the beginning of each time period t, each queue $i \in \mathcal{I} := \{1, \ldots, n\}$ observes a random number of arrivals denoted by $A_i(t)$. For each $i \in \mathcal{I}$, $A_i(t)$ is distributed independently and identically across time periods. Then A_i denotes the arrival distribution to queue i for every $t \in \mathbb{Z}_+$. We assume that $\mathbb{E}[A_i] = \lambda_i > 0$, $Var(A_i) = \sigma_i^2$, and there exists $u > 0$ such that $P(0 \leq A_i \leq u) = 1$.

After observing arrivals, each server $j \in \mathcal{J} := \{1, \ldots, m\}$ serves a compatible queue i with a rate of $\mu_{ij} > 0$, deterministically. The deterministic service assumption might seem restrictive. However, in certain practical systems, this may not be unreasonable. In production systems, for instance, jobs may be processed in batches, in which case service resembles a deterministic process.

The compatibility of servers and queues is determined by the system's *flexibility structure*, denoted by $\mathcal{S}$. Server $j \in \mathcal{J}$ can serve queue $i \in \mathcal{I}$ only if $(i, j) \in \mathcal{S}$. That is, the flexibility structure $\mathcal{S}$ is the arc set of a bipartite graph whose node partition is $\mathcal{I}$ and $\mathcal{J}$. If all servers can serve all queues, we call this special system the *fully flexible system*, denoted by $\mathcal{F} := \mathcal{I} \times \mathcal{J}$.

The neighborhood function on $\mathcal{S}$ is the mapping that returns the set of nodes adjacent to each other in $\mathcal{S}$. That is,

$$N_{\mathcal{S}}(i) = \{j \in \mathcal{J} : (i, j) \in \mathcal{S}\} \quad \forall i \in \mathcal{I},$$
$$N_{\mathcal{S}}(j) = \{i \in \mathcal{I} : (i, j) \in \mathcal{S}\} \quad \forall j \in \mathcal{J}.$$

We note that in the fully flexible structure $\mathcal{F}$, we have $N_{\mathcal{F}}(i) = \mathcal{J}$ for all $i \in \mathcal{I}$, $N_{\mathcal{F}}(j) = \mathcal{I}$ for all $j \in \mathcal{J}$.

For an arbitrary subgraph $\mathcal{H} \subseteq \mathcal{F}$, $I(\mathcal{H})$ denotes the set of queues that are adjacent to any arc in this subgraph. In other words, $I(\mathcal{H})$ is the queue set of the subgraph $\mathcal{H}$. Similarly, $J(\mathcal{H})$ denotes the set of servers that are adjacent to any arc in this subgraph, i.e., the server set of subgraph $\mathcal{H}$.

A given flexibility structure $\mathcal{S}$ is the arc set of the bipartite graph whose node partition is $\mathcal{I}$ and $\mathcal{J}$. This bipartite graph may contain paths. Let $P(d_1, d_2)$ be a path from node d_1 to node d_2 in $\mathcal{S}$, if it exists. Generally, d_1 and d_2 may be servers or queues. If d_1 is a server

and d_2 is a queue,

$$P(d_1, d_2) = \{(d_1, i_1), (i_1, j_2), (j_2, i_2), \ldots, (j_k, d_2)\}$$

denotes a path between d_1 and d_2. In this path, arcs of the form (j_r, i_r) are forward arcs, and arcs of the form (i_r, j_{r+1}) are backward arcs. Forward arcs in the path are denoted by $\overline{P}(d_1, d_2)$, and backward arcs are denoted by $\underline{P}(d_1, d_2)$.

This bipartite graph may also contain cycles consisting of equal number of forward and backward arcs. Without loss of generality, we will adopt the convention that cycles start with a server node, j_1. Let

$$W = \{(j_1, i_1), (i_1, j_2), (j_2, i_2), \ldots, (j_k, i_k), (i_k, j_1)\}$$

be an arbitrary cycle in $\mathcal{S}$, if it exists. Arcs of the form (j_r, i_r) for $r \in \{1, \ldots, k\}$ are forward arcs in this cycle. Let the set of forward arcs of W be denoted by $\overline{W}$. Similarly, arcs of the form (i_r, j_{r+1}) for $r \in \{1, \ldots, k\}$ are backward arcs in W. Let the set of backward arcs of W be denoted by $\underline{W}$.

Definition 3. *(Path/Cycle Multiplier) For a path $P(d_1, d_2)$, the path multiplier $\gamma(P(d_1, d_2))$ is defined as*

$$\gamma(P(d_1, d_2)) = \frac{\prod_{(i',j') \in \overline{P}(d_1,d_2)} \mu_{i'j'}}{\prod_{(i',j') \in \underline{P}(d_1,d_2)} \mu_{i'j'}}.$$

For a cycle W, the cycle multiplier $\gamma(W)$ is defined similarly as

$$\gamma(W) = \frac{\prod_{(i',j') \in \overline{W}} \mu_{i'j'}}{\prod_{(i',j') \in \underline{W}} \mu_{i'j'}}.$$

We say that if $\gamma(W) = 1$, W is a *breakeven cycle*. An in-depth treatment of cycles with multipliers is given in Chapter 15 of Ahuja et al. (1988). Intuitively, when a flow of δ is

pushed from j_1 of the cycle W, this flow traverses the cycle and returns to j_1 as $\delta\gamma(W)$ units of flow. Then, a breakeven cycle conserves the amount of flow pushed from the initial node, since if δ is pushed from j_1, δ returns to j_1. In general, there is no reason to believe that an arbitrary cycle W is breakeven. However, as we discuss in Section 3.3, the special structure of the static planning problem will imply that if we use any cycle in the design of $\mathcal{S}$, they must be breakeven cycles.

3.3 Static Planning Problem and Its Dual

Traditionally, the analysis of control policies in parallel server systems has primarily been conducted under heavy traffic assumptions. This is due to the inherent complexity of the pre-limit stochastic control problem, which is usually many-dimensional. To be able to detach the design problem from performance-related concerns, we make the assumption that the flexible server system operates in heavy traffic. By doing so, we can concentrate on constructing a sparse flexibility structure; knowing that if we design the flexibility structure carefully, there are control policies studied in heavy traffic that are expected to yield comparable performance to a full flexibility system.

Heavy traffic conditions can be characterized by the static planning problem (SPP). This is a deterministic linear program that establishes an optimal resource allocation if the system was static, i.e., running under fluid scale. The SPP yields an optimal processing plan vector, which consists of long run average fractions of time that servers should dedicate to each *activity*. The long run average fraction of time server j dedicates to activity (i, j) is denoted by the decision variable $x_{ij} \geq 0$, for each $(i, j) \in \mathcal{S}$. The SPP minimizes ρ, which denotes the long run utilization level of the busiest server in the system, by choosing an optimal processing plan vector $\mathbf{x}$.

Before formally stating the SPP, we would like to point out that the problem statement is heavily dependent on the flexibility structure $\mathcal{S}$, since $\mathcal{S}$ determines admissible activities.

Below, we state the SPP and its dual for the full flexibility structure $\mathcal{F}$.

In SPP($\mathcal{F}$), $\mathbf{x}$ is a vector of size mn. Without loss of generality, we order x_{ij} first by j, then i. Adopting the notation in Harrison and López (1999), input-output matrix R and the resource consumption matrix A are given by

$$R = \begin{bmatrix} \mu_{11} & 0 & \cdots & 0 & \mu_{12} & 0 & \cdots & 0 & \cdots & \mu_{1m} & \cdots & 0 \\ 0 & \mu_{21} & \cdots & 0 & 0 & \mu_{22} & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mu_{n1} & 0 & 0 & \cdots & \mu_{n2} & \cdots & 0 & \cdots & \mu_{nm} \end{bmatrix},$$

and

$$A = \begin{bmatrix} \mathbf{1}_{1\times n} & \mathbf{0}_{1\times n} & \cdots & \mathbf{0}_{1\times n} \\ \mathbf{0}_{1\times n} & \mathbf{1}_{1\times n} & \cdots & \mathbf{0}_{1\times n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{1\times n} & \mathbf{0}_{1\times n} & \cdots & \mathbf{1}_{1\times n} \end{bmatrix}.$$

R is of size $n \times mn$, and A is of size $m \times mn$. For activity (i, j), R_{ij} represents the departure rate of the jobs from queue i due to service from server j, and A_{ij} represents the consumption rate of the capacity of server j by activity (i, j). Then the SPP($\mathcal{F}$) can be written formally as

$$\begin{aligned} \text{SPP}(\mathcal{F}) \qquad \min_{\mathbf{x},\rho} \quad & \rho \\ s.t. \quad & R\mathbf{x} = \boldsymbol{\lambda} \\ & A\mathbf{x} \leq \rho\mathbf{1} \\ & \mathbf{x} \geq \mathbf{0} \\ & \rho \geq 0, \end{aligned}$$

and its dual, DSPP($\mathcal{F}$) can be stated as

$$\begin{aligned} \text{DSPP}(\mathcal{F}) \qquad \max_{\mathbf{y},\mathbf{z}} \quad & \mathbf{y}^T\boldsymbol{\lambda} \\ s.t. \quad & \mathbf{y}^T R - \mathbf{z}^T A \leq \mathbf{0} \\ & \mathbf{z}^T\mathbf{1} \leq 1 \\ & \mathbf{z} \geq \mathbf{0}. \end{aligned}$$

For an arbitrary flexibility structure $\mathcal{S}$, SPP($\mathcal{S}$) and DSPP($\mathcal{S}$) are defined similarly. The difference is that the columns of the coefficient matrices, $R_{\mathcal{S}}$ and $A_{\mathcal{S}}$, are subsets of columns of R and A that include only the activities in $\mathcal{S}$, respectively.

We assume that heavy traffic and CRP conditions hold for $\mathcal{F}$. That is, we assume:

Assumption. ***A1.*** *SPP($\mathcal{F}$) has at least one solution.*

A2. *All solutions $(\mathbf{x}^*, \rho^*)$ to SPP($\mathcal{F}$) satisfy $\rho^* = 1$ and $A\mathbf{x}^* = \rho^* = 1$.*

A3. *DSPP($\mathcal{F}$) has a unique solution.*

Assumptions **A1.** and **A2.** are heavy traffic conditions. **A1.** only requires that SPP($\mathcal{F}$) is feasible. **A2.** states that all servers are critically loaded. This can be interpreted as we are limiting our attention to a bottleneck subsystem.

Assumption **A3.** is the CRP assumption. This is equivalent to the existence of a connected solution to SPP($\mathcal{F}$) under heavy traffic. Assumption **A3.** is the same as Condition 1 in Shi et al. (2019), or Assumption 2.4 in Gurvich and Whitt (2009).

The coefficient matrix of SPP($\mathcal{F}$),

$$\begin{bmatrix} R & \mathbf{0} \\ A & -\mathbf{1} \end{bmatrix}, \tag{3.1}$$

has $m+n$ linearly independent rows and $mn+1$ columns. Then any optimal basis will have $m+n$ rows and $m+n$ columns. Moreover, any optimal basic feasible solution (BFS) to SPP($\mathcal{F}$) will have at most $m+n$ positive variables among $(\mathbf{x}, \rho)$. In any non-trivial system, $\rho > 0$. Then, at most $m+n-1$ elements of $\mathbf{x}$ will be positive in an optimal BFS.

If exactly $m+n-1$ elements among $\mathbf{x}$ are positive in an optimal BFS, we call that a non-degenerate optimal BFS and the associated basis a non-degenerate basis.

If less than $m+n-1$ elements of $\mathbf{x}$ are positive in an optimal BFS, we call that a degenerate optimal BFS. A degenerate optimal BFS can be represented by many different bases by changing the basis elements that correspond to zero entries in the solution. If we have a degenerate optimal BFS, we call all its associated bases degenerate bases.

If a linear program has a unique optimal solution, it is necessarily a BFS. However, if a linear program has multiple solutions, it has optimal BFSs and optimal non-basic solutions. We extend the definition of non-degeneracy to non-basic solutions as well: if any optimal solution of SPP($\mathcal{F}$) has at least $m+n-1$ elements of $\mathbf{x}$ positive, we call that a non-degenerate optimal solution. Note that degenerate solutions are always basic: they can be appended by arbitrary (i,j) with $x_{ij}=0$ until the basis size (including ρ) reaches $m+n$.

We assume that DSPP($\mathcal{F}$) has a unique solution, but would like to make the distinction between DSPP($\mathcal{F}$), which is defined for a fully flexible system, and the dual of the SPP for an arbitrary sparse graph $\mathcal{S}$. Intuitively, we want to prune a fully flexible graph, whose SPP($\mathcal{F}$) satisfies Assumptions **A1.-A3.**, and obtain a sparse graph $\mathcal{S}$ whose SPP($\mathcal{S}$) still satisfies Assumptions **A1.-A3.**. Then **A3.** for $\mathcal{S}$ would imply that the sparse $\mathcal{S}$ can achieve a single resource pool in heavy traffic, i.e., CRP holds for $\mathcal{S}$. Under CRP, it is expected that there exist asymptotically optimal scheduling policies in heavy traffic for $\mathcal{S}$ that attain performance close to that of $\mathcal{F}$, i.e., the sparse design $\mathcal{S}$ has good performance.

Let SPP($\mathcal{F}$) have κ optimal bases. Let B_k, $k \in \{1, \ldots, \kappa\}$ denote the optimal bases to SPP($\mathcal{F}$). Each B_k consists of basic variables $(\mathbf{x^{k}}^*, \rho^*)$. However, we will sometimes abuse

notation and use B_k to denote the set of arcs corresponding to basic variables of B_k, rather than the basic variables themselves. Then each $B_k \subset \mathcal{F}$ is a flexibility design itself, for which we can write SPP(B_k). If B_k is a non-degenerate optimal basis, all $x_{ij}^{k*} > 0$ for $(i,j) \in B_k$. The next lemma, which is an application of Proposition 3 in Harrison and López (1999), suggests that in that case, B_k itself is a sparse design with $m+n-1$ arcs that achieves a single resource pool in heavy trafffic.

Lemma 3. *Let B be a non-degenerate optimal basis to SPP($\mathcal{F}$). Then SPP(B) has a unique non-degenerate solution, and DSPP(B) has a unique solution.*

Proof. Let B be a basis to SPP($\mathcal{F}$) under Assumptions **A1.**-**A3.**. Then the following system has a unique solution with $\rho^* = 1$:

$$\begin{bmatrix} R_B & 0 \\ A_B & -1 \end{bmatrix} \begin{bmatrix} \mathbf{x_B} \\ \rho \end{bmatrix} = \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{0} \end{bmatrix},$$

where R_B and A_B are matrices that denote the columns of R and A that correspond to the basis B. The unique solution of this system is feasible for SPP(B). It suffices to show that in SPP(B), ρ^* is never strictly less than one. Indeed, if there exists a solution with $\tilde{\rho} < 1$ to SPP(B), it is feasible to SPP($\mathcal{F}$). But then Assumption **A2.** is violated, so $\tilde{\rho} < 1$ is not possible.

Then SPP(B) has a unique non-degenerate solution. By Proposition 3 in Harrison and López (1999), DSPP(B) has a unique solution, and B is a connected graph. □

This result also suggests that if there exists a non-degenerate optimal basis to SPP($\mathcal{F}$), that would be a sparse design with $m+n-1$ arcs and good performance. However, existence of a non-degenerate optimal basis is not a computationally easy property to establish if SPP($\mathcal{F}$) has multiple solutions. Existence of a non-degenerate solution (not necessarily basic) in an LP was established in Corollary 1 of Tijssen and Sierksma (1998). With some modifications

for our problem, the corollary states

> A dual LP-model has a unique and degenerate optimal solution, if and only if the corresponding primal LP-model has multiple optimal solutions of which at least one is non-degenerate.

This result would imply that if we could show that the unique solution to DSPP($\mathcal{F}$) is degenerate, then SPP($\mathcal{F}$) has a non-degenerate solution. However, since this non-degenerate solution is not necessarily basic, it may not be sparse at all. Unfortunately, this characterization does not extend to basic solutions. That is, the characterization is incomplete when only basic solutions are considered; if the dual LP has a unique and degenerate optimal BFS, then it can only be concluded that the primal LP has multiple solutions. (see Theorem 5.6.1 in Sierksma and Zwols (2015)).

To establish the existence of a non-degenerate basis, one could enumerate all optimal BFS and check whether there exists a non-degenerate one, but the number of extreme points of the feasible set grows exponentially as m and n grow. Even starting at an optimal BFS, if it is degenerate, it has $O(mn)$ neighboring basic solutions to check, none of them guaranteed to be non-degenerate. Therefore, in practice, we need a computationally better method to build a sparse graph with good performance.

To that end, first, we extend the result in Lemma 3 to non-basic non-degenerate solutions.

Lemma 4. *Let $\mathcal{S} \subseteq \mathcal{F}$ be a connected graph. If there exists non-degenerate optimal solution to SPP($\mathcal{S}$) that uses the entire graph $\mathcal{S}$ then DSPP($\mathcal{S}$) has a unique optimal solution $(\boldsymbol{y}^*, \boldsymbol{z}^*)$ where $\boldsymbol{y}^* > \boldsymbol{0}$ and $\boldsymbol{z}^* > \boldsymbol{0}$, componentwise.*

Proof. If there exists a non-degenerate optimal solution $\mathbf{x}$ to SPP($\mathcal{S}$) that uses the entire graph, we have $x_{ij} > 0$ for all $(i, j) \in \mathcal{S}$. Then, complementary slackness implies that any

feasible solution of the dual is optimal if

$$\begin{aligned} &\mathbf{y}^T R_{\mathcal{S}} - \mathbf{z}^T A_{\mathcal{S}} = \mathbf{0}, \\ &\mathbf{z}^T e = 1. \end{aligned} \tag{3.2}$$

Rewritten in matrix form, the system is

$$\begin{bmatrix} \mathbf{y}^T & \mathbf{z}^T \end{bmatrix} \begin{bmatrix} R_{\mathcal{S}} & \mathbf{0}_{n\times 1} \\ -A_{\mathcal{S}} & \mathbf{-1}_{m\times 1} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{|\mathcal{S}|\times 1} \\ 1 \end{bmatrix}.$$

Consider the coefficient matrix of this system. Because $\mathcal{S}$ is connected, it has at least $m+n-1$ arcs. Moreover, the columns of this matrix are a subset of the rows of the coefficient matrix 3.1, which are linearly independent. Then this coefficient matrix has rank $m+n$, which is equal to the number of unknowns in system 3.2. Then, system 3.2 has a unique solution, implying that DSPP($\mathcal{S}$) has a unique optimal solution. By Lemma 2.3 in Atar et al. (2022), $\mathbf{y}^* > \mathbf{0}$ and $\mathbf{z}^* > \mathbf{0}$, componentwise. □

This extension enables us to conclude that DSPP($\mathcal{S}$) has a unique solution if we can find a solution to SPP($\mathcal{S}$) that uses all the arcs in a connected sparse $\mathcal{S}$. To be able to construct such a solution to SPP($\mathcal{S}$), we need to choose $\mathcal{S}$ carefully. Because $\mathcal{S} \subseteq \mathcal{F}$, any solution to SPP($\mathcal{S}$) that uses all the arcs in $\mathcal{S}$ will be a solution to SPP($\mathcal{F}$). Then to choose $\mathcal{S}$, we can safely remove the arcs in $\mathcal{F}$ that are never used in any solution.

As an intermediary tool, we define the set of arcs that are used in some BFS of SPP($\mathcal{F}$) as

$$\mathcal{G} = \left\{ (i,j) \in \mathcal{F} : \exists\, B_k, k \in \{1,\dots,\kappa\} \text{ s.t. } x_{ij}^{k^*} > 0 \right\}.$$

In other words, $\mathcal{G}$ is the support graph of all BFS to SPP($\mathcal{F}$). We would like to note that $\mathcal{G}$ is not necessarily sparse. In the worst case, $\mathcal{G} = \mathcal{F}$ since all arcs may be used in some

solution to SPP($\mathcal{F}$).

In practice, $\mathcal{G}$ may be obtained by solving a series of the linear programs. Algorithm 1 suggests a procedure for obtaining $\mathcal{G}$.

Algorithm 1: Construction of $\mathcal{G}$

Input : $\mathcal{F}$

Set $\mathcal{R} = \mathcal{F}$, $\mathcal{G}' = \emptyset$.

while $\mathcal{R} \neq \emptyset$ **do**

  Fix $(i', j') \in \mathcal{R}$.

  Solve the linear program

$$\max_{\mathbf{x}} \{x_{i'j'} : R\mathbf{x} = \lambda,\ A\mathbf{x} = \mathbf{1},\ \mathbf{x} \geq \mathbf{0}\}$$

  and obtain an optimal solution $\mathbf{x}^*$.

  Update $\mathcal{G}' = \mathcal{G}' \cup \{(i, j) \in \mathcal{F} : x^*_{ij} > 0\}$.

  Update $\mathcal{R} = \mathcal{R} \setminus (\mathcal{G}' \cup \{(i', j')\})$.

end while

Set $\mathcal{G} = \mathcal{G}'$.

Return: $\mathcal{G}$

It can be seen that because (i', j') is removed from $\mathcal{R}$ in each iteration, the algorithm terminates. If

$$\max_{\mathbf{x}} \{x_{i'j'} : R\mathbf{x} = \lambda,\ A\mathbf{x} = \mathbf{1},\ \mathbf{x} \geq \mathbf{0}\} = 0$$

for some (i', j'), then $(i', j') \notin \mathcal{G}$ and $x_{i'j'} = 0$ in all iterations. Such (i', j') are never included in $\mathcal{G}'$.

On the other hand, if

$$\max_{\mathbf{x}}\{x_{i'j'} : R\mathbf{x} = \lambda,\ A\mathbf{x} = \mathbf{1},\ \mathbf{x} \geq \mathbf{0}\} > 0$$

for some (i', j'), then $(i', j') \in \mathcal{G}$ and $(i', j') \in \mathcal{G}'$ for that iteration.

Therefore, Algorithm 1 constructs $\mathcal{G}$ by solving fewer than mn simple linear programs. Under Assumption **A3.**, we have the following useful properties for $\mathcal{G}$.

Lemma 5. *(Properties of $\mathcal{G}$)*

1. *There exists a solution $\boldsymbol{x}$ to SPP($\mathcal{F}$) with $x_{ij} > 0$ for all $(i, j) \in \mathcal{G}$.*

2. *$\mathcal{G}$ is connected.*

3. *All cycles in $\mathcal{G}$ are breakeven.*

Proof. 1. By definition, for any $(i, j) \in \mathcal{G}$, there exists a solution $\mathbf{x}^{(i,j)}$ to SPP($\mathcal{F}$) where $x_{ij}^{(i,j)} > 0$. Convex combinations of solutions to a linear program are solutions to the linear program. Then, taking the convex combination of solutions $\mathbf{x}^{(i,j)}$,

$$\mathbf{x}^* = \sum_{(i,j)\in\mathcal{G}} \alpha^{(i,j)}\mathbf{x}^{(i,j)}, \quad \sum_{(i,j)\in\mathcal{G}} \alpha^{(i,j)} = 1, \quad \alpha^{(i,j)} > 0,$$

we obtain a solution $\mathbf{x}^*$ to SPP where all arcs in $\mathcal{G}$ are used.

2. Assume to the contrary that $\mathcal{G}$ is not connected. Then $\mathcal{G}$ has at least two components. We will show that DSPP($\mathcal{F}$) has multiple solutions when $\mathcal{G}$ has two components, in order to avoid notational burden. The same idea works when $\mathcal{G}$ has arbitrary number of components by picking an arbitrary couple of the components.

Let C_1 and C_2 be the two components of $\mathcal{G}$. By complementary slackness,

$$\begin{aligned}
&\mu_{ij} y_i = z_j, \quad i \in I(C_1), j \in J(C_1) \\
&\mu_{i'j'} y_{i'} = z_{j'}, \quad i' \in I(C_2), j' \in J(C_2) \\
&\sum_{j \in J(C_1)} z_j + \sum_{j' \in J(C_2)} z_{j'} = 1, \\
&\sum_{i \in I(C_1)} \lambda_i y_i + \sum_{i' \in I(C_2)} \lambda_{i'} y_{i'} = 1.
\end{aligned}$$

We will perturb $\mathbf{y}$ and $\mathbf{z}$ and obtain another optimal solution to DSPP($\mathcal{F}$). Fix $i^* \in I(C_1)$, and fix $i^{*\prime} \in I(C_2)$. Let

$$\bar{y}_{i^*} = y_{i^*} + \epsilon,$$

where $\epsilon > 0$. Then to preserve feasibility, we perturb the rest of y_i, z_j in component C_1 such that

$$\begin{aligned}
\bar{y}_i &= y_i + \frac{y_i}{y_{i^*}} \epsilon \quad \forall i \in I(C_1), \\
\bar{z}_j &= z_j + \frac{z_j}{y_{i^*}} \epsilon \quad \forall j \in J(C_1).
\end{aligned}$$

This new solution satisfies $\mu_{ij} \bar{y}_i = \bar{z}_j$ for $i \in I(C_1)$, $j \in J(C_1)$ by construction. Similarly, let

$$\begin{aligned}
\bar{y}_{i'} &= y_{i'} - \frac{y_{i'}}{y_{i^{*\prime}}} \delta \quad \forall i' \in I(C_2), \\
\bar{z}_{j'} &= z_{j'} - \frac{z_{j'}}{y_{i^{*\prime}}} \delta \quad \forall j' \in J(C_2).
\end{aligned}$$

Now we need to show that $\epsilon, \delta > 0$ exist such that

$$\sum_{j \in J(C_1)} \bar{z}_j + \sum_{j' \in J(C_2)} \bar{z}_{j'} = 1,$$
$$\sum_{i \in I(C_1)} \lambda_i \bar{y}_i + \sum_{i' \in I(C_2)} \lambda_{i'} \bar{y}_{i'} = 1.$$

These equalities are satisfied if

$$\sum_{j \in J(C_1)} \frac{z_j}{y_{i^*}} \epsilon - \sum_{j' \in J(C_2)} \frac{z_{j'}}{y_{i^{*\prime}}} \delta = 0,$$
$$\sum_{i \in I(C_1)} \lambda_i \frac{y_i}{y_i^*} \epsilon - \sum_{i' \in I(C_2)} \lambda_{i'} \frac{y_{i'}}{y_{i^{*\prime}}} \delta = 0.$$

But this system is consistent for some $\epsilon, \delta > 0$ since

$$\begin{aligned}
\sum_{i \in I(C_1)} \lambda_i \frac{y_i}{y_i^*} &= \sum_{i \in I(C_1)} \left(\sum_{j \in J(C_1)} \mu_{ij} x_{ij} \right) \frac{y_i}{y_{i^*}} \\
&= \frac{1}{y_{i^*}} \sum_{i \in I(C_1)} \sum_{j \in J(C_1)} \mu_{ij} x_{ij} y_i \\
&= \frac{1}{y_{i^*}} \sum_{i \in I(C_1)} \sum_{j \in J(C_1)} x_{ij} z_j \\
&= \frac{1}{y_{i^*}} \sum_{j \in J(C_1)} z_j \sum_{i \in I(C_1)} x_{ij} \\
&= \sum_{j \in J(C_1)} \frac{z_j}{y_i^*},
\end{aligned}$$

where $\mathbf{x}$ is an optimal solution to the primal problem. Similarly,

$$\sum_{i' \in I(C_2)} \lambda_{i'} \frac{y_{i'}}{y_{i^{*\prime}}} = \sum_{j' \in J(C_2)} \frac{z_{j'}}{y_{i^{*\prime}}}.$$

Then we choose $\epsilon, \delta > 0$ such that

$$\frac{\epsilon}{\delta} = \frac{\sum_{j \in J(C_1)} \frac{z_j}{y_i^*}}{\sum_{j' \in J(C_2)} \frac{z_{j'}}{y_{i^{*\prime}}}},$$

$$\delta < y_{i^{*\prime}}.$$

Then the solution $\bar{\mathbf{y}}, \bar{\mathbf{z}}$ is optimal for DSPP($\mathcal{F}$), contradicting the uniqueness of the solution. Then $\mathcal{G}$ must be connected.

3. Let $W \subseteq \mathcal{G}$ be a cycle of $\mathcal{G}$, where

$$W = \{(j_1, i_1), (i_1, j_2), (j_2, i_2), \ldots, (j_k, i_k), (i_k, j_1)\}.$$

Because DSPP($\mathcal{F}$) has a unique solution and there exists a solution to SPP($\mathcal{F}$) that uses the entire $\mathcal{G}$, complementary slackness implies that for any $(i, j) \in \mathcal{G}$, we have $y_i = z_j / \mu_{ij}$. Consider the cycle W.

$$\begin{aligned}
y_{i_1} &= \frac{z_{j_1}}{\mu_{i_1 j_1}} = \frac{z_{j_2}}{\mu_{i_1 j_2}} \\
y_{i_2} &= \frac{z_{j_2}}{\mu_{i_2 j_2}} = \frac{z_{j_3}}{\mu_{i_2 j_3}} \\
&\vdots \\
y_{i_k} &= \frac{z_{j_k}}{\mu_{i_k j_k}} = \frac{z_{j_1}}{\mu_{i_k j_1}}.
\end{aligned}$$

Then,

$$\begin{aligned} z_{j_1} &= z_{j_2}\frac{\mu_{i_1 j_1}}{\mu_{i_1 j_2}} \\ &= z_{j_3}\frac{\mu_{i_2 j_2}\mu_{i_1 j_1}}{\mu_{i_2 j_3}\mu_{i_1 j_2}} \\ &= \cdots = z_{j_k}\frac{\mu_{i_{k-1} j_{k-1}}\cdots\mu_{i_2 j_2}\mu_{i_1 j_1}}{\mu_{i_{k-1} j_k}\cdots\mu_{i_2 j_3}\mu_{i_1 j_2}} \\ &= z_{j_1}\frac{\mu_{i_k j_k}\mu_{i_{k-1} j_{k-1}}\cdots\mu_{i_2 j_2}\mu_{i_1 j_1}}{\mu_{i_k j_1}\mu_{i_{k-1} j_k}\cdots\mu_{i_2 j_3}\mu_{i_1 j_2}}, \end{aligned}$$

implying that the cycle multiplier $\gamma(W)$ satisfies

$$\gamma(W) = \frac{\mu_{i_k j_k}\cdots\mu_{i_2 j_2}\mu_{i_1 j_1}}{\mu_{i_k j_1}\cdots\mu_{i_2 j_3}\mu_{i_1 j_2}} = 1.$$

Then W is a breakeven cycle.

□

Next, we show that any basis B is a forest. In other words, any BFS to SPP($\mathcal{F}$) will either be a tree, or contain several disconnected trees.

Lemma 6. *Support graph of any basis B to SPP($\mathcal{F}$) is cycle free. That is, any basis B is a forest.*

Proof. Assume to the contrary that support of B contains a cycle

$$W = \{(j_1, i_1), (i_1, j_2), (j_2, i_2), \ldots, (j_k, i_k), (i_k, j_1)\}.$$

Let $\mathbf{x}$ denote the unique solution determined by B. By feasibility,

$$\sum_{j:(i,j)\in W} \mu_{ij}x_{ij} + \sum_{j:(i,j)\in W^c} \mu_{ij}x_{ij} = \lambda_i, \quad \forall i \in \{1,\dots,n\}$$
$$\sum_{i:(i,j)\in W} x_{ij} + \sum_{i:(i,j)\in W^c} x_{ij} = 1, \quad \forall j \in \{1,\dots,m\}.$$

Note that $W \subseteq \mathcal{G}$. By Lemma 5, W is a breakeven cycle. Then we can perturb $\mathbf{x}$ on W and remain feasible. Let $P(j_1,k) \subseteq W$ denote the path from node j_1 to any other node k in the cycle W. Let $\overline{P}(j_1,k)$ denote the forward arcs on this path, and let $\underline{P}(j_1,k)$ denote the backward arcs. For all arcs $(i,j) \in W$, the perturbed solution is

$$\bar{x}_{ij} = \begin{cases} x_{ij} + \gamma(P(j_1,j))\epsilon, \text{ if } (i,j) \in \overline{P}(j_1,j) \\ x_{ij} - \gamma(P(j_1,j))\epsilon, \text{ if } (i,j) \in \underline{P}(j_1,j) \end{cases}$$

for some $\epsilon > 0$. We now show that the perturbed solution is feasible. All nodes in cycle W appear in exactly one forward and one backward arc through the cycle. Then for each i_r and $j_r \neq j_1$ in the cycle,

$$\bar{x}_{i_r j_{r-1}} + \bar{x}_{i_r j_r} = x_{i_r j_{r-1}} + x_{i_r j_r}$$
$$\mu_{i_r j_r}\bar{x}_{i_r j_r} + \mu_{i_{r+1} j_r}\bar{x}_{i_{r+1} j_r} = \mu_{i_r j_r}x_{i_r j_r} + \mu_{i_{r+1} j_r}x_{i_{r+1} j_r},$$

maintaining feasibility. For j_1,

$$\bar{x}_{i_1 j_1} + \bar{x}_{i_k j_1} = x_{i_1 j_1} + \epsilon + x_{i_k j_1} - \gamma(W)\epsilon = x_{i_1 j_1} + x_{i_k j_1},$$

since $\gamma(W) = 1$. Then $\mathbf{x}$ and $\bar{\mathbf{x}}$ are both solutions determined by B, contradicting the uniqueness of $\mathbf{x}$. Then B cannot contain a cycle. □

Lemma 5 and Lemma 6 carry essential insights for our main result. Firstly, there exists a potentially non-basic solution to SPP($\mathcal{F}$) that uses all arcs in $\mathcal{G}$. Then, x_{ij} from any other feasible solution that uses a subgraph of $\mathcal{G}$ can be propagated to the entirety of $\mathcal{G}$. Carefully doing so might enable us to move from a sparse graph with undesirable properties (such as multiple solutions to the dual SPP) to a still-sparse graph that uses additional arcs from $\mathcal{G}$ that satisfies Assumptions **A1.- A3.**. Secondly, connectivity of $\mathcal{G}$ is essential to use Lemma 4. If $\mathcal{G}$ was not connected, there would be no chance of constructing a connected sparse graph and utilize Lemma 4 to conclude that the sparse graph has a unique solution for the associated DSPP. Thirdly, all cycles being breakeven implies that all cycles conserve flow. In other words, during the propagation of an existing solution $\mathbf{x}$ to another solution on $\mathcal{G}$, cycles may be used to preserve feasibility, but they cannot discharge any supply or demand imbalance that occur at nodes.

Finally, Lemma 6 implies that a degenerate basis B with τ components contains exactly $m+n-\tau$ arcs. Lemmas 5 and 6 equip us for the main result, presented in the next section.

3.4 Sparse Design of the Flexibility Graph

Up to this point, we have seen that we can solve SPP($\mathcal{F}$) and obtain a BFS $\mathbf{x}^*$. If $\mathbf{x}^*$ is a non-degenerate BFS, associated optimal basis B is a sparse design with a unique solution to DSPP(B) (Lemma 3). If $\mathbf{x}^*$ is a degenerate optimal BFS, DSPP(B) may have multiple solutions. But if there exists a connected non-degenerate (potentially non-basic) solution to SPP($\mathcal{F}$), then for associated $\mathcal{S}$, DSPP($\mathcal{S}$) has a unique solution (Lemma 4).

If $\mathbf{x}^*$ is a degenerate optimal BFS, it has fewer than $m+n-1$ entries so that it is disconnected. We can use $\mathcal{G}$ and augment the basis of $\mathbf{x}^*$ to obtain a sparse graph $\mathcal{S}$ on which there exists a non-degenerate non-basic solution to SPP($\mathcal{F}$). In the next proposition we show that any component of a disconnected optimal solution of SPP($\mathcal{F}$) has at least one incoming and one outgoing arc in $\mathcal{G}$. In this context, a disconnected component C is a strict

subgraph of $\mathcal{F}$, and $I(C) \subset \mathcal{I}$, $J(C) \subset \mathcal{J}$. After establishing that C is actually connected to the rest of the graph in $\mathcal{G}$, with at least one incoming and one outgoing arc, we propose to append any initial disconnected solution using those incoming and outgoing arcs for each component in $\mathcal{G}$. Our main result is an application of the proposition, which states that this procedure results in a sparse $\mathcal{S}$ with a unique solution to DSPP($\mathcal{S}$).

Proposition 4. *Let C be a component of a disconnected optimal solution to SPP($\mathcal{F}$). There exists $(i^*, j^\dagger) \in \mathcal{G}$ where $i^* \in I(C)$ and $j^\dagger \in J(C^c)$, and $(i^\dagger, j^*) \in \mathcal{G}$ where $j^* \in J(C)$ and $i^\dagger \in I(C^c)$.*

Proof. Because C is a component in a disconnected optimal solution $\mathbf{x}^*$ to SPP($\mathcal{F}$), we have

$$R_C \mathbf{x}_C^* = \boldsymbol{\lambda}_C$$
$$A_C \mathbf{x}_C^* = \mathbf{1}$$

where elements of $\mathbf{x}_C^*$ correspond to $(i, j) \in C$, rows of R_C and elements of $\boldsymbol{\lambda}_C$ correspond to queues in $I(C)$ and rows of A_C correspond to servers in $J(C)$. Consider the SPP and the DSPP restricted to this component.

$$\begin{aligned} SPP(C) \qquad \min_{\mathbf{x}_C, \rho_C} \quad & \rho_C \\ s.t. \quad & R_C \mathbf{x}_C = \boldsymbol{\lambda}_C \\ & A_C \mathbf{x}_C \leq \rho_C \mathbf{1} \\ & \mathbf{x}_C \geq \mathbf{0} \\ & \rho_C \geq 0. \end{aligned}$$

$$
\begin{aligned}
DSPP(C) \qquad \max_{\mathbf{y}_C, \mathbf{z}_C} \quad & \mathbf{y}_C^T \boldsymbol{\lambda}_C \\
s.t. \quad & \mathbf{y}_C^T R_C - \mathbf{z}_C^T A_C \leq \mathbf{0} \\
& \mathbf{z}_C^T \mathbf{1} \leq 1 \\
& \mathbf{z}_C \geq \mathbf{0}.
\end{aligned}
$$

We claim that in this restricted system,

1. The optimal objective value of SPP(C) is $\rho_C^* = 1$ in all solutions $(\rho_C^*, \mathbf{x}_C^*)$. Indeed, assume to the contrary that $\rho_C^* < 1$. Then we would have $A_C \mathbf{x}_C^* < \mathbf{1}$ in all solutions to SPP(C). Because C is a component implied by a disconnected optimal solution of SPP($\mathcal{F}$), any solution that satisfies $A_C \mathbf{x}_C^* < \mathbf{1}$ is part of a feasible solution to SPP($\mathcal{F}$). Then $(\mathbf{x}_C^*, \mathbf{x}_{C^c}^*)$ is a feasible solution to SPP($\mathcal{F}$), where $\mathbf{x}_{C^c}^*$ corresponds to elements of $\mathbf{x}^*$ not associated with C, contradicting assumption **A1.**.

2. In all optimal solutions, $A_C \mathbf{x}_C^* = 1$. Otherwise, because $\mathbf{x}_C^*$ is part of a feasible solution to SPP($\mathcal{F}$), we contradict heavy traffic assumptions for SPP($\mathcal{F}$).

3. C is connected, so DSPP(C) has a unique solution $(\mathbf{y}_C^*, \mathbf{z}_C^*)$ where $\mathbf{y}_C^* > \mathbf{0}$ and $\mathbf{z}_C^* > \mathbf{0}$ componentwise. This is true by Lemma 4.

By Lemma 5, there exists an optimal solution $\tilde{\mathbf{x}}$ on $\mathcal{G}$ where $\tilde{x}_{ij} > 0$ for all $(i,j) \in \mathcal{G}$. Also by Lemma 5, $\mathcal{G}$ is connected. Hence, either $(i^*, j^\dagger)$ where $i^* \in I(C)$, $j^\dagger \in J(C^c C)$ or $(i^\dagger, j^*)$ where $i^\dagger \in I(C^c)$, $j^* \in J(C)$ exists in $\mathcal{G}$.

Case 1: Let $(i^\dagger, j^*) \in \mathcal{G}$. We will show that $(i^*, j^\dagger) \in \mathcal{G}$ exists. Because $(i^\dagger, j^*) \in \mathcal{G}$, we have $\tilde{x}_{i^\dagger, j^*} > 0$. Then the vector

$$
\mathbf{w} = \left[\sum_{i \in I(C^c)} \tilde{x}_{ij} \right]_{j \in J(C)}
$$

has all non-negative, and at least one strictly positive component. Specifically, $w_{j^*} > 0$. Note that by construction, servers in $J(C)$ dedicate their remaining capacity **1-w** to queues in $I(C)$.

We want to show that $(i^*, j^\dagger) \in \mathcal{G}$ exists. Assume to the contrary that such incoming arc does not exist. Then the demand of all $i \in I(C)$ are satisfied by $j \in J(C)$ and $\tilde{\mathbf{x}}$ satisfies

$$\begin{aligned} R_C \tilde{\mathbf{x}}_C &= \boldsymbol{\lambda}_C \\ A_C \tilde{\mathbf{x}}_C &= \mathbf{1} - \mathbf{w}, \end{aligned}$$

where $\tilde{\mathbf{x}}_C$ consists of elements of $\tilde{\mathbf{x}}$ corresponding to arcs in C. By strong duality,

$$\mathbf{y}_C^{*T} R_C \tilde{\mathbf{x}}_C = \mathbf{y}_C^{*T} \boldsymbol{\lambda}_C = 1. \tag{3.3}$$

But by complementary slackness,

$$\begin{aligned} \mathbf{y}_C^{*T} R_C \tilde{\mathbf{x}}_C &= \mathbf{z}_C^{*T} A_C \tilde{\mathbf{x}}_C \\ &= \mathbf{z}_C^{*T} (\mathbf{1} - \mathbf{w}) \\ &= 1 - \mathbf{z}_C^{*T} \mathbf{w} \\ &< 1, \end{aligned}$$

where the last inequality follows from $\mathbf{z}_C^* > \mathbf{0}$. Then this is a contradiction with Equation 3.3, there must exist $(i^*, j^\dagger) \in \mathcal{G}$.

Case 2: Let $(i^*, j^\dagger) \in \mathcal{G}$. We will show that $(i^\dagger, j^*) \in \mathcal{G}$ exists. Because $(i^*, j^\dagger) \in \mathcal{G}$, we have $\tilde{x}_{i^*, j^\dagger} > 0$. Then the vector

$$\mathbf{w} = \left[\sum_{j \in J(C^c)} \mu_{ij} \tilde{x}_{ij} \right]_{i \in I(C)}$$

has all non-negative, and at least one strictly positive component. Specifically, we have $w_{i^*} > 0$. Note that by construction, queues in $I(C)$ receive their remaining demand $\boldsymbol{\lambda}_C - \mathbf{w}$ from servers in $J(C)$.

We want to show that $(i^\dagger, j^*) \in \mathcal{G}$ exists. Assume to the contrary that such incoming arc does not exist. Then $\tilde{\mathbf{x}}$ satisfies

$$\begin{aligned} R_C \tilde{\mathbf{x}}_C &= \boldsymbol{\lambda}_C - \mathbf{w} \\ A_C \tilde{\mathbf{x}}_C &= \mathbf{1}, \end{aligned}$$

where $\tilde{\mathbf{x}}_C$ consists of elements of $\tilde{\mathbf{x}}$ corresponding to arcs in C. By complementary slackness,

$$\begin{aligned} {\mathbf{y}_C^*}^T R_C \tilde{\mathbf{x}}_C &= {\mathbf{z}_C^*}^T A_C \tilde{\mathbf{x}}_C \\ &= {\mathbf{z}_C^*}^T \mathbf{1} \\ &= 1. \end{aligned} \tag{3.4}$$

But by strong duality,

$$\begin{aligned} {\mathbf{y}_C^*}^T R_C \tilde{\mathbf{x}}_C &= {\mathbf{y}_C^*}^T (\boldsymbol{\lambda}_C - \mathbf{w}) \\ &= 1 - {\mathbf{y}_C^*}^T \mathbf{w} \\ &< 1, \end{aligned}$$

where the last inequality follows from $\mathbf{y}_C^* > \mathbf{0}$. Then this is a contradiction with Equation 3.4, there must exist $(i^\dagger, j^*) \in \mathcal{G}$. □

Using the idea that each component has an incoming and and outgoing arc in $\mathcal{G}$, we introduce Algorithm 2 that performs the connection and returns the sparse design $\mathcal{S}$.

Algorithm 2 works as follows. We start with a degenerate basic solution B. By Lemma

6, it is a forest, so that it consists of τ trees. We initialize the collection of components $\mathcal{C}$ to consist of these trees.

Unless we have only one component on hand, we pick an arbitrary component C_0 from the collection $\mathcal{C}$. The idea is to pick an outgoing arc until we create a cycle. That is, we move to another component by picking an outgoing arc. Because there exists an incoming and an outgoing arc for each component, we can continue until the ending queue is in one of the components already visited. If we return to one of the components we visited in this iteration, it means we have completed a cycle. We merge the components in the cycle, which reduces the cardinality of the collection $\mathcal{C}$. Then we proceed to picking an outgoing arc again.

When we start with an arbitrary component C_0 and connect until we have a cycle, the we obtain either an intermediate cycle or a self-contained cycle. An intermediate cycle is created if the cycle does not return to the initial component C_0 but returns to one of the intermediate components visited. In this case, C_0 does not receive an incoming arc. We merge the components making up the cycle, and continue connecting this with other components. Eventually, because the number of components keep reducing and there exists an outgoing arc for each component, there will be an outgoing arc that will be the incoming arc to C_0. With that incoming arc to C_0, we complete a self-contained cycle and we can start the iteration again by picking another arbitrary starting component if $|\mathcal{C}| > 1$.

Because each introduced cycle is part of $\mathcal{G}$, they are breakeven cycles. Therefore, introducing these cycles preserves flow conservation at each node. This procedure results in a connected sparse graph $\mathcal{S}$ on which there exists an optimal solution to SPP($\mathcal{S}$) that uses the entire $\mathcal{S}$. In the next Lemma, we show that each introduced cycle results in another optimal solution to SPP($\mathcal{F}$). Then we show that this procedure terminates in finite time. The following Theorem after that is our main result.

Lemma 7. *Let $\mathcal{C} = C_1, \ldots, C_k$ be components of a disconnected optimal solution $\boldsymbol{x}^*$ to*

SPP($\mathcal{F}$). Let the set of arcs $Y = \{(j_{C_1}, i_{C_2}), (j_{C_2}, i_{C_3}), \dots (j_{C_k}, i_{C_1})\} \in \mathcal{G}$ introduce a cycle that connects these components, where i_{C_r} is a queue in component C_r and j_{C_r} is a server in component C_r.

Then, there exists $\tilde{x}^*_{ij} > 0$ for $(i,j) \in \mathcal{C} \cup Y$ such that $(\tilde{\boldsymbol{x}}^*, \boldsymbol{x}^*_{\mathcal{C}^c})$ is an optimal solution to SPP($\mathcal{F}$), where $\boldsymbol{x}^*_{\mathcal{C}^c}$ refers to the part of $\boldsymbol{x}^*$ not associated with $\mathcal{C}$.

Proof. Let $P^{C_r}(i_{C_r}, j_{C_r})$ denote a path from node i_{C_r} to j_{C_r} in component C_r. Such a path always exists, because C_r is connected. We will propagate some flow on these paths through the introduced cycle. Let

$$W = \{(j_{C_1}, i_{C_2}), P^{C_2}(i_{C_2}, j_{C_2}), (j_{C_2}, i_{C_3}), P^{C_3}(i_{C_3}, j_{C_3}), \dots (j_{C_k}, i_{C_1}), P^{C_1}(i_{C_1}, j_{C_1})\}$$

denote the cycle that arcs in Y introduce. Let $P(j_{C_1}, k) \subseteq W$ denote the path from j_{C_1} to any other node in this cycle. Note that arcs of the form (j_{C_r}, i_{C_r}) are forward arcs. The set of forward arcs in path $P(j_{C_1}, k)$ is denoted by $\overline{P}(j_{C_1}, k)$. Similarly, arcs of the form (i_{C_r}, j_{C_r}) are backward arcs and the set of backward arcs in path $P(j_{C_1}, k)$ is denoted by $\underline{P}(j_{C_1}, k)$.

We choose the propagation amount δ such that

$$\delta < x^*_{ij} \frac{1}{\gamma(P(j_{C_1}, j))} \quad \forall (i,j) \in \underline{P}(j_{C_1}, j_{C_1}).$$

This ensures that the updated flow will be positive on all arcs in W. The updated flow is

$$\tilde{x}^*_{ij} = x^*_{ij} + \gamma(P(j_{C_1}, j))\delta, \quad \text{if} \quad (i,j) \in \overline{W},$$
$$\tilde{x}^*_{ij} = x^*_{ij} - \gamma(P(j_{C_1}, j))\delta, \quad \text{if} \quad (i,j) \in \underline{W}$$

Because $W \in \mathcal{G}$, it is a breakeven cycle, so that $\gamma(P(W)) = 1$. Then if δ amount of flow is pushed from node j_{C_1}, $\gamma(P(j_{C_1}, j_{C_1}))\delta = \gamma(W)\delta = \delta$ is received at node j_{C_1}. Then for each

server node, we have

$$\sum_{i \in \bigcup_{r=1}^{k} I(C_r)} \tilde{x}^*_{ij} = 1 \quad \forall j \in \bigcup_{r=1}^{k} J(C_r). \tag{3.5}$$

Each queue node is adjacent to a forward arc and a backward arc in the cycle W. For each queue node, the forward arc is traversed before the backward arc in the cycle. If j' leads the forward arc to queue i in cycle W, and j'' leads the backward arc from queue i in cycle W,

$$\mu_{ij'} \tilde{x}^*_{ij'} = \mu_{ij'} x^*_{ij'} + \mu_{ij'} \gamma(P(j_{C_1}, j')) \delta$$
$$\mu_{ij''} \tilde{x}^*_{ij''} = \mu_{ij''} x^*_{ij''} - \mu_{ij''} \gamma(P(j_{C_1}, j'')) \delta.$$

But j' and j'' have only two arcs between them in the cycle: (j', i) and (i, j''). Then

$$\gamma(P(j_{C_1}, j'')) = \frac{\mu_{ij'} \gamma(P(j_{C_1}, j'))}{\mu_{ij''}}.$$

Then for queue i, we have

$$\mu_{ij'} \tilde{x}^*_{ij'} + \mu_{ij''} \tilde{x}^*_{ij''} = \mu_{ij'} x^*_{ij'} + \mu_{ij''} x^*_{ij''},$$

so that feasibility is preserved. Then more generally, we have

$$\sum_{j \in \bigcup_{r=1}^{k} J(C_r)} \mu_{ij} \tilde{x}^*_{ij} = \lambda_i \quad \forall i \in \bigcup_{r=1}^{k} I(C_r). \tag{3.6}$$

Hence, equations 3.5 and 3.6 imply that $\tilde{\mathbf{x}}^*$ preserves feasibility in the collection of components $C_1, \ldots C_k$. The rest of the network is not affected. Then $(\tilde{\mathbf{x}}^*, \mathbf{x}^*_{\mathcal{C}^c})$ is an optimal solution to SPP($\mathcal{F}$). □

Algorithm 2: Sparse Flexibility Design

Input : Optimal basis B, support graph $\mathcal{G}$

Collection of components: $\mathcal{C} := B$.

$\mathcal{S}' := B$.

while $|\mathcal{C}| > 1$ **do**

  Pick an arbitrary $C_0 \in \mathcal{C}$.

  Initialize component counter $k = 0$, $i_{k+1} = \emptyset$.

  while $i_{k+1} \notin C_0$ **do**

    Pick outgoing arc $(i_{k+1}, j_k) \in \mathcal{G}$ from C_k to some $C_{k+1} \in \mathcal{C}$.

    if $\exists\, k^* \in \{1, \ldots, k\}$ *s.t.* $i_{k+1} \in I(C_{k^*})$ **then**

      Add cycle inducing arcs to the design
      $\mathcal{S}' \leftarrow \mathcal{S}' \cup \{(i_{k^*+1}, j_{k^*}), (i_{k^*+2}, j_{k^*+1}) \ldots (i_k, j_{k-1}), (i_{k^*}, j_k)\}$.

      Merge the components in this cycle:
      $C_{k^*} \leftarrow \left\{ \bigcup_{t=k^*}^{k} C_t \cup \{(i_{k^*+1}, j_{k^*}), (i_{k^*+2}, j_{k^*+1}) \ldots (i_k, j_{k-1}), (i_{k^*}, j_k)\} \right\}$

      Remove merged components from the collection: $\mathcal{C} \leftarrow \left(\mathcal{C} \setminus \bigcup_{t=k^*+1}^{k} C_t \right)$

      $k \leftarrow k^*$

    else

      $k \leftarrow k + 1$

    end if

  end while

  Add cycle inducing arcs to the design
  $\mathcal{S}' \leftarrow \mathcal{S}' \cup \{(i_1, j_0), (i_2, j_1) \ldots (i_k, j_{k-1}), (i_0, j_k)\}$.

  Merge the components in this cycle:
  $C_0 \leftarrow \left\{ \bigcup_{t=0}^{k} C_t \cup \{(i_1, j_0), (i_2, j_1) \ldots (i_k, j_{k-1}), (i_0, j_k)\} \right\}$.

  Remove merged components from the collection $\mathcal{C} \leftarrow \left(\mathcal{C} \setminus \bigcup_{t=1}^{k} C_t \right)$.

end while

Return: $\mathcal{S} := \mathcal{S}'$

Lemma 8. *Algorithm 2 terminates in a finite number of iterations.*

Proof. We will show this by induction. Let $|\mathcal{C}|$ denote the number of components in $\mathcal{C}$ at the start of the algorithm.

- Base Case: The outer loop starts with $|\mathcal{C}| = 1$ component. The algorithm directly terminates.
- Induction Hypothesis: The outer loop stars with $|\mathcal{C}| \leq \kappa$ components, where $1 < \kappa < \tau$. Assume that the algorithm terminates.

- Next Case: The outer loop stars with $|\mathcal{C}| = \kappa + 1$ components. An arbirary component $C_0 \in |\mathcal{C}|$ is picked. By Proposition 4, an outgoing arc (i_1, j_0) exists, where $j_0 \in J(C_0)$ and $i_1 \in I(C_1)$ for some $C_1 \in \mathcal{C}$. Then (i_1, j_0) is an incoming arc to component C_1. By Proposition 4, an outgoing arc (i_2, j_1) exists, where $j_1 \in J(C_1)$ and $i_2 \in I(C_2)$ for some $C_2 \in \mathcal{C}$.

  - If $C_2 = C_0$, a cycle is completed. The algorithm merges C_0, C_1. The number of components is reduced by one, so that the ending number of components $|\mathcal{C}| = \kappa$. By induction hypothesis, the algorithm terminates.
  - If $C_2 \neq C_0$, by Proposition 4, an outgoing arc (i_3, j_2) exists, where $j_2 \in J(C_2)$ and $i_3 \in I(C_3)$ for some $C_3 \in \mathcal{C}$.
    - If $C_3 \in \{C_0, C_1\}$ a cycle is completed. Components in the cycle are merged, that corresponds to either two or three components being merged, so that the ending number of components $|\mathcal{C}| \in \{\kappa, \kappa - 1\}$. By induction hypothesis, the algorithm terminates.
  - If a cycle is not completed until $C_{\kappa-1}$, all the other components are already visited. By Proposition 4, its outgoing arc must be towards one of the visited components, resulting in a cycle being completed. Then the components in the cycle are merged and $|\mathcal{C}| \in \{\kappa, \kappa - 1, \ldots, 1\}$. By induction hypothesis, the algorithm terminates.

With each cycle completion, the number of components is reduced by at least one. The algorithm terminates when the number of components reaches one. Then the algorithm terminates in finite number of operations. □

Theorem 3. *There exists a sparse flexibility design $\mathcal{S}$ with*

$$|\mathcal{S}| \leq m + n + \min\{m, n\} - 2$$

such that DSPP($\mathcal{S}$) has a unique solution.

Proof. Let $\mathbf{x}^*$ denote the initial basic solution to SPP($\mathcal{F}$). Consider Algorithm 2. We will show that there exists an optimal solution $\tilde{\mathbf{x}}^*$ on to SPP($\mathcal{S}$) that uses the entire graph. In particular, we will show by induction that at each iteration, Algorithm 2 yields an optimal solution to SPP($\mathcal{F}$). Then the ending $\mathcal{S}$ will have a solution to SPP($\mathcal{S}$) that uses the entire $\mathcal{S}$. Moreover, this proof implies that the Algorithm does not remove arcs from $\mathcal{S}$, but monotonically adds cycles.

By Lemma 8, the algorithm terminates. Let the algorithm terminate in L iterations. At each iteration l, a cycle is introduced. Let r_l denote the number of components merged in iteration l. This also denotes the number of arcs added to $\mathcal{S}'$ in iteration l.

- Base Case: $l = 1$. The algorithm starts with $|\mathcal{C}| = \tau$ components. At $l = 1$, we merge $2 \leq r_l \leq \tau$ components with a cycle W_1. By Lemma 7, we can push $\delta_1 > 0$ flow on this cycle, and there exists an optimal solution $\tilde{\mathbf{x}}_1^*$ to SPP($\mathcal{F}$) such that

$$[\tilde{\mathbf{x}}_1^*]_{ij} > 0 \text{ for } (i,j) \in \Big\{(i,j) : x_{ij}^* > 0\Big\} \cup \{(i,j) \in W_1\}$$

  Then optimality is preserved.

- Induction Hypothesis: Assume that at iteration $l = \ell$, merge r_ℓ components by introducing cycle W_ℓ, and there exists an optimal solution $\tilde{\mathbf{x}}_\ell^*$ to SPP($\mathcal{F}$) such that

$$[\tilde{\mathbf{x}}_\ell^*]_{ij} > 0 \text{ for } (i,j) \in \big\{(i,j) : [\tilde{\mathbf{x}}_\ell^*]_{ij} > 0\big\} \cup \{(i,j) \in W_\ell\},$$

  so that optimality is preserved.

- Next Case: At iteration $l = \ell + 1$, we merge r_ℓ components by introducing cycle $W_{\ell+1}$. By Lemma 7, we can push $\delta_{\ell+1} > 0$ flow on this cycle and there exists an optimal

solution $\tilde{\mathbf{x}}^*_{\ell+1}$ to SPP($\mathcal{F}$ such that

$$[\tilde{\mathbf{x}}^*_{\ell+1}]_{ij} > 0 \text{ for } (i,j) \in \left\{(i,j) : [\tilde{\mathbf{x}}^*_{\ell+1}]_{ij} > 0\right\} \cup \left\{(i,j) \in W_{\ell+1}\right\},$$

so that optimality is preserved.

Therefore, the algorithm preserves optimality in SPP($\mathcal{F}$) in each iteration. At the end, there is only one component left and none of the added cycles are removed, implying that $\mathcal{S}$ is connected. Restricting the static planning problem to $\mathcal{S}$, by Lemma 4, DSPP($\mathcal{S}$) has a unique solution.

Now we will show that $|\mathcal{S}| \leq m + n + \min\{m, n\} - 2$. Let B be the starting optimal degenerate basis to SPP($\mathcal{F}$) with τ trees. Then B consists of $m+n-\tau$ arcs. Let $v(\tau)$ denote the number of arcs introduced to basis B by Algorithm 2. At iteration l, the algorithm connects $r_l \geq 2$ components by introducing r_l arcs to $|\mathcal{S}|$, which reduces the number of components by $r_l - 1$. Then $v(\tau) = r_1 + r_2 + \ldots + r_L$. The algorithm terminates when the number of components reaches one.

$$\begin{aligned} &\tau - (r_1 - 1) - (r_2 - 1) - \ldots - (r_L - 1) = 1, \\ &r_1 + r_2 + \ldots + r_L = \tau + L - 1. \end{aligned}$$

We have $L \leq \tau - 1$, the equality is realized if we reduce the components one by one. Then $v(\tau) \leq \tau + (\tau - 1) - 1 = 2(\tau - 1)$. Because B is a forest, it can have at most $\min\{m, n\}$

components. Then

$$\begin{aligned}|\mathcal{S}| &= m + n - \tau + v(\tau) \\ &\leq m + n - \tau + 2(\tau - 1) \\ &= m + n + \tau - 2 \\ &\leq m + n + \min\{m, n\} - 2.\end{aligned}$$

□

We have shown that by systematically introducing cycles to an initial degenerate basis, we can obtain a sparse design $\mathcal{S}$ such that the heavy traffic and CRP assumptions hold for $\mathcal{S}$. When these assumptions hold for the system's flexibility structure, performance reasonably close to that of $\mathcal{F}$ is expected from control policies. In the next section, we numerically demonstrate that indeed, $\mathcal{S}$ gets close to $\mathcal{F}$ in terms of performance. Moreover, we demonstrate that haphazardly connected graphs starting from an initial basis B do not necessarily perform better than B. Algorithm 2 provides a systematic way of connecting the initial disconnected graph so as to obtain the CRP property.

3.5 Numerical Experiments

So far, we have assumed that Assumptions **A1.-A3.** hold for $\mathcal{F}$, then constructed sparse $\mathcal{S}$ ensuring Assumptions **A1.-A3.** still hold for $\mathcal{S}$. We have argued that when these assumptions hold, $\mathcal{S}$ should have performance close to that of $\mathcal{F}$. In this section, we numerically justify that discussion. We compare the performance of $\mathcal{S}$ to that of $\mathcal{F}$ in different pre-limit settings.

Our model is a discrete time parallel server system, and we adopt the MaxWeight policy to schedule the system (Tassiulas and Ephremides 1990, Stolyar et al. 2004). We adapt the description of MaxWeight given in Shi et al. (2019) to our setting. System dynamics are

as follows. $Q_i(t)$ denotes the length of queue i at the end of period t. At the beginning of each time unit $t \geq 0$, queue i sees a random number of arrivals denoted by $A_i(t)$. For each i, $A_i(t)$ is distributed according to a truncated normal distribution, independently across queues and periods. The distribution has mean λ_i, and is truncated below at zero and above at $2\lambda_i$. The distribution has coefficient of variation of 0.3.

After observing arrivals, each server j serves queue i at a rate of μ_{ij}, deterministically. When assumptions **A1.-A3.** hold and the service is deterministic, MaxWeight asymptotically minimizes

$$\limsup_{T\to\infty} L(T)$$

where

$$L(T) = \frac{1}{T}\sum_{t=1}^{T}\sum_{i\in\mathcal{I}} \mathbb{E}\left[y_i^* Q_i(t)\right],$$

and $\mathbf{y}^*$ is the unique solution of the associated DSPP (Eryilmaz and Srikant 2012). However, when the DSPP associated with the system does not have a unique solution, MaxWeight has no optimality guarantees.

In our system, MaxWeight works as follows. At each time unit t, the policy chooses a

processing schedule $\mathbf{g}(t) \in \mathbb{R}^n_+$ according to the linear program below.

$$
\begin{aligned}
(MW) \qquad \max_{\mathbf{g}(t),\mathbf{x}} \quad & \sum_{i\in\mathcal{I}} (Q_i(t-1) + A_i(t)) g_i(t) \\
s.t. \quad & g_i(t) \le Q_i(t-1) + A_i(t) \quad \forall i \in \mathcal{I} \\
& \sum_{\substack{j\in\mathcal{J}: \\ (i,j)\in\mathcal{S}}} \mu_{ij} x_{ij} = g_i(t) \quad \forall i \in \mathcal{I} \\
& \sum_{\substack{i\in\mathcal{I}: \\ (i,j)\in\mathcal{S}}} x_{ij} \le 1 \quad \forall j \in \mathcal{J} \\
& \mathbf{g}(t) \ge \mathbf{0}
\end{aligned}
$$

We consider two different systems, and compare $\mathcal{F}$, the starting basis used as a flexibility design $\mathcal{B}$, an intermediate design with arcs added from the support graph to the starting basis denoted by $\mathcal{P}$, and our proposed design $\mathcal{S}$.

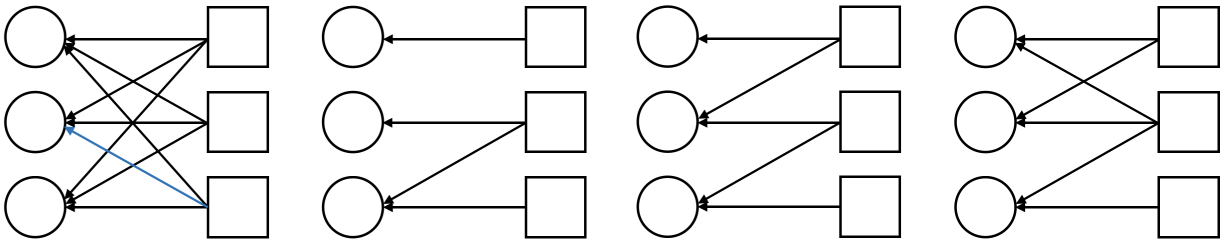
The first system consists of $m = 3$ and $n = 3$, and Table 3.1 contains the service rate and heavy traffic arrival rate information. SPP($\mathcal{F}$) has multiple solutions, and DSPP($\mathcal{F}$) has a unique solution. To test the uniqueness of solutions in these LPs, we use the PUFAS algorithm suggested in Appa (2002). We obtain $\mathcal{G}$ according to Algorithm 1.

		Queues		
	$\lambda_i =$	10	10	12
Servers	$\mu_{i1} =$	10	10	6
	$\mu_{i2} =$	15	15	9
	$\mu_{i3} =$	15	6	9

Table 3.1: Service and heavy traffic arrival rates for simulated system 1

$\mathcal{F}$, $\mathcal{B}$, $\mathcal{P}$ and $\mathcal{S}$ for the first system are given in Figure 3.4. In $\mathcal{F}$, all arcs except the blue highlighted one are also present in the support graph $\mathcal{G}$.

To test the performance in pre-limit settings, we scale down the heavy traffic arrival rates given in Table 3.1 by $[0.9, 0.95, 0.99]$. We run the system for 1000 time units for $\rho = 0.9$ and



(a) $\mathcal{F}$ (b) $\mathcal{B}$ (c) $\mathcal{P}$ (d) $\mathcal{S}$

Figure 3.4: Graphs for simulated system 1

$\rho = 0.95$, and for 10,000 time units for $\rho = 0.99$. We calculate the long run average weighted cost by taking the average values of 50 sample paths in each setting. We report the long-run average costs in in Figure 3.5, and ratio of costs of $\mathcal{F}$ and $\mathcal{S}$ in Figure 3.6. The ratio of costs of $\mathcal{F}$ and $\mathcal{S}$ at time T is defined as

$$\mathcal{D}(T) = \frac{L_{\mathcal{S}}(T)}{L_{\mathcal{F}}(T)},$$

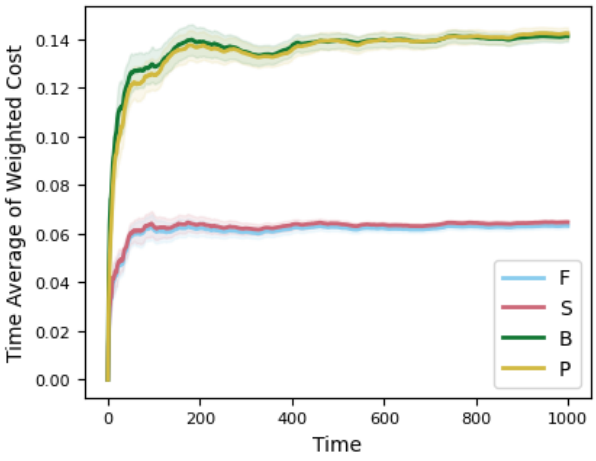
where $L_{\mathcal{S}}(T)$ and $L_{\mathcal{F}}(T)$ denote $L(T)$ under designs $\mathcal{S}$ and $\mathcal{F}$, respectively. For $\mathcal{D}(T)$, we report the values starting at time 200 in order to allow the system to somewhat converge to its steady state.

As can be seen in the figures, $\mathcal{S}$ significantly outperforms $\mathcal{B}$ and $\mathcal{P}$, and performs very close to $\mathcal{F}$ in all settings. As system load increases, performance of $\mathcal{S}$ gets closer to $\mathcal{F}$.

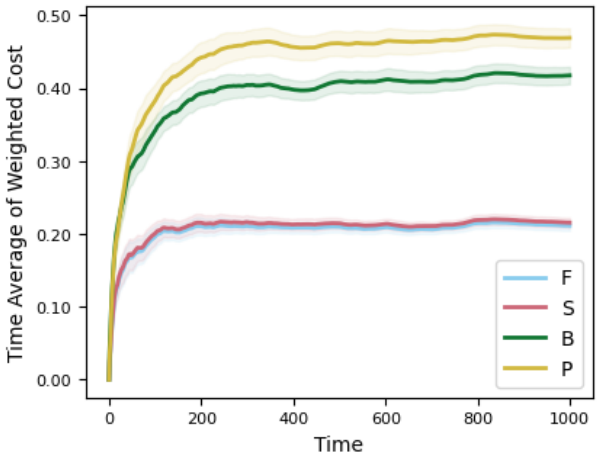
The second system we consider consists of $m = 6$ servers and $n = 9$ queues, and Table 3.2 contains the service rate and heavy-traffic arrival rate information. Again, SPP($\mathcal{F}$) has multiple solutions, and DSPP($\mathcal{F}$) has a unique solution, verified by PUFAS algorithm (Appa 2002). $\mathcal{G}$ is obtained using Algorithm 1.

$\mathcal{F}$, $\mathcal{B}$, $\mathcal{P}$ and $\mathcal{S}$ for the second system are given in Figure 3.7. In $\mathcal{F}$, all arcs except the blue highlighted ones are also present in the support graph $\mathcal{G}$.

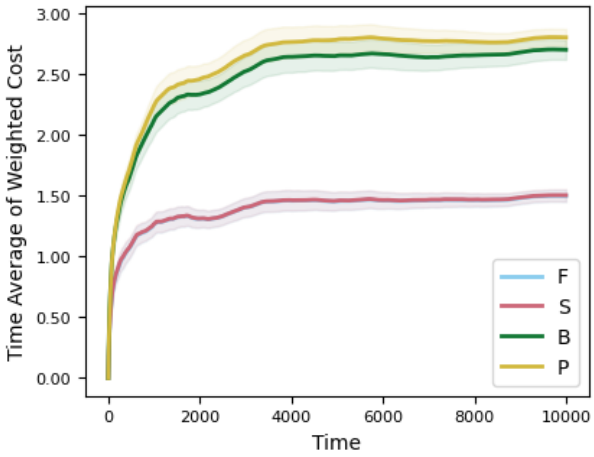
Similar to the first system, we scale down the heavy-traffic arrival rates given in Table 3.2 by $[0.9, 0.95, 0.99]$. We run the system for 1000 time units for loads 0.9 and 0.95, and for



(a) $L(T)$ with load $\rho = 0.9$



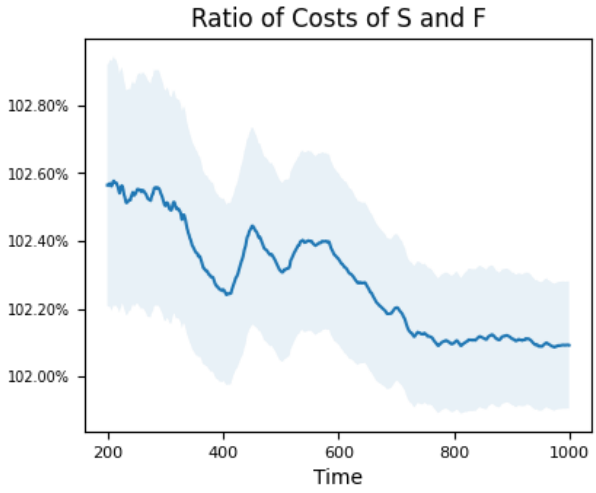
(b) $L(T)$ with load $\rho = 0.95$



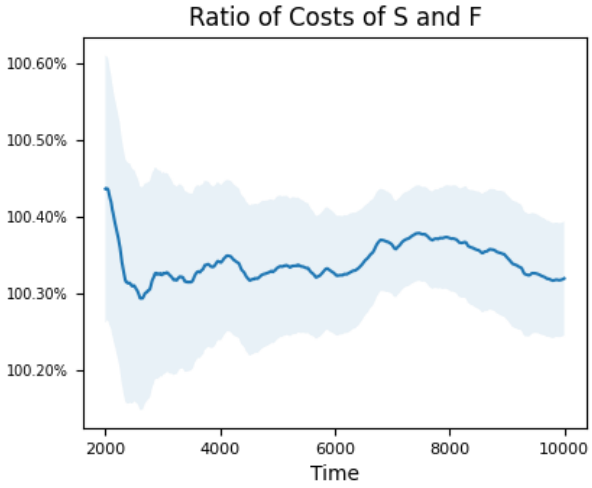
(c) $L(T)$ with load $\rho = 0.99$

Figure 3.5: Long-run Average Costs for Simulated System 1

(a) $\mathcal{D}(T)$ where $\rho = 0.9$



(b) $\mathcal{D}(T)$ where $\rho = 0.95$

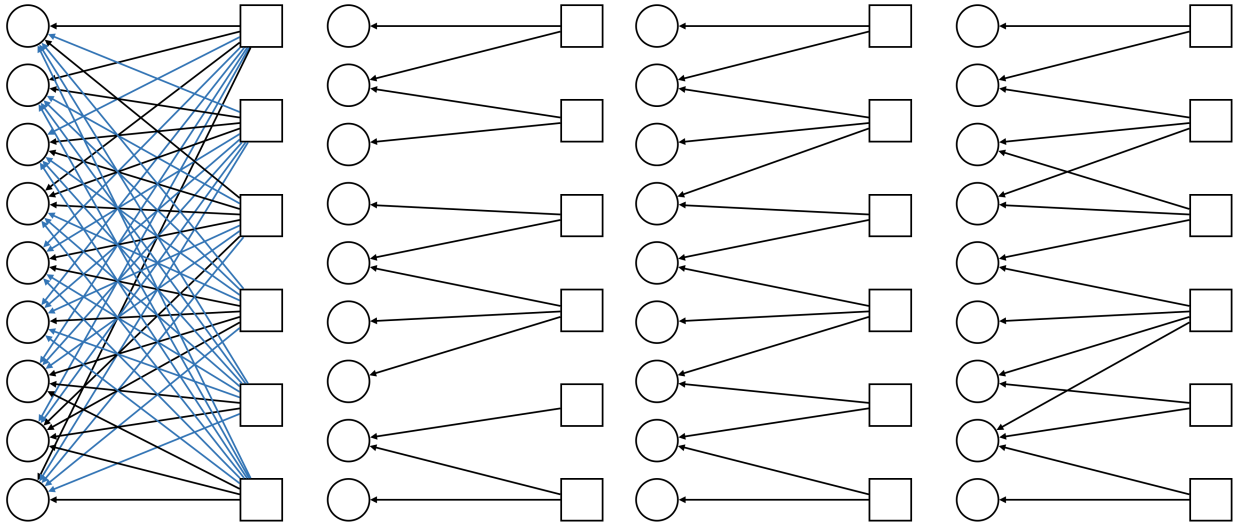


(c) $\mathcal{D}(T)$ where $\rho = 0.99$

Figure 3.6: Relative Difference of Costs of $\mathcal{S}$ and $\mathcal{F}$ for Simulated System 1

		Queues								
	$\lambda_i =$	16	25	36	1	36	25	81	144	9
Servers	$\mu_{i1} =$	28	35	1	7	1	1	1	1	21
	$\mu_{i2} =$	1	40	48	8	1	1	1	1	1
	$\mu_{i3} =$	20	1	30	5	30	1	1	60	1
	$\mu_{i4} =$	1	1	1	1	96	80	144	192	1
	$\mu_{i5} =$	1	1	1	1	1	1	63	84	1
	$\mu_{i6} =$	1	1	1	1	1	1	72	96	24

Table 3.2: Service and heavy traffic arrival rates for simulated system 2

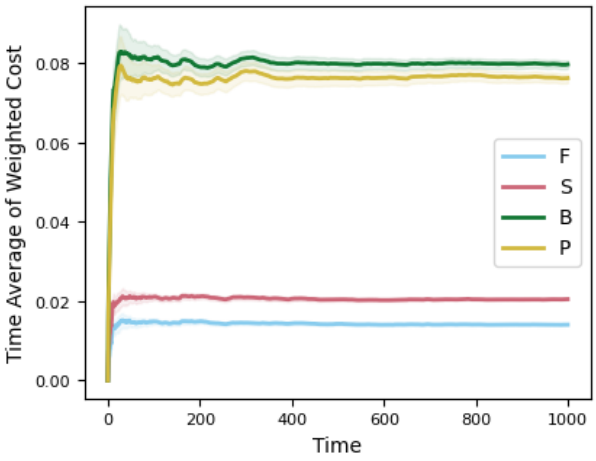


(a) $\mathcal{F}$ (b) $\mathcal{B}$ (c) $\mathcal{P}$ (d) $\mathcal{S}$

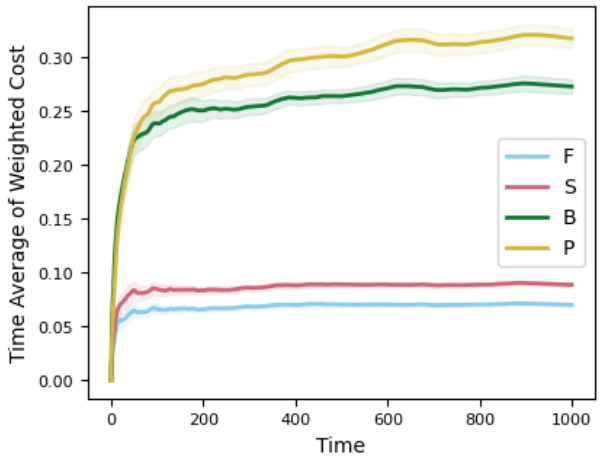
Figure 3.7: Graphs for simulated system 2

10,000 time units for load 0.99. We calculate the long run average weighted cost by taking the average values of 50 sample paths in each case. In Figure 3.8 we report the long-run average costs, and in Figure 3.9, we report the ratio of costs of $\mathcal{F}$ and $\mathcal{S}$.

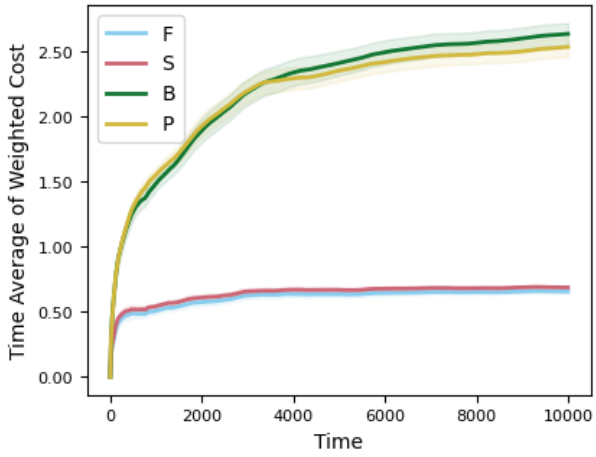
We see that as the system load ρ gets closer to one, $\mathcal{S}$ gets closer to the performance of $\mathcal{F}$. Moreover, for the same pre-limit loads, the gap between the performance of $\mathcal{S}$ and $\mathcal{F}$ is smaller for the first system. This implies that $\mathcal{S}$ has asymptotic performance closer to $\mathcal{F}$ when the system is smaller, and the system load is closer to one; all consistent results with the flexibility design literature. This example also highlights the importance of adding the final two arcs, comparing $\mathcal{P}$ and $\mathcal{S}$. Where $\mathcal{P}$ performs close to the disconnected solution $\mathcal{B}$,



(a) $L(T)$ with load $\rho = 0.9$



(b) $L(T)$ with load $\rho = 0.95$



(c) $L(T)$ with load $\rho = 0.99$

Figure 3.8: Long-run Average Costs for Simulated System 2

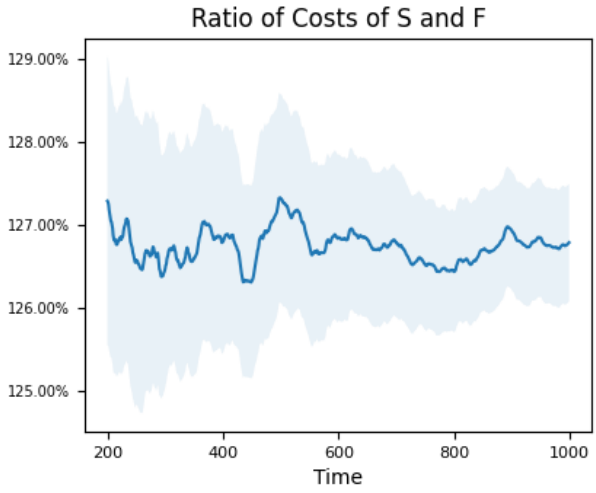
just by adding two arcs, $\mathcal{S}$ performs close to $\mathcal{F}$.

3.6 Discussion and Conclusion

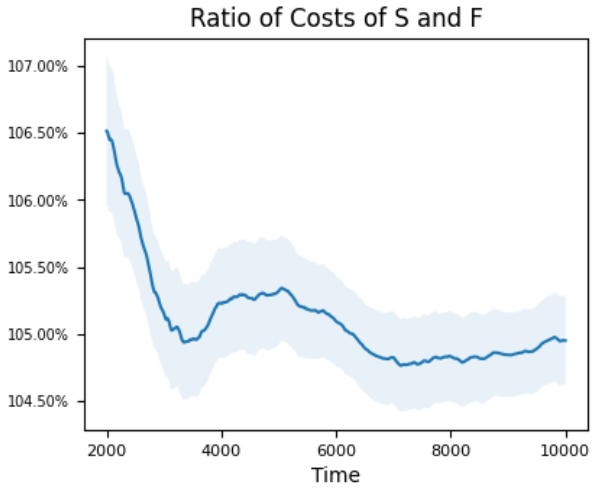
We considered the flexibility structure design problem in systems where the processing rates are general. To our knowledge, this is the first work that addresses the design problem with general processing rates.

We assumed that the full flexibility structure $\mathcal{F}$ satisfies the heavy traffic and CRP

(a) $\mathcal{D}(T)$ where $\rho = 0.9$



(b) $\mathcal{D}(T)$ where $\rho = 0.95$



(c) $\mathcal{D}(T)$ where $\rho = 0.99$

Figure 3.9: Relative Difference of Costs of $\mathcal{S}$ and $\mathcal{F}$ for Simulated System 2

assumptions, but allowed that SPP($\mathcal{F}$) can have multiple solutions. We showed that there exists a sparse flexibility structure $\mathcal{S}$ with $O(m+n)$ arcs that satisfies the same heavy traffic and CRP assumptions, with the expectation that CRP would ensure performance close to that of $\mathcal{F}$.

We presented an algorithm that constructs the sparse design $\mathcal{S}$, and numerically justified our proposed design. We demonstrated that as expected, CRP property of $\mathcal{S}$ enables the sparse design to achieve performance close to that of $\mathcal{F}$. Our construction method that builds instances to test our design indicates that degeneracy is actually not an uncommon problem in the SPP.

It is important to note here that our design methodology only guarantees that CRP holds for the sparse design $\mathcal{S}$. In the terminology of Shi et al. (2019), our algorithm guarantees that $GCG > 0$, as opposed to $GCG > \eta$ where η does not depend on the solution to SPP($\mathcal{S}$). An interesting future research direction is to consider the design problem for general processing rates, but trying to ensure $GCG > \eta$. If we can do that, intuitively, we expect that the performance gap between $\mathcal{S}$ and $\mathcal{F}$ would get smaller for pre-limit systems such as the simulated system 2 in Section 3.5.

Another immediate future research direction is to theoretically show that $\mathcal{S}$ achieves performance close to $\mathcal{F}$ with policies other than MaxWeight, and possibly under random service realizations. MaxWeight minimizes a specifically weighted linear cost of keeping backlog, but arbitrary linear costs are also interesting. We suspect that threshold-type policies similar to Bell and Williams (2005) might be promising for arbitrary linear holding costs.

We could also consider pre-limit systems approaching heavy-traffic. In the existing literature, projection of such systems to the boundary of the capacity region is a more straightforward task because the capacity region has one "smooth" face. The same task becomes a non-trivial when processing rates are general, because the capacity region might have several

faces with "kink" points in between. In such cases, it is unclear which point on the boundary of the capacity region best captures the behavior of a pre-limit system.

A more ambitious goal would be to address the design problem when $\mathcal{F}$ has multiple resource pools in heavy traffic. Varma and Maguluri (2021) address this problem when the system has flow conservation, but the problem remains open when the processing rates are general.

Since flow conservation might not be observed in practice, we believe understanding systems with general processing rates more deeply holds significant practical importance.

CHAPTER 4

EXTENSIONS AND RELATED TOPICS

In this chapter, we address several additional topics related to the first two chapters. In Section 4.1, we explore parameter agnostic policies that are not switching curve policies. Some of the results in Chapter 2 naturally extend to more general policies, and we also show a new instability result for one of the cases where previous results do not extend. In Section 4.2, we highlight an interesting connection between the flexibility problem addressed in Chapter 3 and the so-called generalized transportation problem. Using this connection, we present an application of Algorithm 2 to a special setting. The same connection also allows us to derive a new characterization of the uniqueness of the solution of the static planning problem in Section 4.2.2.

4.1 Instability Results for Non-Switching Curve Parameter Agnostic Policies

In Chapter 2, we derived sufficient conditions for instability for various cases of switching curve policies. Many of these structures that we imposed on the parameter agnostic policies can actually be relaxed, and the instability results continue to hold.

In Section 2.3, we showed a sufficient condition for instability using a test function $w_1\bar{Q}_1(t) + w_2\bar{Q}_2(t)$. We showed that this test function has positive drift everywhere in the state space. Our proof did not make use of the fact that service decision was made according to a switching curve policy. The only critical assumption related to the policy was that in the entire state space, only three of the four possible service configurations was used. Therefore, the instability condition in 2 naturally still holds if the parameter agnostic policy is an arbitrary one that only uses three of the four available service configurations.

Similarly, in Section 2.4, we derived a sufficient condition for instability using a test

function that utilizes the fluid model, and we showed that the test function has positive drift everywhere. Then this result naturally extends to all parameter agnostic policies where the servers are synchronized. That is, as long as both servers serve the same queue at the same time, the sufficient condition for instability in Proposition 3 is still valid.

The instability result for intersecting switching curves cannot be similarly extended naturally. Intuitively, this is due to these policies utilizing a service configuration that is also used in maximally stable policies, providing stabilizing properties to the policy. However, we are able to extend the instability result for intersecting switching curves to cases where the policy has a "free corridor". More specifically, let the policy be such that for some $C > 0$,

- when $Q_2(t) \geq Q_1(t) + C$, both servers serve queue 2,
- when $Q_2(t) \leq Q_1(t) - C$, both servers serve queue 1,
- when $Q_1(t) - C < Q_2(t) < Q_1(t) + C$, the policy is arbitrary, it may use all four service configurations in this free corridor.

Proposition 5. *If the policy has a "free corridor" of the form $q_1 \pm C$ for some $C > 0$, and both servers serve queue 2 above the corridor, and both servers serve queue 1 below the corridor, there exist $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$ such that the system is unstable.*

Proof. Consider a symmetrical parameter setting where $\lambda_1 = \lambda_2 := \lambda$, $\mu_{11} = \mu_{22} := \mu$ and $\mu_{12} = \mu_{21} = \mu'$ with $\boldsymbol{\lambda} \in \Lambda(\boldsymbol{\mu})$. We will consider the uniformized discrete time Markov chain underlying the continuous time Markov chain $(\mathbf{Q}(t))_{t \geq 0}$, with the uniformization constant $P = 2\lambda + 2\mu + 2\mu'$. Let $s_1 \geq 0$ and $s_2 \geq 0$ denote the service rates to queue 1 and queue 2 determined by the scheduling policy.

We will show that there exists a test function $V : \mathbb{Z}_+^2 \to \mathbb{R}_+$ such that for a finite set $E_0 \subset \mathbb{Z}_+^2$,

1. For all $\mathbf{q} \notin E_0$, $\Delta V(\mathbf{q}) \geq 0$,

2. There exists some $\mathbf{q}^* \notin E_0$ such that for all $\mathbf{q} \in E_0$, $V(\mathbf{q}^*) > V(\mathbf{q})$,

3. $V(\mathbf{q})$ is bounded below and for all $\mathbf{q} \in E_0$, $\Delta V(\mathbf{q}) < A$ for some $A < \infty$.

The drift operator $\Delta V(\mathbf{q})$ is defined as

$$\Delta V(\mathbf{q}) := \mathbb{E}[V(\mathbf{q}(1)) - V(\mathbf{q}(0)) \mid \mathbf{q}(0) = \mathbf{q}].$$

These conditions will imply that $(\mathbf{Q}(t))_{t\geq 0}$ cannot be positive recurrent (Meyn and Tweedie 2012). For some $M > 0$ whose value will be discussed later, let the proposed test function

$$V(\mathbf{q}) = q_1 + q_2 + M\left[(C+1)|q_1 - q_2| + \sum_{i=1}^{C} ||q_1 - q_2| - i|\right].$$

This test function was initially suggested by Farias et al. (2005) for a simple parallel server system setting where service rates are identical, and the jobs generate penalties if they decide to migrate to the dedicated queue of another server.

For the proposed V, the last two conditions hold. We will show that condition 1 holds. To do that, we define the following sets to distinguish where the scheduling policy employs different service configurations.

$$\begin{aligned}
A_1 &= \{(q_1, q_2) \in \mathbb{Z}_+^2 : q_2 \leq q_1 - C,\ s_1 = \mu + \mu',\ s_2 = 0\},\\
A_2 &= \{(q_1, q_2) \in \mathbb{Z}_+^2 : q_2 \geq q_1 + C,\ s_1 = 0,\ s_2 = \mu + \mu'\},\\
B_1 &= \{(q_1, q_2) \in \mathbb{Z}_+^2 : q_1 - C < q_2 < q_1 + C,\ s_1 = \mu,\ s_2 = \mu\},\\
B_2 &= \{(q_1, q_2) \in \mathbb{Z}_+^2 : q_1 - C < q_2 < q_1 + C,\ s_1 = \mu',\ s_2 = \mu'\},\\
B_3 &= \{(q_1, q_2) \in \mathbb{Z}_+^2 : q_1 - C < q_2 < q_1 + C,\ s_1 = \mu + \mu',\ s_2 = 0\},\\
B_4 &= \{(q_1, q_2) \in \mathbb{Z}_+^2 : q_1 - C < q_2 < q_1 + C,\ s_1 = 0,\ s_2 = \mu + \mu'\}.
\end{aligned}$$

Figure 4.1 illustrates how the state space is partitioned into these sets.

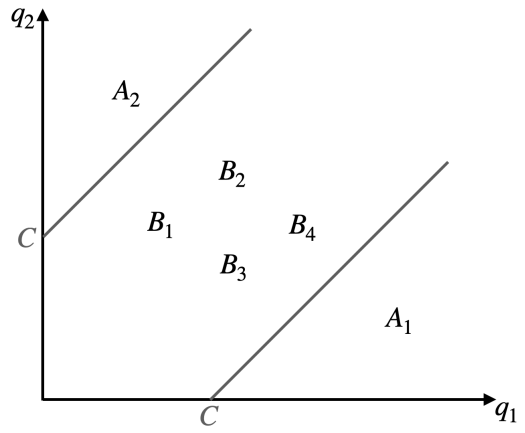


Figure 4.1: Partitioning of the State Space of $(\mathbf{Q}(t))_{t\geq 0}$

In each of these sets, V must have non-negative drift. We will consider each set separately. Let $\Delta V_{\mathcal{S}}(\mathbf{q})$ denote the drift in set $\mathcal{S} \in \{A_1, A_2, B_1, B_2, B_3, B_4\}$. Because of the symmetries in the parameter setting, some of these sets have the same drift expression with the proposed V.

$$\Delta V_{A_1}(\mathbf{q}) = \Delta V_{A_2}(\mathbf{q}) = \frac{2\lambda - \mu - \mu'}{P} + M\left[(2C+1)\frac{(-\mu-\mu')}{P}\right] \tag{4.1}$$

$$\Delta V_{B_1}(\mathbf{q}) = \frac{2\lambda - 2\mu}{P} + M\left[\frac{2\lambda + 2\mu}{P}\right] \tag{4.2}$$

$$\Delta V_{B_2}(\mathbf{q}) = \frac{2\lambda - 2\mu'}{P} + M\left[\frac{2\lambda + 2\mu'}{P}\right] \tag{4.3}$$

For sets B_3 and B_4, two additional cases should be considered. Let $\ell := |q_1 - q_2|$, and note that in B_3 and B_4, $0 \leq \ell < C$. Again due to symmetries in the assumed parameter setting, the drift conditions for some these cases are the same.

- If $\ell = |q_1 - q_2| = q_1 - q_2 \geq 0$,

$$\begin{aligned}
\Delta V_{B_3}(\mathbf{q}) &= \frac{2\lambda - \mu - \mu'}{P} + M\left[2\ell\frac{(-\mu - \mu')}{P} + \frac{2\lambda + \mu + \mu'}{P}\right] \\
&\geq \frac{2\lambda - \mu - \mu'}{P} + M\left[2C\frac{(-\mu - \mu')}{P} + \frac{2\lambda + \mu + \mu'}{P}\right], \qquad (4.4) \\
\Delta V_{B_4}(\mathbf{q}) &= \frac{2\lambda - \mu - \mu'}{P} + M\left[2\ell\frac{(\mu + \mu')}{P} + \frac{2\lambda + \mu + \mu'}{P}\right] \\
&\geq \frac{2\lambda - \mu - \mu'}{P} + M\left[\frac{2\lambda + \mu + \mu'}{P}\right]. \qquad (4.5)
\end{aligned}$$

- If $\ell = |q_1 - q_2| = q_2 - q_1 > 0$,

$$\begin{aligned}
\Delta V_{B_3}(\mathbf{q}) &= \frac{2\lambda - \mu - \mu'}{P} + M\left[2\ell\frac{(\mu + \mu')}{P} + \frac{2\lambda + \mu + \mu'}{P}\right] \\
&\geq \frac{2\lambda - \mu - \mu'}{P} + M\left[\frac{2\lambda + \mu + \mu'}{P}\right], \qquad (4.6) \\
\Delta V_{B_4}(\mathbf{q}) &= \frac{2\lambda - \mu - \mu'}{P} + M\left[2\ell\frac{(-\mu - \mu')}{P} + \frac{2\lambda + \mu + \mu'}{P}\right] \\
&\geq \frac{2\lambda - \mu - \mu'}{P} + M\left[2C\frac{(-\mu - \mu')}{P} + \frac{2\lambda + \mu + \mu'}{P}\right]. \qquad (4.7)
\end{aligned}$$

For V to have non-negative drift everywhere, 4.1-4.7 being simultaneously non-negative is a sufficient condition. These seemingly numerous conditions can be significantly reduced. Note that equations 4.4 and 4.7 are the same as are 4.5 and 4.6. In addition, 4.1 is a lower bound on both 4.4 and 4.5. Then the set sufficient conditions λ, μ and μ' must satisfy for V to have non-negative drift everywhere are

$$\begin{aligned}
2\lambda - \mu - \mu' + M[(2C+1)(-\mu - \mu')] &\geq 0, \\
2\lambda - 2\mu + M[2\lambda + 2\mu] &\geq 0, \\
2\lambda - 2\mu' + M[2\lambda + 2\mu'] &\geq 0,
\end{aligned}$$

which imply that there must exist $M > 0$ with

$$\max\left\{\frac{\mu' - \lambda}{\mu' + \lambda}, \frac{\mu - \lambda}{\mu + \lambda}\right\} \leq M < \frac{2\lambda - \mu - \mu'}{(2C+1)(\mu + \mu')}.$$

Note that as C gets larger, the parameters λ^*, μ^*, μ' can be chosen accordingly large. Let $C = 50$, $\lambda = 525.0841$, $\mu = 527.3004$, $\mu' = 249.5696$. Then we have $0.0021 \leq M \leq 0.0036$. □

This type of a scheduling policy can be considered as an extension of LQF, which can be interpreted as the aforementioned policy with $C = 0$. Here, the tie breaking set of LQF ($q_1 = q_2$) is enlarged to include a bigger set between $q_2 = q_1 + C$ and $q_2 = q_1 - C$. Then we can infer that this simple extension is not sufficient in moving from LQF to maximal stability.

Similarly, we can compare this policy to MaxWeight. Suppose the service configuration that results in stability according to MaxWeight is used in the free corridor. Proof of Proposition 5 still suggests that the policy is not maximally stable. Then the slopes of the two switching lines, μ_{11}/μ_{21} and μ_{12}/μ_{22}, are essential for maximal stability, and the switching lines cannot be simply replaced by $q_1 \pm C$ to simplify the policy while ensuring maximal stability.

4.2 Connection between the Static Planning Problem and the Transportation Problems

In this section, we present the connection between SPP($\mathcal{F}$) (introduced in Section 3.3) and the generalized transportation problem. This connection allows us to consider some special cases of parallel server systems in more depth. In Section 4.2.1, we demonstrate how our design algorithm can be used in so-called product form systems to obtain exactly $m+n$ arcs in $\mathcal{S}$. In Section 4.2.2, using the connection between SPP($\mathcal{F}$) and the generalized transportation

problem, we establish a new characterization of the uniqueness of the solutions to SPP($\mathcal{F}$). This characterization also acts as an intuitive construction principle which we use to build systems for our numerical experiments in Section 3.5.

Transportation problems are a special case of network flow problems (Ahuja et al. 1988), where the goal is to find a feasible flow on an arbitrary graph to usually minimize the total cost of carrying the flow, or to maximize the total flow received at a sink node. Usually, transportation problems are concerned with finding a minimum cost flow to carry supply from m supply nodes to n demand nodes on a bipartite graph (for a detailed treatment of the topic, see Chapter 9 of Hadley (1961)).

The most well-known version of the transportation problem is called the pure transportation problem, where one unit of flow sent from a supply node j on arc (i, j) arrives at the destination i unchanged, as one unit of flow. That is, in a pure transportation problem, flow is conserved on each arc. In contrast, in the *generalized* transportation problem, one unit of flow sent from a supply node j on arc (i, j) arrives at the destination i as μ_{ij} units of flow. These types of networks are also called gainy or lossy networks, as flow is not conserved. If $\mu_{ij} = 1$ for all $(i, j) \in \mathcal{F}$, the generalized transportation problem is reduced to the pure transportation problem. The same definitions extend to more general network flow problems, if flow is conserved on each arc, the problem is called a pure network flow problem, otherwise the problem is called a generalized network flow problem.

Let $f_{ij} \geq 0$ be the total flow carried on arc $(i, j) \in \mathcal{F}$, $h_{ij} \geq 0$ denote the cost of carrying one unit of flow on arc $(i, j) \in \mathcal{F}$, $D_i \geq 0$ be the total demand of node $i \in \mathcal{I}$, and $C_j \geq 0$ denote the total capacity of supply node $j \in \mathcal{J}$. Then the pure transportation problem

(PTP) can be written as

$$\begin{aligned} \text{(PTP)} \qquad \min_{\mathbf{f}} \quad & \sum_{(i,j)\in\mathcal{F}} h_{ij} f_{ij} \\ s.t. \quad & \sum_{j\in\mathcal{J}} f_{ij} = D_i \quad \forall i \in \mathcal{I} \\ & \sum_{i\in\mathcal{I}} f_{ij} = C_j \quad \forall j \in \mathcal{J} \\ & \mathbf{f} \geq \mathbf{0} \end{aligned}$$

Similarly, when flow conservation assumption is relaxed and one unit of flow from server j arrives as μ_{ij} units in queue i, the generalized transportation problem (GTP) can be written as

$$\begin{aligned} \text{(GTP)} \qquad \min_{\mathbf{f}} \quad & \sum_{(i,j)\in\mathcal{F}} h_{ij} f_{ij} \\ s.t. \quad & \sum_{j\in\mathcal{J}} \mu_{ij} f_{ij} = D_i \quad \forall i \in \mathcal{I} \\ & \sum_{i\in\mathcal{I}} f_{ij} = C_j \quad \forall j \in \mathcal{J} \\ & \mathbf{f} \geq \mathbf{0} \end{aligned}$$

A well known necessary and sufficient condition for the feasibility of the pure transportation problem is

$$\sum_{i\in\mathcal{I}} D_i = \sum_{j\in\mathcal{J}} C_j.$$

This condition is not well defined for the generalized transportation problem. In general, there is no equivalent simple condition for the feasibility of the generalized transportation

problem (Hadley 1961).

SPP($\mathcal{F}$) can be interpreted in the framework of transportation problems. Each of the m servers in a flexible server system correspond to a supply node in a transportation problem, and each of the n queues in the flexible server system correspond to a demand node in the transportation problem. Under Assumption **A2.** in Section 3.3, each server (supply node) has a total supply of 1 available. Total supply of 1 can be interpreted as the total fraction of time a server has available. Then x_{ij} is the supply sent on arc (i, j), or the average fraction of time spent on activity (i, j). Similarly, each queue i (demand node i) has a total demand of λ_i.

In SPP($\mathcal{F}$), under Assumption **A2.**, we assume our optimal objective value is fixed at one. Then our goal is equivalent to finding a feasible flow in the analogous transportation problem. With this perception, we see that the feasible region of SPP($\mathcal{F}$) is very similar to the feasible region of the generalized transportation problem.

In the context of our problem in Chapter 3, when the service rates are server dependent (that is, $\mu_{ij} = \mu_j$ for all $j \in \mathcal{J}$), SPP($\mathcal{F}$) can be scaled such that the constraint set is equivalent to the constraint set of a pure transportation problem. Let $f_{ij} := \mu_j x_{ij}$ for all $(i, j) \in \mathcal{F}$, and consider the constraints of SPP($\mathcal{F}$) under Assumption **A2.**.

$$\sum_{j \in \mathcal{J}} \mu_j x_{ij} = \lambda_i \quad \forall i \in \mathcal{I}, \tag{4.8}$$

$$\sum_{i \in \mathcal{I}} x_{ij} = 1 \quad \forall j \in \mathcal{J}. \tag{4.9}$$

Applying the transformation $f_{ij} = \mu_j x_{ij}$,

$$\sum_{j \in \mathcal{J}} \mu_j x_{ij} = \sum_{j \in \mathcal{J}} f_{ij} = \lambda_i \quad \forall i \in \mathcal{I},$$

$$\sum_{i \in \mathcal{I}} x_{ij} = \sum_{i \in \mathcal{I}} \frac{f_{ij}}{\mu_j} = 1 \quad \forall j \in \mathcal{J}.$$

Equivalently,

$$\sum_{j\in\mathcal{J}} f_{ij} = \lambda_i \quad \forall i \in \mathcal{I}, \tag{4.10}$$

$$\sum_{i\in\mathcal{I}} f_{ij} = \mu_j \quad \forall j \in \mathcal{J}. \tag{4.11}$$

Indeed, 4.10-4.11 are the constraints of the pure transportation problem. Then one could find a feasible solution the pure transportation problem and obtain $\mathbf{f}$, then reverse the transformation and obtain an optimal solution $x^*_{ij} = f_{ij}/\mu_j$ to SPP($\mathcal{F}$) under Assumption **A2.**.

This procedure essentially reduces the generalized transportation problem to the pure transportation problem. Such a reduction is not always possible. The necessary and sufficient conditions for the reducibility of a generalized network problem (not necessarily a transportation problem) to a pure network problem were given by Truemper (1976). Theorem 4 states this result in less generality than given in the original paper.

Theorem 4. *(Theorem 1, Truemper (1976)) The following are equivalent:*

1. *A generalized transportation problem on graph $\mathcal{S}$ can be reduced to a pure transportation problem on $\mathcal{S}$.*

2. *There exists $\alpha_i > 0$, $\beta_j > 0$ such that $\mu_{ij} = \alpha_i\beta_j$ for all $(i,j) \in \mathcal{S}$.*

3. $$rank\left(\begin{bmatrix} R_{\mathcal{S}} \\ A_{\mathcal{S}} \end{bmatrix}\right) = m + n - 1.$$

If there exist α_i, β_j such that $\mu_{ij} = \alpha_i\beta_j$, μ_{ij} are called product form. Such special systems are also practically interesting, as service can be thought as having a server dependent component and a queue dependent component. Similar to the reduction of the

server dependent case to the pure transportation problem, we can apply the transformation $\tilde{\lambda}_i = \lambda_i/\alpha_i$ and $f_{ij} = x_{ij}\beta_j$ and reduce feasible region of the product form SPP($\mathcal{F}$) to a pure transportation polytope:

$$\sum_{j\in\mathcal{J}} \mu_{ij}x_{ij} = \lambda_i \iff \sum_{j\in\mathcal{J}} \alpha_i\beta_j x_{ij} = \lambda_i$$
$$\iff \sum_{j\in\mathcal{J}} \beta_j x_{ij} = \frac{\lambda_i}{\alpha_i} \iff \sum_{j\in\mathcal{J}} f_{ij} = \tilde{\lambda}_i \qquad \forall i \in \mathcal{I},$$

and

$$\sum_{i\in\mathcal{I}} x_{ij} = 1 \iff \sum_{i\in\mathcal{I}} \frac{f_{ij}}{\beta_j} = 1$$
$$\iff \sum_{i\in\mathcal{I}} f_{ij} = \beta_j \qquad \forall j \in \mathcal{J}.$$

It is important to note that a server dependent system is essentially a product form system with $\alpha_i = 1$ for all $i \in \mathcal{I}$. Similarly, a product form system can be interpreted as a server dependent system with a simple scaling of the demand, $\tilde{\lambda}_i = \lambda_i/\alpha_i$. Thus, in the forthcoming discussion we do not distinguish between product form and server dependent systems. In any case, since these systems are reducible to pure transportation problems where flow conservation holds, all existing methods in the literature addressing the design problem can be used to obtain a sparse design $\mathcal{S}$.

4.2.1 Application of Algorithm 2 in Product Form Systems

When μ_{ij} are product form, our design Algorithm 2 can be used to obtain an efficient design with exactly $m + n$ arcs, similar to Shi et al. (2019). However, unlike their work, our design can only guarantee that the GCG is strictly positive. Our design does not necessarily result in a flexibility graph with a GCG lower bounded away from zero. Below, we present the

application of our algorithm to product form systems: we first establish that the support graph is $\mathcal{F}$, then choose a particular way of connecting the disconnected components in an optimal degenerate BFS of SPP($\mathcal{F}$).

Lemma 9. *If the system is product-form, then $\mathcal{G} = \mathcal{F}$.*

Proof. Support graph $\mathcal{G}$ is the subgraph of $\mathcal{F}$ where arcs that are never used in any solution are removed. Then, to show that $\mathcal{G} = \mathcal{F}$, it is sufficient to construct a solution that uses the entire graph $\mathcal{F}$. Algorithm 3 describes a procedure to construct such a solution.

Algorithm 3 starts by scaling SPP($\mathcal{F}$) appropriately and uses the equivalent pure transportation problem to construct a solution on $\mathcal{F}$. We recall that

$$\sum_{i \in \mathcal{I}} \tilde{\lambda}_i = \sum_{j \in \mathcal{J}} \beta_j$$

is satisfied because the pure transportation problem is feasible. We claim that at each iteration of Algorithm 3,

$$\sum_{i \in \mathcal{I}'} \tilde{\lambda}'_i = \sum_{j \in \mathcal{J}'} \beta'_j \tag{4.12}$$

is maintained. We will prove this with induction.

- Base case: $k = 1$.
  - If ξ_k is a demand node, we have

$$\beta'_j = \beta_j - \frac{\tilde{\lambda}_{\xi_k}}{m}$$
$$\tilde{\lambda}'_{\xi_k} = 0.$$

Then,

$$\begin{aligned}\sum_{j\in\mathcal{J}'}\beta'_j &= \sum_{j\in\mathcal{J}}\left(\beta_j - \frac{\tilde{\lambda}_{\xi_k}}{m}\right)\\ &= \sum_{j\in\mathcal{J}}\beta_j - \tilde{\lambda}_{\xi_k}\\ &= \sum_{i\in\mathcal{I}}\tilde{\lambda}_i - \tilde{\lambda}_{\xi_k}\\ &= \sum_{i\in\mathcal{I}'}\tilde{\lambda}_i\\ &= \sum_{i\in\mathcal{I}'}\tilde{\lambda}'_i.\end{aligned}$$

– If ξ_k is a supply node, we have

$$\begin{aligned}\tilde{\lambda}'_i &= \tilde{\lambda}_i - \frac{\beta_{\xi_k}}{n}\\ \beta'_{\xi_k} &= 0.\end{aligned}$$

Then,

$$\begin{aligned}\sum_{i\in\mathcal{I}'}\tilde{\lambda}'_i &= \sum_{i\in\mathcal{I}}\left(\tilde{\lambda}_i - \frac{\beta_{\xi_k}}{n}\right)\\ &= \sum_{i\in\mathcal{I}}\tilde{\lambda}_i - \beta_{\xi_k}\\ &= \sum_{j\in\mathcal{J}}\beta_j - \beta_{\xi_k}\\ &= \sum_{j\in\mathcal{J}'}\beta_j\\ &= \sum_{j\in\mathcal{J}'}\beta'_j.\end{aligned}$$

- Induction hypothesis: for iteration k,

$$\sum_{i \in \mathcal{I}'} \tilde{\lambda}'_i = \sum_{j \in \mathcal{J}'} \beta'_j,$$

  where $|\mathcal{I}'| = n - k_q$ and $|\mathcal{J}'| = m - k_s$.

- Next case: $k + 1$.

  - If ξ_{k+1} is a demand node, we have

$$\beta''_j = \beta'_j - \frac{\tilde{\lambda}'_{\xi_k}}{m - k_s}$$
$$\tilde{\lambda}''_{\xi_k} = 0,$$

    where β''_j and $\tilde{\lambda}''_i$ are end-of-iteration remaining supplies and demands. Then,

$$\begin{aligned}
\sum_{j \in \mathcal{J}''} \beta''_j &= \sum_{j \in \mathcal{J}'} \left(\beta'_j - \frac{\tilde{\lambda}'_{\xi_k}}{m - k_s} \right) \\
&= \sum_{j \in \mathcal{J}'} \beta'_j - \tilde{\lambda}'_{\xi_k} \\
&= \sum_{i \in \mathcal{I}'} \tilde{\lambda}'_i - \tilde{\lambda}'_{\xi_k} \\
&= \sum_{i \in \mathcal{I}''} \tilde{\lambda}'_i \\
&= \sum_{i \in \mathcal{I}''} \tilde{\lambda}''_i.
\end{aligned}$$

  - If ξ_{k+1} is a supply node, we have

$$\tilde{\lambda}''_i = \tilde{\lambda}'_i - \frac{\beta'_{\xi_k}}{n - k_q}$$
$$\beta''_{\xi_k} = 0.$$

where β''_j and $\tilde{\lambda}''_i$ are end-of-iteration remaining supplies and demands. Then,

$$\begin{aligned}
\sum_{i\in\mathcal{I}''} \tilde{\lambda}''_i &= \sum_{i\in\mathcal{I}'} \left(\tilde{\lambda}'_i - \frac{\beta'_{\xi_k}}{n-k_q} \right) \\
&= \sum_{i\in\mathcal{I}'} \tilde{\lambda}'_i - \beta'_{\xi_k} \\
&= \sum_{j\in\mathcal{J}'} \beta'_j - \beta'_{\xi_k} \\
&= \sum_{j\in\mathcal{J}''} \beta'_j \\
&= \sum_{j\in\mathcal{J}''} \beta''_j.
\end{aligned}$$

Thus, we conclude that 4.12 is maintained at each iteration of Algorithm 3.

At each iteration of Algorithm 3, if ξ_k is a demand node, that node pulls δ_k amount of flow from every remaining server. Unless there is a single demand node remaining (i.e., unless $n - k_q = 1$),

$$\beta'_j - \delta_k = \beta'_j - \frac{\tilde{\lambda}'_{\xi_k}}{m-k_s} \geq \beta'_j - \frac{\beta'_j}{n-k_q} > 0$$

by construction. Then we only deplete the remaining demand of node ξ_k without fully depleting any of the remaining supply nodes. This implies that we use and then remove exactly $m-k_s$ arcs from the network at this iteration. If a single demand node is remaining, that demand node pulls required supply from all of the remaining supply nodes, and because 4.12 is maintained at each iteration, no supply remains in the network and we end up with a solution that uses all mn arcs.

If ξ_k is a server node, the dynamics are very similar. Therefore, Algorithm 3 returns a solution to SPP($\mathcal{F}$) using the entire $\mathcal{F}$. □

Starting with a degenerate basis B, we want to apply Algorithm 2 to obtain a sparse $\mathcal{S}$ that achieves a single resource pool. Relying on Theorem 3, we want to connect components of B. Because $\mathcal{G} = \mathcal{F}$, we have a lot of flexibility in choosing the incoming and outgoing arcs from each component when the system is product form. Then we can connect components by creating a long chain-like graph. Since all the chosen arcs exist in $\mathcal{G}$, the resulting graph is a feasible sparse $\mathcal{S}$ that contains exactly $m + n$ arcs.

Corollary 3. *If the system is product form, there exists a sparse design $\mathcal{S}$ with $m + n$ arcs such that DSPP($\mathcal{S}$) has a unique solution.*

Proof. Let B be a degenerate basis to SPP($\mathcal{F}$) with τ trees. Consider Algorithm 2. Because $\mathcal{G} = \mathcal{F}$, every tree can be connected to every other tree. Without loss of generality, we can order the trees $C_1, \ldots, C_\tau$. In the execution of Algorithm 2, we start by picking the first tree C_1. Subsequently, the outgoing arcs can be chosen such that C_1 has an outgoing arc to C_2, which has an outgoing arc to C_3. Then C_3 can have an outgoing arc to C_4, and continuing like this, C_τ can have an outgoing arc to C_1.

Therefore, we can terminate Algorithm 2 in a single iteration. Because we have connected τ components using τ arcs, we end up with $|\mathcal{C}| = 1$ at the end of this iteration, so that

$$|\mathcal{S}| = m + n - \tau + \tau = m + n.$$

□

4.2.2 Uniqueness of Solutions in SPP($\mathcal{F}$)

As we have established so far, systems with flow conservation have desirable properties in the context of the design problem. However, such systems do not satisfy the traditional heavy-traffic assumptions that much of the performance analysis literature relies on. In particular,

systems with flow conservation do not satisfy the assumption that SPP($\mathcal{F}$) has a unique solution. Indeed, Atar et al. (2022) had shown that if $\mu_{ij} = \alpha_i \beta_j$ for all $(i, j) \in \mathcal{F}$, SPP($\mathcal{F}$) has multiple solutions. An application of Theorem 4 enables us to prove a partial converse to that statement, obtaining a simple characterization of uniqueness in SPP($\mathcal{F}$).

Theorem 5. *If SPP($\mathcal{F}$) has multiple solutions, μ_{ij} are partially product form.*

Proof. If a bounded feasible linear program has multiple solutions, then at least two of them are adjacent basic feasible solutions (Property 1b, p. 195, Hillier and Lieberman (2001)).

Let $(\mathbf{x}^*_B, 1)$ and $(\mathbf{x}^*_{B'}, 1)$ be adjacent basic feasible solutions to SPP($\mathcal{F}$). Then, $\mathbf{x}^*_B$ and $\mathbf{x}^*_{B'}$ share $m + n - 2$ components. Because they are basic feasible solutions, they uniquely solve

$$\begin{bmatrix} R_B \\ A_B \end{bmatrix} \mathbf{x}_B = \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{1} \end{bmatrix},$$

and

$$\begin{bmatrix} R_{B'} \\ A_{B'} \end{bmatrix} \mathbf{x}_{B'} = \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{1} \end{bmatrix},$$

respectively. Then the coefficient matrices,

$$\begin{bmatrix} R_B \\ A_B \end{bmatrix}$$

and

$$\begin{bmatrix} R_{B'} \\ A_{B'} \end{bmatrix}$$

each have $m+n-1$ linearly independent columns.

Let B_C and B'_C denote the set of common components of bases B and B'. Note that $B_C = B'_C$ and they have $m+n-2$ elements. Let B_N and B'_N denote the distinct components of the bases, and note that $B_N \neq B'_N$ and they are singletons. Then x_B and $x_{B'}$ uniquely solve

$$\begin{bmatrix} R_{B_C} & R_{B_N} \\ A_{B_C} & A_{B_N} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{B_C} \\ x_{B_N} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\lambda} \\ \mathbf{1} \end{bmatrix},$$

and

$$\begin{bmatrix} R_{B'_C} & R_{B'_N} \\ A_{B'_C} & A_{B'_N} \end{bmatrix} \begin{bmatrix} x_{B'_C} \\ \mathbf{x}_{B'_N} \end{bmatrix} = \begin{bmatrix} \lambda \\ 1 \end{bmatrix},$$

respectively. This implies that

$$\begin{bmatrix} R_{B_C} & R_{B_N} & R_{B'_N} \\ A_{B_C} & A_{B_N} & A_{B'_N} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{B_C} - \mathbf{x}_{B'_C} \\ \mathbf{x}_{B_N} \\ -\mathbf{x}_{B'_N} \end{bmatrix} = \mathbf{0}. \tag{4.13}$$

This system has a nontrivial solution because $\mathbf{x}_B \neq \mathbf{x}_{B'}$. Since the homogeneous system in $m+n$ equations and $m+n$ unknowns has a nontrivial solution, it must be that the determinant of its coefficient matrix is zero (Corollary 2.13, p.37, Shafarevich and Remizov (2012)). Then the coefficient matrix must be rank deficient. Because the coefficient matrix has at least $m+n-1$ linearly independent columns, being rank deficient implies that its rank is exactly $m+n-1$. Since the coefficient matrix corresponds to that of a generalized network, by Theorem 4, μ_{ij} associated with $B \cup B'$ must be product form. □

Theorem 5 implies that if in $\mathcal{F}$, fewer than $m+n$ of μ_{ij} are product form, then SPP($\mathcal{F}$)

has a unique solution. Additionally, it implies that when SPP($\mathcal{F}$) has multiple solutions, at least $m + n$ of the μ_{ij} must be product form. This insight allows us to construct partially product form systems with multiple solutions to SPP($\mathcal{F}$) and a unique solution to DSPP($\mathcal{F}$), which we use as a construction principle in Section 3.5 where we numerically compare the performance of our proposed design $\mathcal{S}$ to that of the full flexibility structure $\mathcal{F}$.

Theorem 5 uncovers a new part of the involved connection between the geometry of the static planning problem and the system parameters. Discovering deeper details of this relationship may allow researchers to propose flexibility designs with even fewer connections that can achieve the benefits of the full flexibility structure.

Algorithm 3: Solution on $\mathcal{F}$ in product form systems

Input : Product form SPP($\mathcal{F}$)

Apply the appropriate scaling to obtain the equivalent pure transportation problem with demand $\tilde{\lambda}_i$ for all $i \in \mathcal{I}$ and supply β_j for all $j \in \mathcal{J}$.

Initialize remaining demands and supplies: $\tilde{\lambda}'_i = \tilde{\lambda}_i$ for all $i \in \mathcal{I}$ and $\beta'_j = \beta_j$ for all $j \in \mathcal{J}$.

Initialize remaining node sets $\mathcal{I}' = \mathcal{I}$, $\mathcal{J}' = \mathcal{J}$.

Initialize counters: $k_q = 0$, $k_s = 0$.

Set flows: $f_{ij} = 0$ for all $(i, j) \in \mathcal{F}$.

while $k_q + k_s \leq m + n - 1$ **do**

  Set iteration counter $k = k_q + k_s + 1$.

  Calculate

$$\delta_k = \min_{i \in \mathcal{I}', j \in \mathcal{J}'} \left\{ \frac{\tilde{\lambda}'_i}{m - k_s}, \frac{\beta'_j}{n - k_q} \right\},$$

$$\xi_k = \arg \min_{i \in \mathcal{I}', j \in \mathcal{J}'} \left\{ \frac{\tilde{\lambda}'_i}{m - k_s}, \frac{\beta'_j}{n - k_q} \right\}.$$

  if $\xi_k \in \mathcal{I}'$ **then**

    Update $f_{\xi_k j} = f_{\xi_k j} + \delta_k$ for all $j \in \mathcal{J}'$.

    Update $\tilde{\lambda}'_{\xi_k} = 0$ and $\beta'_j = \beta'_j - \delta_k$ for all $j \in \mathcal{J}'$.

    $k_q = k_q + 1$, $\mathcal{I}' = \mathcal{I}' \setminus \{\xi_k\}$.

  else

    Update $f_{i \xi_k} = f_{i \xi_k} + \delta_k$ for all $i \in \mathcal{I}'$.

    Update $\beta'_{\xi_k} = 0$ and $\tilde{\lambda}'_i = \tilde{\lambda}'_i - \delta_k$ for all $i \in \mathcal{I}'$.

    $k_s = k_s + 1$, $\mathcal{J}' = \mathcal{J}' \setminus \{\xi_k\}$.

  end if

end while

Return: f_{ij}, $(i, j) \in \mathcal{F}$.

REFERENCES

Saghar Adler, Mehrdad Moharrami, and Vijay Subramanian. Learning a discrete set of optimal allocation rules in queueing systems with unknown service rates. *arXiv preprint arXiv:2202.02419*, 2022.

Ravindra K Ahuja, Thomas L Magnanti, and James B Orlin. Network flows. 1988.

Matthew Andrews and Lisa Zhang. Achieving stability in networks of input-queued switches. *IEEE/ACM Transactions on networking*, 11(5):848–857, 2003.

Gautam Appa. On the uniqueness of solutions to linear programs. *Journal of the Operational Research Society*, 53:1127–1132, 2002.

Arash Asadpour, Xuan Wang, and Jiawei Zhang. Online resource allocation with limited flexibility. *Management Science*, 66(2):642–666, 2020.

Baris Ata and Sunil Kumar. Heavy traffic analysis of open processing networks with complete resource pooling: Asymptotic optimality of discrete review policies. 2005.

Rami Atar, Eyal Castiel, and Martin I Reiman. Parallel server systems under an extended heavy traffic condition: A lower bound. *arXiv preprint arXiv:2201.07855*, 2022.

Golshid Baharian and Tolga Tezcan. Stability analysis of parallel server systems under longest queue first. *Mathematical Methods of Operations Research*, 74(2):257, 2011.

Steven Bell and Ruth Williams. Dynamic scheduling of a parallel server system in heavy traffic with complete resource pooling: Asymptotic optimality of a threshold policy. 2005.

Dimitris Bertsimas and John N Tsitsiklis. *Introduction to linear optimization*, volume 6. Athena scientific Belmont, MA, 1997.

Maury Bramson, Bernardo D'Auria, and Neil Walton. Stability and instability of the maxweight policy. *Mathematics of Operations Research*, 46(4):1611–1638, 2021.

Maury Bramson et al. Instability of fifo queueing networks. *The Annals of Applied Probability*, 4 (2):414–431, 1994.

Xi Chen, Jiawei Zhang, and Yuan Zhou. Optimal sparse designs for process flexibility via probabilistic expanders. *Operations Research*, 63(5):1159–1176, 2015.

Xi Chen, Tengyu Ma, Jiawei Zhang, and Yuan Zhou. Optimal design of process flexibility for general production systems. *Operations Research*, 67(2):516–531, 2019.

Mabel C Chou, Geoffrey A Chua, Chung-Piaw Teo, and Huan Zheng. Design for process flexibility: Efficiency of the long chain and sparse structure. *Operations research*, 58(1):43–58, 2010.

Mabel C Chou, Geoffrey A Chua, Chung-Piaw Teo, and Huan Zheng. Process flexibility revisited: The graph expander and its applications. *Operations research*, 59(5):1090–1105, 2011.

Mabel C Chou, Geoffrey A Chua, and Huan Zheng. On the performance of sparse process structures in partial postponement production systems. *Operations research*, 62(2):348–365, 2014.

Tuhinangshu Choudhury, Gauri Joshi, Weina Wang, and Sanjay Shakkottai. Job dispatching policies for queueing systems with unknown service rates. In *Proceedings of the Twenty-second International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, pages 181–190, 2021.

Jim G Dai. On positive harris recurrence of multiclass queueing networks: a unified approach via fluid limit models. *The Annals of Applied Probability*, pages 49–77, 1995.

Jim G Dai and Wuqin Lin. Maximum pressure policies in stochastic processing networks. *Operations Research*, 53(2):197–218, 2005.

Jim G Dai and Gideon Weiss. Stability and instability of fluid models for reentrant lines. *Mathematics of Operations Research*, 21(1):115–134, 1996.

Antoine Désir, Vineet Goyal, Yehua Wei, and Jiawei Zhang. Sparse process flexibility designs: Is the long chain really optimal? *Operations Research*, 64(2):416–431, 2016.

Antonis Dimakis and Jean Walrand. Sufficient conditions for stability of longest-queue-first scheduling: Second-order properties using fluid limits. *Advances in Applied probability*, 38(2):505–521, 2006.

Atilla Eryilmaz and Rayadurgam Srikant. Asymptotically tight steady-state queue length bounds implied by drift conditions. *Queueing Systems*, 72:311–359, 2012.

Vivek F Farias, Ciamac C Moallemi, and Balaji Prabhakar. Load balancing with migration penalties. In *Proceedings. International Symposium on Information Theory, 2005. ISIT 2005.*, pages 558–562. IEEE, 2005.

David Gamarnik and Dmitriy Katz. On deciding stability of multiclass queueing networks under buffer priority scheduling policies. *The Annals of Applied Probability*, pages 2008–2037, 2009.

Itay Gurvich and Ward Whitt. Scheduling flexible servers with convex delay costs in many-server service systems. *Manufacturing & Service Operations Management*, 11(2):237–253, 2009.

George Hadley. *Linear programming*. Narosa publishing house, 1961.

J Michael Harrison. Heavy traffic analysis of a system with parallel servers: asymptotic optimality of discrete-review policies. *Annals of applied probability*, pages 822–848, 1998.

J Michael Harrison and Marcel J López. Heavy traffic resource pooling in parallel-server systems. *Queueing systems*, 33:339–368, 1999.

Frederick S Hillier and Gerald J Lieberman. Introduction to operations research, 2001.

Wei-Kang Hsu, Jiaming Xu, Xiaojun Lin, and Mark R Bell. Integrate learning and control in queueing systems with uncertain payoffs. *Purdue University, available at https://engineering. purdue. edu/% 7elinx/papers. html, Tech. Rep*, 2018.

Seyed MR Iravani, Bora Kolfal, and Mark P Van Oyen. Call-center labor cross-training: It'sa small world after all. *Management Science*, 53(7):1102–1112, 2007.

William C Jordan and Stephen C Graves. Principles on the benefits of manufacturing process flexibility. *Management science*, 41(4):577–594, 1995.

Srikanth Kandula, Sudipta Sengupta, Albert Greenberg, Parveen Patel, and Ronnie Chaiken. The nature of data center traffic: measurements & analysis. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement*, pages 202–208, 2009.

Subhashini Krishnasamy, Rajat Sen, Ramesh Johari, and Sanjay Shakkottai. Learning unknown service rates in queues: A multiarmed bandit approach. *Operations Research*, 69(1):315–330, 2021.

Avishai Mandelbaum and Alexander L Stolyar. Scheduling flexible servers with convex delay costs: Heavy-traffic optimality of the generalized cμ-rule. *Operations Research*, 52(6):836–855, 2004.

Sean P Meyn and Richard L Tweedie. *Markov chains and stochastic stability*. Springer Science & Business Media, 2012.

Sean P Meyn et al. Transience of multiclass queueing networks via fluid limit models. *The Annals of Applied Probability*, 5(4):946–957, 1995.

Ramtin Pedarsani and Jean Walrand. Stability of multiclass queueing networks under longest-queue and longest-dominating-queue scheduling. *Journal of Applied Probability*, 53(2):421–433, 2016.

Ramtin Pedarsani, Jean Walrand, and Yuan Zhong. Robust scheduling for flexible processing networks. *Advances in Applied Probability*, 49(2):603–628, 2017.

Igor R Shafarevich and Alexey O Remizov. *Linear algebra and geometry*. Springer Science & Business Media, 2012.

Cong Shi, Yehua Wei, and Yuan Zhong. Process flexibility for multiperiod production systems. *Operations Research*, 67(5):1300–1320, 2019.

Gerard Sierksma and Yori Zwols. *Linear and integer optimization: theory and practice*. CRC Press, 2015.

David Simchi-Levi and Yehua Wei. Understanding the performance of the long chain and sparse designs in process flexibility. *Operations research*, 60(5):1125–1141, 2012.

David Simchi-Levi and Yehua Wei. Worst-case analysis of process flexibility designs. *Operations Research*, 63(1):166–185, 2015.

David Simchi-Levi, He Wang, and Yehua Wei. Increasing supply chain robustness through process flexibility and inventory. *Production and Operations Management*, 27(8):1476–1491, 2018.

Thomas Stahlbuhk, Brooke Shrader, and Eytan Modiano. Learning algorithms for minimizing queue length regret. *IEEE Transactions on Information Theory*, 67(3):1759–1781, 2021.

Alexander L Stolyar et al. Maxweight scheduling in a generalized switch: State space collapse and workload minimization in heavy traffic. *Annals of Applied Probability*, 14(1):1–53, 2004.

Leandros Tassiulas and Anthony Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. In *29th IEEE Conference on Decision and Control*, pages 2130–2132. IEEE, 1990.

Tolga Tezcan. Instability of fifo in a simple queueing system with arbitrarily low loads. *Operations research letters*, 37(5):312–316, 2009.

Tolga Tezcan. Stability analysis of n-model systems under a static priority rule. *Queueing Systems*, 73(3):235–259, 2013.

Gert A Tijssen and Gerard Sierksma. Balinski—tucker simplex tableaus: Dimensions, degeneracy degrees, and interior points of optimal faces. *Mathematical programming*, 81:349–372, 1998.

Klaus Truemper. An efficient scaling procedure for gain networks. *Networks*, 6(2):151–159, 1976.

John N Tsitsiklis and Kuang Xu. On the power of (even a little) resource pooling. *Stochastic Systems*, 2(1):1–66, 2013.

John N Tsitsiklis and Kuang Xu. Flexible queueing architectures. *Operations Research*, 65(5): 1398–1413, 2017.

Sushil Mahavir Varma and Siva Theja Maguluri. Transportation polytope and its applications in parallel server systems. *arXiv preprint arXiv:2108.13167*, 2021.

Shixin Wang, Xuan Wang, and Jiawei Zhang. A review of flexible processes and operations. *Production and Operations Management*, 30(6):1804–1824, 2021.

Shixin Wang, Xuan Wang, and Jiawei Zhang. Robust optimization approach to process flexibility designs with contribution margin differentials. *Manufacturing & Service Operations Management*, 24(1):632–646, 2022.

Xuan Wang and Jiawei Zhang. Process flexibility: A distribution-free bound on the performance of k-chain. *Operations Research*, 63(3):555–571, 2015.

Zhen Xu, Hailun Zhang, Jiheng Zhang, and Rachel Q Zhang. Online demand fulfillment under limited flexibility. *Management Science*, 66(10):4667–4685, 2020.

Yueyang Zhong, John R Birge, and Amy Ward. Learning the scheduling policy in time-varying multiclass many server queues with abandonment. *Available at SSRN*, 2022.