The Dissertation Committee for Emre Arda Sisbot
certifies that this is the approved version of the following dissertation:

Fluid and Queueing Networks with Gurvich-type Routing

Committee:

John J. Hasenbein, Supervisor

J. Eric Bickel

Milica Cudina

Dragan Djurdjanovic

Aida Khajavirad

Fluid and Queueing Networks with Gurvich-type Routing

by

Emre Arda Sisbot, B.S., M.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2015

Dedicated to my family.

Acknowledgments

First and foremost, I would like to thank my supervisor, Dr. Hasenbein. I am deeply grateful to him, not only for his guidance but also for his patience and kindness over the last few years. He showed me how to have an eye for detail while keeping the big picture in mind. I also want to thank my committee members, Professors J. Eric Bickel, Milica Cudina, Dragan Djurdjanovic and Aida Khajavirad for their insightful suggestions.

I have been fortunate to have made many good friends in Austin. Special thanks are due to Tao, for not only being the greatest office-mate but also for the constant stream of interesting conversations. I am grateful to the Gomez-Contreras clan for making me feel like a part of their family. My roommate for years, Gokhan, deserves special thanks for his support on every bump along the way. Finally, I would like to thank my many friends back home, especially Sedef, for their genuine support regardless of the distance.

Last but not least, I would like to thank my family for their endless love and encouragement: my father, Sedat, for motivating me through out this journey, my mom, Nuket, for her unconditional love and my brother, Akin, for always being a great example to follow.

To those I have mentioned and the countless others who have supported and helped me along the way, I offer a heartfelt thank you.

Fluid and Queueing Networks with Gurvich-type Routing

Publication No. ___________________

Emre Arda Sisbot, Ph.D.
The University of Texas at Austin, 2015

Supervisor: John J. Hasenbein

Queueing networks have applications in a wide range of domains, from call center management to telecommunication networks. Motivated by a healthcare application, in this dissertation, we analyze a class of queueing and fluid networks with an additional routing option that we call Gurvich-type routing. The networks we consider include parallel buffers, each associated with a different class of entity, and Gurvich-type routing allows to control the assignment of an incoming entity to one of the classes. In addition to routing, scheduling of entities is also controlled as the classes of entities compete for service at the same station. A major theme in this work is the investigation of the interplay of this routing option with the scheduling decisions in networks with various topologies.

The first part of this work focuses on a queueing network composed of two parallel buffers. We form a Markov decision process representation of this system and prove structural results on the optimal routing and scheduling controls. Via these results, we determine a near-optimal discrete policy by solving the associated fluid model along with perturbation expansions. In the second part, we analyze a single-station fluid network composed of N parallel buffers with an arbitrary N. For this network, along with structural proofs on the optimal scheduling policies, we show that the optimal routing policies are threshold-based. We then develop a numerical procedure to compute the optimal policy for any initial state. The final part of this work extends the analysis of the previous part to tandem fluid networks composed of two stations. For two different models, we provide results on the optimal scheduling and routing policies.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In a simple emergency system patients arrive to the hospital, check-in at the reception and wait until they are called for an examination. Consider a system composed of a single doctor and two waiting rooms with only one room having a stand-by intern/doctor. The stand-by intern collects information and makes an initial diagnosis on the patient's situation. Therefore the doctor spends less time in the examination compared to patients coming from other waiting room. Note the trade-off: patients who wait in the room with the intern are more costly to the hospital yet faster to examine. Hospital management then faces two decision problems: (1) which room to route a patient upon arrival, (2) which type of patient is sent to doctor, when she becomes free.

This example can be stylized as a 2-buffer single flexible server queueing model. Queues correspond to waiting rooms, the server to the doctor and for consistency with the queueing theory terminology we use the term "job" instead of "patient." We refer to jobs that wait in Queue 1(2) as class 1(2) jobs and both classes of jobs wait in queues for service. The server is flexible in the sense that it can process jobs of either class. However it can only process a single job at once. Jobs belonging to the same class i have the same time-homogeneous holding cost, c_i, and the service time in the server is an exponentially distributed random variable with rate μ_i. Furthermore, we assume that jobs arrive to the system according to a Poisson process with rate λ. Figure 1.1 depicts this system.

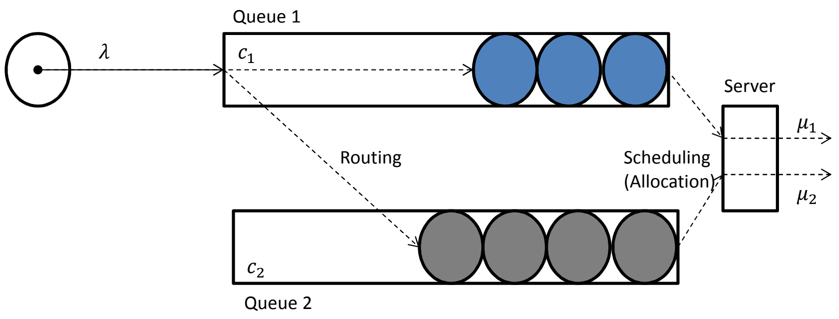


Figure 1.1: Two-buffer single flexible server model

The system manager needs to decide whether to route an incoming job to Queue 1 or Queue 2 and whether to process a class 1 or class 2 job when the server becomes available. What is then the optimal policy? We can consider the simple cases. First, suppose $c_1 = c_2$. It is then optimal to route every incoming job to the queue with faster service. Second, if $\mu_1 = \mu_2$ the optimal policy is to route any incoming patient to the queue with lower cost. How about when $c_1 > c_2$ and $\mu_1 > \mu_2$? Then, there is a trade-off between waiting costs and service times and the optimal policy is not obvious. It is intuitive to think that the actions that minimize the marginal cost may compose the optimal policy. Denoting the number of jobs in Queue i at time t by $Q_i(t)$, such a policy would route the incoming job to the queue with lowest $(c_i\mu_i(Q_i(t)+1))$. As we analyze in this dissertation, this turns out to be suboptimal.

We just introduced a small queueing system but this system can be generalized to a large number of queues, as shown in Figure 1.2. Corresponding to the initial example, there may be multiple waiting rooms with stand-by interns receiving different wages based on their rank. This type of routing option which we call Gurvich type routing, considers routing to different classes of jobs competing for service at the same station. In this dissertation, we focus on various topologies of queueing networks with Gurvich type routing. The models considered include single and multi-station networks in discrete-stochastic and continuous-deterministic contexts corresponding to queueing and fluid models, respectively.

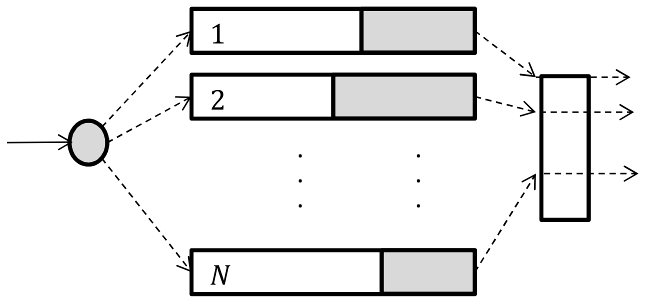


Figure 1.2: A single-station network with Gurvich-type routing

In the single-station case, as given in Figure 1.2, the network is theoretically interesting because of its affinity to two well-known classes of networks: multiclass queueing networks and inverted-V queueing networks. Without the routing control (assuming each queue has a dedicated arrival stream), a queueuing network with Gurvich type routing reduces to a multiclass network. For such a network, Cox and Smith [11] proved that processing priority should be given to the class of jobs with the highest product of processing rate and holding cost. This result, the $c\mu$ rule, implies that optimal scheduling policy is a static priority rule. Multiclass networks have many applications from production systems to packet processing in wireless networks and there has been significant work on extending the $c\mu$ rule to more general cost functions (see Van Mieghem [32], Mandelbaum and Stolyar [26], Gurvich and Whitt [17]).

If, instead of a single server every job class had a dedicated server the resulting network would be an inverted-V network. These networks model call centers, for example, and determining where to route an incoming job is one of the most challenging queueing control problems. Consider such a network with only two queues, each having their own server with one being fast and the other slow. With the goal of minimizing total number of jobs in system, the Faster Servers First (FSF) policy makes intuitive sense. This policy suggests that upon a job arrival or a service completion, a job should be sent to the fastest server available. Surprisingly, it is sometimes necessary to keep the customers waiting even if the slow server is idle (Lin and Kumar [24]). Even in small queueing systems no closed form optimal routing policy is available (Ahn and Lewis [1]).

Therefore, even in a small-scale queueing network, it is interesting to investigate the interplay between scheduling and this additional option of routing. In Chapter 2, we focus on the model shown in Figure 1.1. Key questions include: Does the $c\mu$ rule apply for the optimal scheduling policy? What is the structure of the optimal routing policy? Using the Markov decision process formulation of the model, we obtain structural results to answer questions regarding the structure of the optimal policy. The fact that the $c\mu$ rule holds and the routing policy is of threshold form motivate us to develop a heuristic based on a continuous (fluid) relaxation of this model. We use the optimal policy resulting from the fluid relaxation and additionally perform perturbation expansions for further improvements. Translating the resulting continuous policy into a discrete one requires particular care as naive translations may result in instability of the discrete-network (Stolyar [37]). Therefore, we study the asymptotic behaviour of the fluid based policy and propose a discrete translation that is asymptotically optimal in the total cost sense.

Although the queueing model analyzed in Chapter 2 gives insight into the behaviour of larger sized networks, the "curse of dimensionality" limits the queueing network analysis to moderate sized ones. Approximations are often used when policies for large-scale queueing systems are desired. Fluid approximations, as used in Chapter 2 to obtain an approximate policy, have been popular in the literature due to their tractability and close connections to their corresponding discrete-stochastic models. In the context of multi-class queueing networks, Dai [13] established a link between stability of fluid and discrete models and academic interest has been shifting to the optimization of these models. In that respect, Chapter 3 is devoted to the fluid model as shown in Figure 1.2. For this model, we again investigate the optimal scheduling and routing policies. Along with structural proofs on the optimal scheduling policies, we show that optimal routing policies are threshold-based and we develop a procedure to explicitly determine the set of policies.

As a natural extension to single-station models, multi-station models allow a more general modelling framework. A tandem network is a network in which a station is serially connected to another. Even 2-station tandem networks can have drastically different stability and optimality properties than a single-station network. For example, the $c\mu$ rule in a single station MCQN is proven not to hold for a two-station MCQN under particular parameter settings (Hordjik and Koole [19]). Given that fluid models are useful in giving insights on their discrete counterparts and the fact that one can translate a fluid optimal policy to the discrete setting and still get an asymptotically optimal policy (through the procedure outlined in Maglaras [25]), in Chapter 4 we again focus on fluid models. In that chapter, we analyze two different tandem models. In the former one, the first station is composed of a single buffer and server and the second station is a fluid network composed of N parallel buffers and Gurvich type routing. In the latter model, we again have a fluid network composed of N parallel buffers and Gurvich type routing, but this time in station one and the second station is now composed of a single buffer. Figure 1.3 and Figure 1.4 depict these networks.

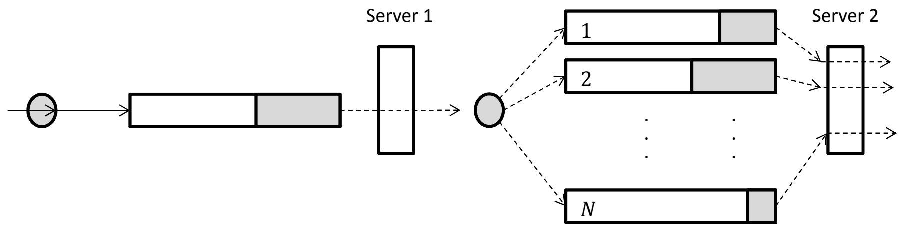


Figure 1.3: Network representation for the Tandem 1-N

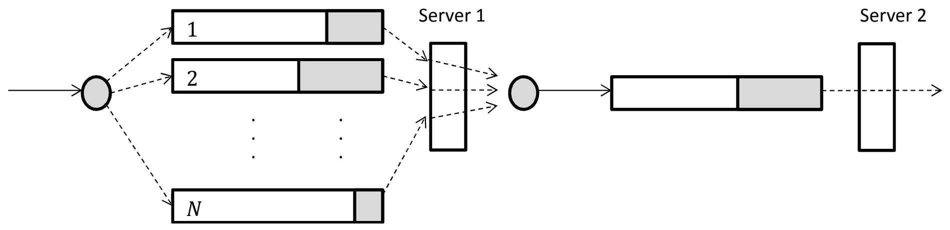


Figure 1.4: Network representation for the Tandem N-1

1.1 Contributions

Here, we provide a brief summary of our contributions in this dissertation.

Chapter 2:

- We analyze in detail a novel queueing network.
- We prove that the $c\mu$ rule holds for the optimal scheduling control of the server and that the optimal routing policy is of threshold form.
- We form and solve the associated fluid optimization problem.
- Through perturbation of the optimal fluid quadratic, we approximate

an offset term in the routing policy, and based on this offset term we propose a policy to be implemented in the discrete network.

- Using fluid limits, we prove that the proposed policy is asymptotically optimal and we analyze its performance through numerical studies.

Chapter 3:

- For a network composed of N-parallel buffers and a single flexible server, we prove that optimal scheduling policy conforms to the $c\mu$ rule.
- We prove that the optimal routing policy is bang-bang and that the class index that receives the incoming fluid is non-increasing with respect to time.
- We characterize the full structure of Lagrange multipliers which in turn lead to a general procedure to compute the optimal routing trajectory under any initial state.

Chapter 4:

For a Tandem 1-N network:

- We prove that for the second station of the network, the optimal scheduling control is the $c\mu$ rule.
- We outline explicitly the parameter regimes where the server in Station 1 never idles or idles until all buffers belonging to Station 2 are empty. We

also show that under particular parameter regimes, the optimal routing policy of Station 2 is not bang-bang.

- We prove that idleness in Station 1 has a special structure: there can exist at most a single time interval such that the server idles uninterruptedly.

- Following the implications of these proofs, we derive fully the Lagrange multipliers and then again show how to compute the optimal state trajectory starting from a given initial state.

For Tandem N-1 network:

- For the network with $N = 2$, we prove that a static index-based priority rule is optimal for the scheduling control of Station 1 under restricted parameter settings.

- For the same network, we show that a cycle-type behavior where a buffer first drains, then builds-up and re-drains is possible. Furthermore we determine parameter regimes where this phenomena arises.

- Lastly, we derive the Lagrange multipliers for the network with $N = 2$ which in turn allow us to compute the optimal state trajectory.

Chapter 2

A queueing model with two parallel buffers

Queueing theory finds applications in many different areas such as wireless telecommunications networks, call center management and semiconductor manufacturing. Various topologies arise in queuing networks in practice and often these networks include various operational constraints. Therefore, the research goals in the area include developing general modelling frameworks to accomodate these various networks. Yet, general modelling attempts have been fairly limited due to the trade-off between tractability and model complexity. As a result, research in the area has evolved in two major directions: exact analysis of small-scale models and approximations to large-scale ones. The analysis in this chapter fits into the first research direction.

In this chapter, we focus on a single-station queueing network composed of two parallel buffers and a single flexible server. The decision-maker has to determine which class of job is pushed into service when the server becomes idle and to which queue an incoming job is routed. If each of the queues had their own arrival stream, the network would reduce to a multi-class queueing network. For a MCQN with any number of parallel buffers and a single flexible server, the $c\mu$ rule gives the optimal allocation sequence when the holding cost is a linear combination of the number of tasks in the competing queues. Following this rule, the queues are ordered according to the value of the product $c\mu$, from largest to smallest, and the server always selects a task from the first queue (the one with largest $c\mu$ value) unless it is empty; in that case, the server selects the second queue and so on. A striking property of this rule is that it neither depends on the arrival streams of individual queues nor on the number of jobs at a queue. The rule is intuitive to grasp and very practical for applications.

In literature many studies explore the validity of this rule in various other network topologies and different cost settings. A related model to ours can be found in Koole [23] in which a system with two parallel queues and a flexible server is analyzed. Both of the queues receive independent Poisson arrivals and the server incurs switching costs from moving from one queue to another. It is shown that in addition to linear holding costs, if one considers linear switching costs from one queue to another, the $c\mu$ rule only holds when switching costs are in accordance with holding costs. Harrison [18] shows that the $c\mu$ rule can be highly sub-optimal, in fact it can lead to an unstable system for a network with two servers working in parallel. There has also been significant effort in extending the $c\mu$ rule for more general cost functions. For example, Mandelbaum and Stolyar [26] prove that, under an asymptotic (heavy-traffic) regime, a generalized (cost is a convex function of queue length) $c\mu$ rule is optimal for a network composed of multiple job classes and flexible servers. Further work in the area includes Gurvich and Whitt [17] as well as

Tezcan and Dai [38].

In our model, through Gurvich-type routing, the arrival streams of the two buffers become part of the decisions. Despite the interplay of routing and scheduling decisions, we show in Section 2.3 that the $c\mu$ rule is, in fact, optimal for server scheduling. When the scheduling policy is fully determined, the problem reduces to a routing problem where the decision-maker decides to route an incoming job. When a job is routed to Queue 1 (2), it becomes a class 1 (2) job, incurring a holding cost $c_1(c_2)$ and receiving random service time with rate $\mu_1(\mu_2)$. There is a significant amount of literature on routing to parallel queues. The join the shortest queue (JSQ) policy has been analyzed thoroughly dating back to the work of Kingman [22]. Winston [41] proved the optimality of shortest queue under exponential job size distributions. Later on, Horjik and Koole [20] studied assignment problems in a Markov decision process (MDP) framework and demonstrated that that an arriving customer should prefer a faster server and shorter queue. The routing models also attract significant attention in call-center management. For a thorough review on the relevant literature, we refer the interested reader to Armony [3] and Aksin et al. [2].

Through proofs based on the value function, we prove that the optimal routing policy is of threshold-form. A natural question is, how can we determine or approximate this threshold? The answer lies in fluid models. By capturing the asymptotic behaviour of discrete stochastic queueing systems, fluid models are useful continuous approximations to their discrete counterparts. Chen and Mandelbaum [9] show that a class of queuing networks converges under appropriate time and space scaling to fluid networks. Moreover, recent results have shown a close connection between stability of stochastic networks and stability of their associated fluid models (Dai [12], Stolyar [37]). Corresponding fluid models are easier to solve than stochastic queueing networks and this greatly motivates the translation of fluid optimal control to discrete stochastic models. Fluid models for reentrant-lines (multiclass queueing networks with multiple stations and a single class) are analyzed in Weiss [39]. Avram et al. [5] provide optimal fluid policies for networks in which fluid classes compete for service. Furthermore, fluid optimal policies can better approximate discrete optimal ones through perturbation expansions as outlined in Avram [4].

Although fluid models are more tractable than their discrete counterparts, how to translate a fluid policy into the discrete setting has been an important question. For example, naive translations in a 2-station tandem network composed of two classes of jobs, can in fact yield to an unstable discrete system. Furthermore, when optimal fluid policy is translated into a discrete setting, a desired condition is that it satisfies an optimality criterion in the discrete system itself. For that purpose, asymptotic optimality provides a consistency criterion between fluid and discrete models. Important works on establishing conditions of asymptotic optimality include Maglaras [25], Bauerle [6] and Meyn [29]. Our analysis is also based on computing the fluid optimal policy, improving it through perturbation expansions and studying its asymptotic properties. We show in fact, that the proposed fluid-based policy is asymptotically optimal under fluid scaling.

This chapter is organized in 7 sections. In Section 2.2, we formally describe the model. Section 2.3 is devoted to structural results on optimal policies. There we prove that the optimal scheduling control is given by the $c\mu$ rule and the optimal routing policy is of threshold type. Later on, we define and solve the associated fluid model in Section 2.4. Further improvements on the fluid policy are performed in Section 2.5. In Section 2.6, we analyze the asymptotic behaviour of the fluid policy. After translating fluid policy to discrete-system, we present the performance of the proposed policy with respect to the optimal policy in Section 2.7.

2.1 Preliminaries and model description

The network under study has two job classes and one flexible server. Job classes are labelled by $k = 1, 2$ and we use the same index to denote their respective queues. The server can only process a single job at a given time. The service times are assumed to be i.i.d. exponential random variables with rates μ_1 and μ_2 for job classes 1 and 2, respectively. Without loss of generality we set $\mu_1 > \mu_2$. Jobs arrive to the system according to a Poisson process with rate λ where $\lambda < \mu_1$ to insure that the system is stabilizable. The decision-maker determines to which queue to route an incoming job upon arrival. Routing the job upon arrival defines its class and this decision cannot be rescinded. Upon service termination, the decision maker also determines which type of job is served next. In addition, if a high priority job arrives we allow preemption of lower priority jobs (since all service times are exponential, it does matter if we employ a "preempt-resume" assumption or not). A job of class k incurs a holding cost c_k per unit time with $c_1 > c_2$ and the goal of the decision-maker is to characterize optimal routing and allocation policy in order to minimize the average total cost rate over the infinite horizon. A depiction of the system of interest appears in Figure 2.1. We sometimes refer to the parameter assumptions as falling into the *strong $c\mu$ case*. Notice that we have $c_1\mu_1 > c_2\mu_2$, so without routing, the $c\mu$ is the optimal scheduling rule (we prove later that it is still optimal, with routing).

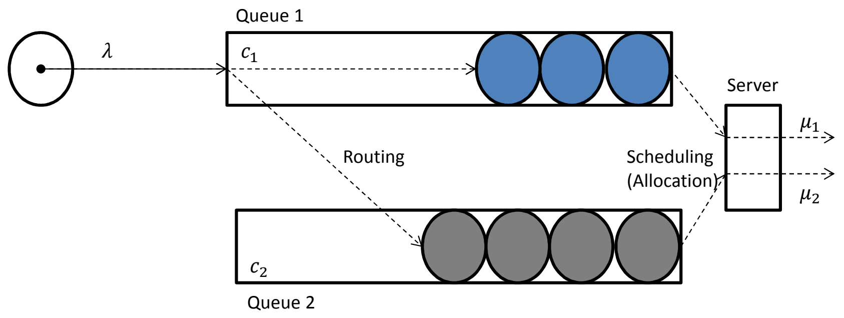


Figure 2.1: 2-buffer queueing network with Gurvich-type routing

The decision-maker's problem can be modelled via a continuous time Markov decision process (CTMDP). The state definition of the CTMDP then includes the number of jobs in each class (including any job in service) and the class of job currently being processed. Explicitly, the state space is $S := \{(i(t), j(t), C(t)) : i(t) \in \mathbb{Z}_+, j(t) \in \mathbb{Z}_+, C(t) \in \{C_0, C_1, C_2\}\}$ where $i(t)$ and

$j(t)$ refer to the number of jobs in each class at time t and $C(t)$ denotes the class of job being served at time t with C_k denoting that a class k job is in service. $C(t) = C_0$ when no jobs are in service at time t. The current number of jobs in class k is denoted by $Q_k(t)$ and if the dependence on a given policy π is to be emphasized then we write $Q_k^\pi(t)$. The routing action, denoted as $U_R(t)$, is employed when there is an arrival to the system at time t. $U_R(t)$ is an element of the set $\{R1_P, R1_{NP}, R2_P, R2_{NP}\}$ where $R1$ and $R2$ indicate if jobs are routed to class 1 or class 2. The subscripts P and NP indicate whether or not preemption is employed. In a similar fashion the scheduling action, $U_A(t)$ is employed when the server finishes processing a job. $U_A(t)$ is an element of the set $\{Q1, Q2\}$. $Q1$ indicates that the system next serves a class 1 job and $Q2$ indicates the same for a class 2 job. Given an initial state of the system (i, j, C) the decision-maker uses a policy π which in the usual manner indicates the sequence of actions to be taken when there is a change of state. The objective is to minimize the total average cost rate given as follows:

$$\limsup_{t\to\infty} \frac{1}{t}\mathbb{E}_{(i,j,C)}\left(\int_0^t [c_1 Q_1^\pi(s) + c_2 Q_2^\pi(s)] ds \right).$$

For purposes of deriving structural results we also consider the total discounted cost version of the objective function given as:

$$\mathbb{E}_{(i,j,C)}\left(\int_0^\infty e^{-\beta s}\left[c_1 Q_1^\pi(s) + c_2 Q_2^\pi(s)\right] ds\right),$$

for a discount factor $\beta > 0$.

We now define performance processes that describe the evolution of the CTMDP. For $t \geq 0$, let $E(t)$ be the number of exogenous jobs that arrive in

$[0,t]$. $D_k(t)$ indicates the number of service completions of class k customers in $[0,t]$ if the server devotes all its time to class k in $[0,t]$. $\phi_k(m)$ is the number of jobs that are routed to Queue k among the first m arrivals. We use $A_k(t)$ to denote the number of jobs that arrive to Queue k in $[0,t]$. Let $T_k(t)$ be the cumulative time the server has spent processing class k jobs in $[0,t]$. $Y(t)$ refers to the cumulative time the server has spent idle in $[0,t]$.

We assume that $E(t)$ and $D_k(t)$ are right-continuous with left limits. Furthermore, in our model $E(t)$ is a Poisson process with rate λ and $D_k(t)$ is a Poisson process with rate μ_k. We then define the 5-tuple $\mathbb{X}(t) = (Q(t),A(t),D(t),T(t),Y(t)),\ t \geq 0$ as the queueing network process. Formally the dynamics of $\mathbb{X}$ is then defined via following equations, for $t \geq 0$ and $k \in \{1,2\}$:

$$
\begin{aligned}
Q_k(t) &= Q_k(0) + A_k(t) - D_k(T_k(t)) && (2.1)\\
A_k(t) &= \phi_k(E(t)) && (2.2)\\
T_1(t) + T_2(t) + Y(t) &= t, && (2.3)
\end{aligned}
$$

where the functions $T_k(t), \phi_k(m)$ are determined by the server assignment and routing controls, $U_r(t)$ and $U_a(t)$.

As usual, we uniformize the CTMDP and formulate a corresponding discrete Markov decision process (DTMDP). Let $\Lambda = \lambda + \mu_1$ be the uniformization constant. Without loss of generality assume that $\Lambda = 1$ and let $\delta = \frac{\Lambda}{\Lambda+\beta}$. Note that the discrete-time equivalent of $\mathbb{X}$ is defined by discrete-time equivalents of $Q(t)$, $A(t)$, $D(t)$, $T(t)$, $Y(t)$ and is still defined by (2.1)-(2.3). Dropping the time index t a state is denoted as (i, j, C). In order to present the state evolution for the discrete-time model, let f be a real-valued function measuring the value of each state and define a mapping H as follows:

$$
\begin{aligned}
H_\delta f(i,j,C) =& \delta\lambda \min \begin{cases} f(i+1,j,C_1) \\ f(i,j+1,C_1) \\ f(i+1,j,C_2) \\ f(i,j+1,C_2) \end{cases} \\
&+ \ \delta \min \begin{cases} \mu_1 f((i-1)^+,j,C_1) + (\Lambda-\mu_1) f(i,j,C) \\ \mu_2 f(i,(j-1)^+,C_2) + (\Lambda-\mu_2) f(i,j,C). \end{cases}
\end{aligned}
$$

In the expression above, the minima represent the routing and the scheduling controls, respectively. The optimality equations for the average cost formulation then satisfy the following set of equations:

$$g + y(i,j,C) = c_1 i + c_2 j + H_1 y(i,j,C). \tag{2.4}$$

where g is the optimal average cost and y is the relative value function. Similarly, the discrete-time finite and infinite discounted horizon formulations are: $v_\delta^{n+1}(i,j,C) = c_1 i + c_2 j + H_\delta v_\delta^n(i,j,C)$ and $v_\delta(i,j,C) = c_1 i + c_2 j + H_\delta v_\delta(i,j,C)$, respectively. In Section 3, we derive structural results for the relative value function y.

A state (i,j,C) is *accessible* from state (i',j',C') if there exists a stationary policy μ and an integer k such that $P\left(x_k = (i',j',C') | x_0 = (i,j,C), \mu\right) > 0$. The *weak accessibility* (WA) condition holds if the states can be partitioned into two subsets S_t and S_c such that all states in S_t are transient under every stationary policy and for every two state pairs (i,j,C) and (i',j',C') in S_c,

(i', j', C') is accessible from (i, j, C). If the WA condition holds then optimum average cost is the same for all initial states and there exists an optimal policy that is unichain [8].

Theorem 2.1.1. *The weak accessibility condition holds for the model described in (2.1)-(2.4).*

Proof. Consider the policy that simply routes customers to class 1 with probability 0.5 and schedules customers by giving preemptive priority to class 1. Then, it is clear that state $(0, 0, C_0)$ is accessible from any state and similarly any state is accessible from $(0, 0, C_0)$ under this policy. Hence, S_c can be taken to be S in the WA characterization. $\square$

2.2 Structural results

In this section, we present structural results for the average-cost relative value function $y(i, j, C)$ and the corresponding implications for optimal policies. We first obtain results for the discounted cost finite horizon problem with value function $(v_\delta^n(i, j, C))$. Using standard techniques we extend the results to the infinite horizon average cost case. The first important result of this section is that the $c\mu$ rule holds for optimal scheduling policy. Thus if $c_1\mu_1 > c_2\mu_2$, then it is always optimal to process a job from Queue 1 if such a job is present. In this paper, we assume the parameters fall into the strong $c\mu$ case, and hence the preceding inequality also holds. The second important result is that the optimal routing policy is of the threshold type. The proof is organized by establishing the following statements for various values of n:

- Statement 1: $v_\delta^n(i,j,C_1) \leq v_\delta^n(i,j,C_2)$, $\forall i \geq 1, \forall j \geq 1$,
- Statement 2: $v_\delta^n(i,j,C_1)$ is non-decreasing in i for each fixed j and is non-decreasing in j for each fixed i,
- Statement 3: $\Delta^n(i,j) = v_\delta^n(i{+}1,j,C_1){-}v_\delta^n(i,j{+}1,C_1)$ is non-decreasing in i given fixed $j \geq 1$ and non-increasing in j given fixed $i \geq 1$,
  $\forall i \geq 1, \forall j \geq 0$,
- Statement 4: $\mu_1 v_\delta^n(i{-}1,j,C_1) \leq \mu_2 v_\delta^n(i,j{-}1,C_1){+}(\mu_1{-}\mu_2)v_\delta^n(i{+}1,j{-}1,C_1)$,
  $\forall i \geq 1$, $\forall j \geq 0$,
- Statement 5: $\mu_1 v_\delta^n(i{-}1,j,C_1) \leq \mu_2 v_\delta^n(i,j{-}1,C_1)$ $+(\mu_1 - \mu_2)v_\delta^n(i,j,C_2)$,
  $\forall i \geq 1, \forall j \geq 0$.

Theorem 2.2.1. *If $c_1 > c_2$ and $\mu_1 > \mu_2$, then Statements 1-5 hold for $n = 0$.*

Proof. For $n = 0$ there is only a one-step cost which leads to the relatively simple arguments below:

- Statement 1 : Both expressions are equal to $c_1 i{+}c_2 j$, so the statement holds.
- Statement 2 : We have $v_\delta^0(i{+}1,j,C_1){-}v_\delta^0(i,j,C_1) = c_1 \geq 0$ and
  $v_\delta^0(i,j{+}1,C_1)$ $-v_\delta^0(i,j,C_1)$ $=c_2 \geq 0$. Therefore $v_\delta^0(i,j,C_1)$ is monotonically non-decreasing in i and j.

- Statement 3 : Since $\Delta^0(i,j) = c_1{-}c_2$, the statement clearly holds.
- Statement 4 : Note that

  $$\mu_1(c_1(i{-}1){+}c_2j) \leq \mu_2(c_1i{+}c_2(j{-}1)){+}(\mu_1{-}\mu_2)(c_1(i{+}1){+}c_2(j{-}1)),$$

  implies the following:

  $$\Rightarrow 0 \leq c_1\mu_1{-}c_2\mu_2{+}(c_1{-}c_2)(\mu_1{-}\mu_2).$$

  Since $c_1\mu_1 \geq c_2\mu_2, c_1 \geq c_2$ and $\mu_1 \geq \mu_2$, the inequality holds.
- Statement 5 : Note that $0 \leq c_1\mu_1{-}c_2\mu_2$ implies that

  $$\mu_1(c_1(i{-}1){+}c_2j) \leq \mu_2(c_1i{+}c_2(j{-}1){+}(\mu_1{-}\mu_2)(c_1i{+}c_2j),$$

  which verifies the statement.

□

Theorem 2.2.2. *Suppose $c_1 > c_2$ and $\mu_1 > \mu_2$. If Statements 1 to 5 hold for n then they also hold for $n{+}1$.*

Proof of Theorem 2.2.2. Note that if Statement 1 holds at time n, then $\forall i \geq 1, \forall j \geq 0$ and $\forall n \geq 0$ we have:

$$\begin{aligned} v_\delta^{n+1}(i,j,C_2){-}v_\delta^{n+1}(i,j,C_1) &= \delta\mu_2 v_\delta^n(i,(j{-}1)^+,C_1){+}\delta(\mu_1{-}\mu_2)v_\delta^n(i,j,C_2) \\ &\quad -\delta\mu_1 v_\delta^n((i{-}1)^+,j,C_1). \end{aligned}$$

If Statement 1 and 5 holds at time n, $v_\delta^{n+1}(i,j,C_1) \leq v_\delta^{n+1}(i,j,C_2)$ implies that Statement 1 holds at time $(n{+}1)$. This result requires that Statements

1-4 hold at time (n+1), therefore we characterize each result individually by Lemmas 2.2.3, 2.2.4, 2.2.5, 2.2.6.

Lemma 2.2.3. *Statement 1 at time n, Statement 2 at time n $\Rightarrow$ Statement 2 at time* (n+1)*.*

Proof. Given $v_\delta^n(i,j,C_1) \leq v_\delta^n(i,j,C_2)$, the marginal increase $v_\delta^{n+1}(i{+}1,j,C_1)$- $v_\delta^{n+1}(i,j,C_1)$ in i, at time (n+1) can be expressed as follows:

$$\begin{aligned} &= c_1{+}\delta\lambda\Big(\min(v_\delta^n(i{+}2,j,C_1), v_\delta^n(i{+}1,j{+}1,C_1)) \\ &\quad - \min(v_\delta^n(i{+}1,j,C_1), v_\delta^n(i,j{+}1,C_1))\Big) + \delta\mu_1(v_\delta^n(i,j,C_1){-}v_\delta^n(i{-}1,j,C_1)) \\ &\geq c_1{+}\delta\lambda\Big(\min(v_\delta^n(i{+}2,j,C_1){-}v_\delta^n(i{+}1,j,C_1), \\ &\quad v_\delta^n(i{+}1,j{+}1,C_1){-}v_\delta^n(i,j{+}1,C_1)\Big){+}\delta\mu_1(v_\delta^n(i,j,C_1){-}v_\delta^n(i{-}1,j,C_1)). \end{aligned}$$

The following comes from initial assumption:

$$\min(v_\delta^n(i{+}2,j,C_1){-}v_\delta^n(i{+}1,j,C_1), v_\delta^n(i{+}1,j{+}1,C_1){-}v_\delta^n(i,j{+}1,C_1)) \geq 0.$$

In addition $v_\delta^n(i{+}1,j,C_1){-}v_\delta^n(i{-}1,j,C_1) \geq 0$. Thus $v_\delta^n(i,j,C_1){-}v_\delta^n(i{-}1,j,C_1)$ is non-negative. Hence the argument holds for time (n+1) as well. A symmetric argument can be used to prove the monotonicity in j.

□

Lemma 2.2.4. *Statement 1 at time n, Statement 3 at time n $\Rightarrow$ Statement 3 at time (n+1).*

Proof. $\Delta^{n+1}(i,j)$ can be expressed as follows:

$$\begin{aligned}\Delta^{n+1}(i,j) &= (c_1{-}c_2) + \delta\lambda \min(v^n_\delta(i{+}2,j,C_1), v^n_\delta(i{+}1,j{+}1,C_1))\\ &\quad -\delta\lambda \min(v^n_\delta(i{+}2,j,C_1), v^n_\delta(i{+}1,j{+}2,C_1))\\ &\quad +\delta\mu_1(v^n_\delta(i,j,C_1) - v^n_\delta(i{-}1,j{+}1,C_1)).\end{aligned}$$

□

Next, we prove that each term in the RHS above is non-decreasing in i and non-increasing in j. The first term is constant therefore plays no role on monotonicity. The second term can be expressed as follows (omitting λ for clarity):

$$\begin{aligned}&= \min(v^n_\delta(i{+}2,j,C_1), v^n_\delta(i{+}1,j{+}1,C_1)) - \min(v^n_\delta(i{+}1,j{+}1,C_1), v^n_\delta(i,j{+}2,C_1))\\ &= \min(v^n_\delta(i{+}2,j,C_1){-}v^n_\delta(i{+}1,j{+}1,C_1),0) - \max(v^n_\delta(i{+}1,j{+}1,C_1){-}v^n_\delta(i,j{+}2,C_1))\\ &= \min(\Delta^n(i{+}1,j),0) + \max(\Delta^n(i,j{+}1),0).\end{aligned}$$

By initial assumption $\min(\Delta^n(i{+}1,j),0) + \max(\Delta^n(i,j{+}1),0)$ is non-decreasing in i and non-increasing in j. As for the last term, it is equal to $\mu_1\Delta^n(i{-}1,j)$ therefore non-decreasing in i and non-increasing in j following initial assumption.

For the remaining proofs, discount factor δ is multiplied by both sides of the inequalities therefore we omit it for clarity.

Lemma 2.2.5. *Statement 1, 2, 3, 4 at time n $\Rightarrow$ Statement 4 at time $(n{+}1)$.*

Proof. Recall that the value function at time $(n{+}1)$ of a particular state-pair combination is composed of step costs, arrivals and departures. In order to prove that Statement 4 holds for time $(n{+}1)$, we decompose value functions into following individual components at time n.

- Step costs :

$$\begin{aligned} \mu_1(c_1(i{-}1){+}c_2 j) &\leq \mu_2(c_1 i{+}c_2(j{-}1)){+}(\mu_1{-}\mu_2)(c_1(i{+}1){+}c_2(j{-}1)) \\ 0 &\leq c_1\mu_1{-}c_2\mu_2{+}(c_1{-}c_2)(\mu_1{-}\mu_2). \end{aligned}$$

- Departures :

  By the initial assumption $\forall i \geq 0, \forall j \geq 0, v_\delta^n(i,j,C_1) \leq v_\delta^n(i,j,C_2)$. Thus a customer from Queue 1 will always be processed over a customer from Queue 2. Therefore the terms relating to departures would hold by the initial arguments at time n:

$$\Rightarrow \mu_1 v_\delta^n(i{-}2,j,C_1) \leq \mu_1\mu_2 v_\delta^n(i{-}1,j{-}1,C_1){+}\mu_1(\mu_1{-}\mu_2)v_\delta^n(i,j{-}1,C_1).$$

- Arrivals :

  To compare arrival terms all the possible routing combinations should be taken into account. Table 2.1 shows such possible combinations. Note that $(v_\delta^n(i,j,C_1) \leq v_\delta^n(i,j,C_2))$ implies that at any state (i,j,C_1) preemption is suboptimal therefore are omitted.

Case	$v_\delta^n(i-1,j,C_1)$	$v_\delta^n(i,j-1,C_1)$	$v_\delta^n(i{+}1,j-1,C_1)$
1	$R1_{NP}/R2_{NP}$	$R1_{NP}$	$R1_{NP}$
2	$R1_{NP}/R2_{NP}$	$R2_{NP}$	$R2_{NP}$
3	$R2_{NP}$	$R2_{NP}$	$R1_{NP}$

Table 2.1: Routing actions for states $(i\text{-}1,j,C_1),(i,j\text{-}1,C_1),(i{+}1,j\text{-}1,C_1)$

Table 2.1 shows only feasible routing combinations. For example if the routing action of $v_\delta^n(i,j{-}1,C_1)$ is $R1_{NP}$ then by Statement 3 same action should be chosen for $v_\delta^n(i{+}1,j{-}1,C_1)$. If the routing action for $v_\delta^n(i,j{-}1,C_1)$ is $R2_{NP}$ and the routing action for $v_\delta^n(i{+}1,j{-}1,C_1)$ is $R1_{NP}$ then the routing action for $v_\delta^n(i{-}1,j,C_1)$ should be $R2_{NP}$.

In Cases 1 and 2, the routing action of $(i{-}1,j,C_1)$ can be either $R1_{NP}$ or $R2_{NP}$. Note that this term is on LHS therefore it is sufficient to prove that the inequality is satisfied either for $R1_{NP}$ or $R2_{NP}$. For Case 1 (2) we can prove the inequality for routing action $R1_{NP}$ ($R2_{NP}$). Consequently for both of these cases the arrivals simply imply a fixed increase in one index and multiplication by a constant. Equations (2.5) and (2.6) correspond to Case 1 and 2 as follows:

$$\lambda\mu_1 v_\delta^n(i,j,C_1) \leq \lambda\mu_2 v_\delta^n(i{+}1,j{-}1,C_1){+}\lambda(\mu_1{-}\mu_2)v_\delta^n(i{+}2,j{-}1,C_1) \quad (2.5)$$

$$\lambda\mu_1 v_\delta^n(i{-}1,j{+}1,C_1) \leq \lambda\mu_2 v_\delta^n(i,j,C_1){+}\lambda(\mu_1{-}\mu_2)v_\delta^n(i{+}1,j,C_1). \quad (2.6)$$

These inequalities hold by the induction assumption. For Case 3, we have to prove that following inequality holds: $\lambda\mu_1 v_\delta^n(i{-}1,j{+}1,C_1) \leq \lambda\mu_2 v_\delta^n(i,j,C_1){+}\lambda(\mu_1{-}\mu_2)v_\delta^n(i{+}2,j,C_1)$.

Recall that the routing action of $(i{-}1, j, C_1)$ implies $v_\delta^n(i{-}1, j, C_1) \leq v_\delta^n(i, j, C_1)$ therefore proving $\lambda(\mu_1{-}\mu_2)v_\delta^n(i, j, C_1) \leq \lambda(\mu_1{-}\mu_2)v_\delta^n(i{+}1, j, C_1)$ is sufficient. Recall that Statement 2 implies that $v_\delta^n(i{+}1, j, C_1) \geq v_\delta^n(i, j, C_1)$, therefore the inequality holds. Having considered all the possible routing combinations, we proved that at time $(n{+}1)$:

$$\mu_1 v_\delta^{n+1}(i{-}1, j, C_1) \leq \mu_2 v_\delta^{n+1}(i, j{-}1, C_1){+}(\mu_1{-}\mu_2)v_\delta^{n+1}(i{+}1, j{-}1, C_1).$$

□

Now using Lemmas 2.2.3 to 2.2.5, we prove that Statement 5 holds at time $(n{+}1)$.

Lemma 2.2.6. *Statement 1, 2, 3, 4, 5 at time $n \Rightarrow$ Statement 5 at time $(n{+}1)$.*

Proof. The proof follows the same lines as Lemma 2.2.5. We start by comparing individually step costs, terms corresponding to departures and arrivals.

- Step costs:

$$\begin{aligned} \mu_1(c_1(i{-}1){+}c_2 j) &\leq \mu_2(c_1 i{+}c_2(j{-}1)){+}(\mu_1{-}\mu_2)(c_1 i{+}c_2 j)) \\ 0 &\leq c_1\mu_1{-}c_2\mu_2. \end{aligned}$$

- Departures:

  Terms relating to departures can be expressed as:

$$\Rightarrow \mu_1 v_\delta^n(i{-}2, j, C_1) \quad \leq \quad \mu_1\mu_2 v_\delta^n(i{-}1, j{-}1, C_1){+}\mu_1(\mu_1{-}\mu_2)v_\delta^n(i, j{-}1, C_1).$$

Note that this inequality holds via Statement 3.

- Arrivals :

  Again we compare possible routing scenarios. Table 2.2 shows possible routing combinations for states $(i{-}1, j, C_1), (i, j{-}1, C_2), (i, j, C_2)$. Note that now we include preemption cases because of states with a C_2 component.

Case	$v^n_\delta(i-1, j, C_1)$	$v^n_\delta(i, j-1, C_1)$	$v^n_\delta(i, j, C_2)$
1	$R1_{NP}/R2_{NP}$	$R1_{NP}$	$R1_P$
2	$R1_{NP}/R2_{NP}$	$R2_{NP}$	$R2_P$
3	$R1_{NP}/R2_{NP}$	$R2_{NP}$	$R1_P/R2_P$
4	$R2_{NP}$	$R1_{NP}$	$R2_P$

Table 2.2: Routing actions for states $(i\text{-}1, j, C_1), (i, j\text{-}1, C_1), (i, j, C_2)$

Similar to the ones in Lemma 2.2.5, Cases 1 and 2 imply that arrivals result in simply a fixed increase in one index and multiplication by a constant. Case 3 can be expressed as follows:

$$\lambda\mu_1 v^n_\delta(i, j, C_1) \leq \lambda\mu_2 v^n_\delta(i, j, C_1) + \lambda(\mu_1 - \mu_2)\min(v^n_\delta(i{+}2, j, C_1), v^n_\delta(i, j{+}1, C_1)),$$

which holds as $\min(v^n_\delta(i{+}1, j, C_1), v^n_\delta(i, j{+}1, C_1) \geq v^n_\delta(i, j, C_1)$ by Statement 1. Therefore:

$$\Rightarrow \mu_1(v^n_\delta(i, j{+}1, C_1) - v^n_\delta(i, j, C_1)) \quad \geq \quad \mu_2(v^n_\delta(i, j{+}1, C_1) - v^n_\delta(i{+}1, j{-}1, C_1)),$$

proving Case 3. Now we move one to Case 4. Note that from Lemma 2.2.4, $v^n_\delta(i{+}1, j{-}1, C_1) \geq v^n_\delta(i{-}1, j{+}1, C_1)$. Thus following inequality holds:

$$\mu_1(v_\delta^n(i, j{+}1, C_1){-}v_\delta^n(i{-}1, j{+}1, C_1)) \geq \mu_2(v_\delta^n(i, j{+}1, C_1){-}v_\delta^n(i{+}1, j{-}1, C_1)),$$

which can be expressed as:

$$\lambda\mu_1 v_\delta^n(i{-}1, j{+}1, C_1) \geq \lambda\mu_2 v_\delta^n(i{+}1, j{-}1, C_1){+}\lambda(\mu_1{-}\mu_2)v_\delta^n(i, j{+}1, C_1),$$

which corresponds to Case 4.

As in Lemma 2.2.5, enumeration of all possible routing scenarios leads us to prove that at time $(n{+}1)$ the following inequality holds:

$$\mu_1 v_\delta^{n+1}(i{-}1, j, C_1) \quad \leq \quad \mu_2 v_\delta^{n+1}(i, j{-}1, C_1){+}(\mu_1{-}\mu_2)v_\delta^{n+1}(i, j, C_2).$$

□

□

Corollary 2.2.7. *If $c_1 > c_2$ and $\mu_1 > \mu_2$ then $\forall i \geq 1$, $j \geq 1$, and $n \geq 1$, $v_\delta^n(i, j, C_1) \leq v_\delta^n(i, j, C_2)$. Thus for the discounted-cost finite horizon model, under any optimal policy it is optimal to give preemptive priority to class 1 jobs.*

Proof. The proof proceeds by induction on n. Theorem 2.2.1 establishes the result for $n = 0$. By Theorem 2.2.2 if the statement holds for n, then it must hold for $n{+}1$. Hence this implies that $\forall n \geq 0, \forall i \geq 1, \forall j \geq 1, v_\delta^n(i, j, C_1) \leq v_\delta^n(i, j, C_2)$. Recall that if there is at least 1 job available in both queues, then the optimal scheduling decision is to select the job from class $i^* = \arg\min_{i\in(1,2)} v_\delta^n(i,j,C_i)$. Therefore it is always optimal to give preemptive priority to class 1 jobs. $\square$

Corollary 2.2.8. *If $c_1 > c_2$ and $\mu_1 > \mu_2$ then the optimal routing policy for the discounted-cost finite horizon problem has a threshold form.*

Proof. For $i > 0$, when a job arrives to the system, Corollary 2.2.7 implies that the next job in service will be of class 1 (possibly through preemption). Therefore, the arrival decision reduces to minimization of $v_\delta^n(i{+}1,j,C_1)$ and $v_\delta^n(i,j{+}1,C_1)$ for $i \geq 1, \forall j \geq 0, \forall n \geq 1$. By Theorem 2.2.2 Statement 3 indicates that $v_\delta^n(i{+}1,j,C_1) - v_\delta^n(i,j{+}1,C_1)$ is non-decreasing in $i(j)$ given fixed $j(i)$ $\forall i \geq 1, \forall j \geq 0$ and $\forall n \geq 0$. Therefore $\Delta^n(i,j)$ changes sign at most once when either i or j is fixed, proving the existence of a switching policy.

$\square$

Theorem 2.2.9. *The results of Corollaries 2.2.7 and 2.2.8 hold under the average cost criterion.*

Proof. The initial step of the proof is based on establishing the link between finite and infinite horizon discounted-cost problems. Note that given the one-step costs are non-negative, v_δ^n is increasing in n. By Proposition 4.3.1 in Sennott [36] v_δ^n forms a monotonically increasing sequence with $\lim_{n\to\infty} v_\delta^n := v_\delta$. Furthermore, if π_δ^n is an optimal policy for the finite horizon problem, then any limit point of the sequence $\pi_{\delta,n_{n\geq 1}}$ is discount optimal for the infinite horizon problem. Thus, Theorems 2.2.1 and 2.2.2 then hold with v_δ^n replaced by v_δ.

The next step is to make the correspondence between discounted cost and average cost optimal value functions. Conditions under which the average cost optimal policy is obtained from the limit of the discounted cost optimal policies are outlined in Sennott [36]. A sufficient condition is that there exists a policy with finite average cost and that the Markov chain under this policy is either irreducible or consists of a single recurrent class (the transient states are absorbed in finite expected time).

For that purpose, we follow the construction in Ahn and Lewis [1]. Consider the stationary policy that routes every incoming job to Queue 1. It is then clear that every state with a positive number of class 2 jobs is transient. Thus, under this policy, the system reduces to a positive recurrent $M/M/1$ queue. By Little's law, the average queue length of Queue 1 is given by $L = (\lambda/(\mu_1 - \lambda))$ and the average cost rate is then $Lc_1 < \infty$.

Applying Theorems 7.2.3 and 7.5.6 in Sennott [36] we then have:

1. $v_\delta(i, j, C) = v_\delta(i, j, C) - v_\delta(0, 0, C_0)$ converges along a subsequence to a function on y such that (g, y) satisfy average cost optimality equations,
2. Any limit point of a sequence of discounted cost optimal policies (as $\delta \to 1$) is average cost optimal.

As a consequence, the results of Theorems 2.2.1 and 2.2.2 hold under average cost criterion with v_δ replaced by the relative value function y.

$\square$

2.3 Fluid approximation

There exist various numerical solution approaches to compute optimal average cost policies for discrete MDP's. Value iteration, policy iteration and linear programming are among well–known approaches. It should be noted that all these approaches suffer from curse of dimensionality as the state–action space grows large. Continuous approximation schemes to discrete–systems help overcome these issues and characterize near–optimal policies. Fluid models give insight on the original discrete model by replacing discrete jobs with continuous fluids, servers with fluid pumps and queues with buffers. In this scheme, the randomness in the discrete–stochastic model is simplified as the arrival and service processes are characterized only via their average rates. Fluid models are particularly useful in establishing stability results on the original model. In our context, we employ the fluid approximation to further characterize a near–optimal routing policy for the discrete model.

First, we note the correspondence between discrete and fluid models. A fluid model is the deterministic equivalent of the queueing network $\mathbb{X}$, where the arrival and service rates are replaced by $\lim_{t\to\infty} \frac{E(t)}{t} = \lambda$ and $\lim_{t\to\infty} \frac{D_k(t)}{t} = \mu_k$ for $k = 1, 2$.

The routing control vector $u_r(s)$ is composed of pair $u_r^1(s)$ and $u_r^2(s)$ denoting the proportion of incoming fluid routed to buffers 1 and 2 at time s. Similarly the server allocation (scheduling) control vector, $u^a(s)$ is composed of $(u_a^0(s), u_a^1(s), u_a^2(s))$ where $u_a^0(s)$ refers to the proportion of server capacity spent idle and $u_a^1(s), u_a^2(s)$ denotes the proportion of server capacity dedicated to processing fluid from buffers 1 and 2 at time s. The control vectors (u_r, u_a) are in $\hat{U} := \{(u_r, u_a) \in \mathbb{R}_+^2 \times \mathbb{R}_+^3$ with $\|u_r\|_1 = 1, \|u_a\|_1 = 1\}$. If the fluid level in buffer 1 drops to 0, then the maximum rate of allocation to buffer 1 is bounded by (λ/μ_1). A fluid policy Π_F consists of routing and allocation controls $(u_r, u_a) \in \hat{U}$ for each time point t. For all $t \geq 0$, $\hat{X}(t) = (\hat{Q}(t), \hat{A}(t), \hat{D}(t), \hat{T}(t), \hat{Y}(t))$ defines the fluid model that evolves according to the following dynamics, $\forall t \geq 0,\ k = 1, 2$:

$$\hat{Q}_k(t) = \hat{Q}_k(0) + \hat{A}_k(t) - \mu_k \hat{T}_k(t) \tag{2.7}$$

$$\hat{A}_k(t) = \hat{\Phi}_k(\lambda t) = \int_{s=0}^{t} u_r^k(s) \lambda ds \tag{2.8}$$

$$\hat{T}_1(t) + \hat{T}_2(t) + \hat{Y}(t) = t. \tag{2.9}$$

The effect of controls on fluid dynamics is easier to characterize with time derivatives. Fluid dynamics can be equivalently defined as:

$$\frac{d}{dt}\hat{Q}_1(t) = \lambda u_r^1(t) - \mu_1 u_a^1(t) \tag{2.10}$$

$$\frac{d}{dt}\hat{Q}_2(t) = \lambda u_r^2(t) - \mu_2 u_a^2(t). \tag{2.11}$$

Let x_1, x_2 denote the initial amount of fluid in buffers 1 and 2 at $t = 0$. Let T_k be the first time at which buffer k becomes empty, for $k = 1, 2$. Once a buffer becomes empty, the optimal policy will then keep it empty from then on. The control problem is then choosing actions (u_r, u_a) for each time s before the emptying time $T = \max(T_1, T_2)$. The optimal value for the fluid model can then be defined as follows:

$$V^*(x_1, x_2) = \min_{(u_r,u_a)\in\hat{U}} \int_0^T [c_1\hat{Q}_1(s)+c_2\hat{Q}_2(s)]ds. \tag{2.12}$$

Our solution approach for minimizing (2.12) with respect to (2.10–2.11) includes using Pontryagin maximum principle along with solving Hamilton-Jacobi-Bellman (HJB) equations. The Maximum principle provides necessary but not sufficient conditions for optimality. On the other hand, the HJB equations give sufficient conditions for optimality but they require the knowledge of a value function beforehand. Here we employ a mixed approach. By the Pontryagin maximum principle, the optimization problem can be solved by using a Hamiltonian function H for fixed $t \geq 0$. Formally, H is defined as follows:

$$H(\cdot) = \min_{(u_r,u_a)\in\hat{U}} \left\{ c_1x_1+c_2x_2+p_1(\lambda u_r^1-\mu_1u_a^1)+p_2(\lambda u_r^2-\mu_2u_a^2) \right\},$$

where $p_1(\cdot), p_2(\cdot)$ are Lagrange multipliers. The optimality equation requires that there exist $\eta_1(t), \eta_2(t) \geq 0$ satisfying the following conditions for all $t \geq 0$: $\dot{p_1}(t) = -c_1+\eta_1(t)$, $\dot{p_2}(t) = -c_2+\eta_2(t)$ and complementary slackness conditions $\eta_1(t)\hat{Q}_1(t) = 0$, $\eta_2(t)\hat{Q}_2(t) = 0$. The structure of $\eta_1(\cdot), \eta_2(\cdot)$ is to be determined through the analysis that follows.

Note that these conditions imply that $p_1(t) = -c_1t+p_1(0)$ for $t < T_1$ and $p_2(t) = -c_2t+p_2(0)$ for $t < T_2$ with $p_1(0), p_2(0)$ constants. The optimality equation also implies that $\frac{dH}{d(u_r)} = \lambda(p_1-p_2)$ and $\frac{dH}{d(u_a)} = \lambda(\mu_2p_2-\mu_1p_1)$. $H(\cdot)$

can be equivalently expressed as follows:

$$H(\cdot) = c_1x_1+c_2x_2+\min_{u_r:|u_r|_1=1}\left\{u_r^1p_1+u_r^2p_2\right\}-\max_{u_a:|u_a|_1=1}\left\{\mu_1u_a^1p_1+\mu_2u_a^2p_2\right\}. \tag{2.13}$$

Now by focusing on the allocation problem, we can prove additional properties of the optimal policy.

Proposition 2.3.1. *The optimal allocation controls are non–idling.*

Proof. For simplicity, we drop the time index t. Note that given Lagrange multipliers p_1, p_2, optimal allocation control is selected such that $\max\left\{\mu_1u_a^1p_1 +\mu_2u_a^2p_2\right\}$ is maximized with respect to constraint $u_a^1+u_a^2 \leq 1, u_a^1 \geq 0, u_a^2 \geq 0$. Then by LP theory the optimal vector will be an extreme point of the feasible polyhedral set, implying that the constraint $u_a^1+u_a^2 = 1$ will always hold for optimal allocation vector u_a^*. Since $u_a \in \hat{U}$, by definition $|u_a| \leq 1$ thus $u_a^{*0} = 0$. Therefore, the optimal policy is non–idling. □

Proposition 2.3.2. *Under the optimal policy, $T_1 \leq T_2$.*

Proof. We prove by contradiction. Suppose $T_2 > T_1$ then $\hat{Q}_1(T_2) > 0, \hat{Q}_2(T_2) = 0$. For $t \in (T_2, T_1)$, the optimal policy would require that the server dedicates its full capacity to process fluid from buffer 1, as the other buffer is empty. As a result, $p_1(T_2)\mu_1 = p_2(T_2)\mu_2$. Yet for $t < T_2$, $\dot{p}_1(t) = -c_1$ and $\dot{p}_2(t) = -c_2$ with $-c_1 < -c_2$. It then follows that $p_1(t)\mu_1 > p_2(t)\mu_2$ for $t < T_2$ implying that it should be optimal to process fluid from first buffer rather than the second for all previous time points. This would pose a contradiction to original assumption of $T_2 < T_1$. $\square$

For all $t \geq 0$ by replacing $u_r^2(t) = 1-u_r^1$, $u_a^2(t) = 1-u_a^1(t)$, in (2.13) we can further characterize the optimal controls depending on the values of Lagrange multipliers. Note that given $u_r^1(\cdot), u_a^1(\cdot)$ we can completely determine optimal policy, therefore we only note the conditions on these controls in the sequel. In order to derive the optimal routing controls, we only need to compare $p_1^*(t)$ to $p_2^*(t)$ for all time $t \geq 0$ as the comparison is mapped to the optimal control decision as follows:

$$u_r^{1*}(t) = \begin{cases} 1 & p_1(t) < p_2(t) \\ 0 & p_1(t) > p_2(t) \\ ? & p_1(t) = p_2(t). \end{cases}$$

Given Lagrange multipliers at any time $t \geq 0$, the optimal routing decision can be easily determined for the case where $p_1^*(t) \neq p_2^*(t)$. However, when $p_1(t) = p_2(t)$, $u_r^{1*}(t)$ can take any value in $(0,1)$. It is important to note that $(p_1(\cdot)-p_2(\cdot))$ can be 0 at most once. This can be proved as follows: let $t \in (T_1, T_2)$ with $p_1(t) = p_2(t)$. By the definition of draining times, $\hat{Q}_1(t) = 0$ for $t > T_1$. Also, it is clear that under the optimal policy after time $t > T_1$ the fluid from buffer 2 must be drained. Consider time $t \in (T_1, T_2)$. If $p_1(t) = p_2(t)$, then $p_1(t)\mu_1 > p_1(t)\mu_2$ and thus all processing priority is given to class 1. If this is the case, the server can only use up to a capacity of (λ/μ_1) which would contradict with Proposition 2.3.1. Therefore, $(p_1(\cdot)-p_2(\cdot))$ can be 0 at most once implying that optimal routing control would change at most once. Next, the allocation (scheduling) decision can be derived for all time $t \geq 0$ as follows:

$$u_a^{1*}(t) = \begin{cases} \mathbb{1}_{\{\hat{Q}_1(t)>0\}} + \frac{\lambda u_a^{1*}(t)}{\mu_1} \mathbb{1}_{\{\hat{Q}_1(t)=0\}} & p_1(t)\mu_1 < p_2(t)\mu_2 \\ 0 & p_1(t)\mu_1 > p_2(t)\mu_2 \\ ? & p_1(t)\mu_1 = p_2(t)\mu_2. \end{cases}$$

Following the same lines as previous argument, as $-c_1\mu_1 \neq -c_2\mu_2$ and $T_1 \leq T_2$, $(p_1(\cdot)\mu_1 - p_2(\cdot)\mu_2)$ can only hit 0 once. Note that through the link via Lagrange multipliers, optimal routing controls have implications on the optimality of particular allocation controls. For example, if $u_r^{1*} = 0$ then $(p_1(\cdot) > p_2(\cdot))$ which in turn implies $(\mu_1 p_1(\cdot) > \mu_2 p_2(\cdot))$, resulting in $u_a^{1*} = 1$. Also note that since $p_1(T_1)\mu_1 = p_2(T_1)$ it implies that $p_1(t)\mu_1 > p_2(t)\mu_2$ for $t < T_1$. Therefore, for any $t \geq 0$, if $\hat{Q}_1(t) > 0$ then $u_a^{2*}(t) = 0$.

In addition, feasible allocation controls can depend on the current state. Consider a time point $t \geq 0$ where $\hat{Q}_1(t) = 0, \hat{Q}_2(t) > 0$. As $\lambda > \mu_2$, selecting control pairs $u_r^1(t) = 0, u_a^1(t) = 0$ would result in $\frac{d}{dt}\hat{Q}(t) \geq 0$ therefore these pairs can not be selected under the optimal policy. At such a time point, in order to drain the fluid levels, the controls must be set to $u_r^{1*} = 1, u_a^{1*} = \frac{\lambda}{\mu_1}$ in order to dedicate the capacity of the server to processing fluid from buffer 2. For any state $\hat{Q}(t)$ the possible control pairs are then given as follows:

I. $u_r^{1*} = 0, u_a^{1*} = 1$	*II.* $u_r^{1*} = 1, u_a^{1*} = \mathbb{1}_{\{\hat{Q}_1(t)>0\}} + \frac{\lambda}{\mu_1}\mathbb{1}_{\{\hat{Q}_1(t)=0\}}$
$\frac{d}{dt}\hat{Q}_1(t) = -\mu_1;$	$\frac{d}{dt}\hat{Q}_1(t) = -(\mu_1(\mathbb{1}_{\{\hat{Q}_1(t)>0\}} + \frac{\lambda}{\mu_1}\mathbb{1}_{\{\hat{Q}_1(t)=0\}}) - \lambda);$
$\frac{d}{dt}\hat{Q}_2(t) = \lambda;.$	$\frac{d}{dt}\hat{Q}_2(t) = \mu_2(1 - \frac{\lambda}{\mu_1})\mathbb{1}_{\{\hat{Q}_1(t)=0\}};$

Note that action control pair $(I.)$ is only feasible when $\hat{Q}_1(t) > 0$ for $t \geq 0$. Also note that only action control pair $(II.)$ can drain buffer 2. For that reason, this control must be surely employed given any initial state. As a side note, this observation sheds light on the structure of $\eta_1(\cdot), \eta_2(\cdot)$. For $t \in (T_1, T_2)$, $\eta_1(t) = c_1 - \frac{c_2\mu_2}{\mu_1}$ and $\eta_2(t) = 0$.

Also note that it is possible to have a trajectory where the controls in $(II.)$ follow the ones in $(I.)$. To see this, consider time $t < T_1$ with $p_1(t) < p_2^*(t)$. Since $\dot{p}_1(t) < \dot{p}_2(t)$, it is possible that it exists a time $t' < t$ with $p_1(t') = p_2(t')$ and for all $s < t'$, $p_1(s) > p_2(s)$. As a result, the progression of the optimal controls are as follows:

$$\left(u_r^{1*} = 0, u_a^{1*} = 1\right) \Rightarrow \left(u_r^{1*} = 1, u_a^{1*} = 1\right) \Rightarrow \left(u_r^{1*} = 1, u_a^{1*} = \frac{\lambda}{\mu_1}\right).$$

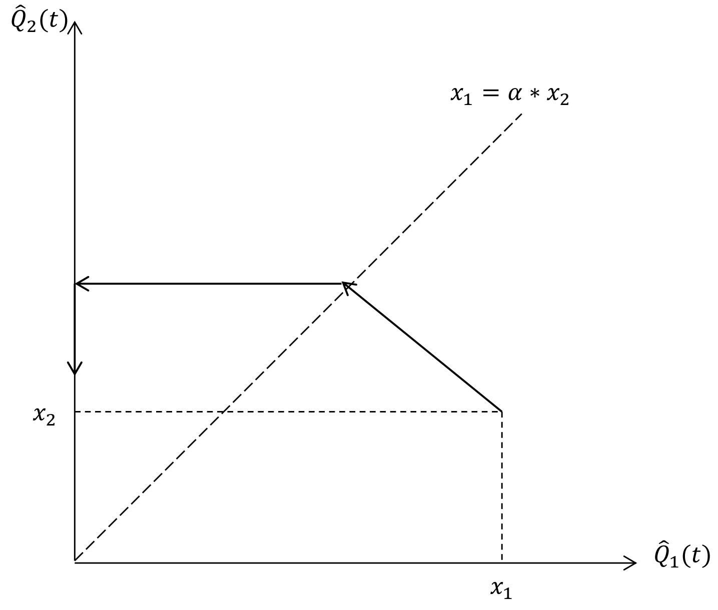


Figure 2.2: Progression of the optimal fluid policy

So far we used the maximum principle and Lagrange multipliers to obtain the progression of optimal fluid policy. In order to determine explicitly the switching policy, one approach would be to solve HJB equations using a prudent guess of the value function. If the resulting fluid solution satisfies the HJB equations then the guess of the value function is correct and obtained policy is optimal. Note that the optimal value function $V(x_1, x_2)$ has a piecewise quadratic form before the routing switch occurs, and after that it is purely quadratic. This form of $V(\cdot)$ indicates the presence of a linear optimal switching policy for routing control where $u_r^* = 1$ if $(\alpha\hat{Q}_1(t)–\hat{Q}_2(t) < 0)$ and 0 otherwise. Assume that initial fluid levels (x_1, x_2) satisfy $(\alpha x_1–x_2 < 0)$ such that a routing switch does not occur for $t \geq 0$. Then for such a pair (x_1, x_2), the total cost $V(x_1,x_2)$ is as follows:

$$V(x_1,x_2) \;=\; \frac{c_1x_1^2}{2(\mu_1-\lambda)}+\frac{c_2x_1x_2}{(\mu_1-\lambda)}+\frac{c_2x_2^2}{2\mu_2(1-\frac{\lambda}{\mu_1})}. \tag{2.14}$$

The HJB equation implies the following conditions hold for all time $t \geq 0$:

$$\begin{aligned}
&\Rightarrow \min_{u_r^1,u_a^1}(\lambda u_r^1(t)-\mu_1u_a^1(t))\frac{dV}{dx_1}+(\lambda(1-u_r^1(t))-\mu_2(1-u_a^1(t)))\frac{dV}{dx_2}+c_1x_1+c_2x_2 &= 0\\
&\Rightarrow \big(\min_{u_r^1}\lambda u_r^1(t)(\frac{dV}{dx_1}-\frac{dV}{dx_2})\big)+\big(\min_{u_a^1}(u_a^1(t)(\mu_2\frac{dV}{dx_1}-\mu_1\frac{dV}{dx_2}))+c_1x_1+c_2x_2 &= 0.
\end{aligned}$$

The optimal controls are bang–bang: $\big\{\frac{dV}{dx_1} < (>)\frac{dV}{dx_2}\big\}$ implies $u_r^{*1} = 1(0)$ and $\big\{\mu_2\frac{dV}{dx_1} < (>)\mu_1\frac{dV}{dx_2}\big\}$ implies $u_a^{*1} = 1(0)$. For the routing control, the optimal slope parameter α is then obtained by setting $\frac{dV}{dx_1} = \frac{dV}{dx_2}$:

$$\frac{(c_1-c_2)\mu_2}{c_2(\mu_1-\mu_2)}x_1 = x_2 \Rightarrow \alpha = \frac{(c_1-c_2)\mu_2}{c_2(\mu_1-\mu_2)}.$$

For allocation control, note that $\big\{\frac{dV}{dx_1} < \frac{dV}{dx_2}\big\} \Rightarrow \big\{\mu_2\frac{dV}{dx_1} < \mu_1\frac{dV}{dx_2}\big\}$, given $\mu_2 < \mu_1$. Therefore it is optimal to dedicate the server to buffer 1, as long as there is fluid in buffer 1. Note when buffer 1 is drained, optimal control has to dedicate a portion of the server $(\frac{\lambda}{\mu_1})$ to buffer 1 in order to keep it empty. This is coherent with our structural results in Section 2.2 although the fluid policy is composed without any such enforcement.

To summarize, the fluid optimal controls suggest that it is always optimal to process fluid from buffer 1, as long as the buffer is not empty. And for routing, it is optimal to route incoming fluid to buffer 2 iff $\alpha\hat{Q}_1(t) > \hat{Q}_2(t)$. In terms of the original processes, the optimal fluid policy satisfies the following equations:

$$\hat{Y} \text{ cannot increase when } \hat{Q}_1(t) + \hat{Q}_2(t) > 0 \tag{2.15}$$

$$\hat{T}_2(t) \text{ cannot increase when } \hat{Q}_1(t) > 0 \tag{2.16}$$

$$\hat{A}_1(t) \text{ cannot increase when } \alpha\hat{Q}_1(t) - \hat{Q}_2(t) > 0 \tag{2.17}$$

$$\hat{A}_2(t) \text{ cannot increase when } \alpha\hat{Q}_1(t) - \hat{Q}_2(t) \leq 0. \tag{2.18}$$

Any fluid policy Π_F for which $\hat{X}$ satisfies (2.7)–(2.9) and (2.15) –(2.18) is then an optimal policy.

2.4 Perturbation expansions

Asymptotic and singular perturbation techniques can be used to gain more insight into the qualitative structure of a model. In this section, the goal is to improve our approximation to the discrete stochastic network through perturbation expansions. Empirical observations of the switching curve (see Section 2.6) indicate that a purely linear function does represent the optimal policy as well as an affine function with a positive x_1 intercept. This "off-set" value can be approximated by following the perturbation-based approach described in Avram [4].

The main idea of our approach here is to perturb the quadratic fluid value function by adding a term that is linear in x_2 and solve the HJB equation along with the boundary conditions. Given any initial buffer level $x = (x_1, x_2)$, the draining time of buffer 2 under Π_F (the optimal policy for the fluid model) is always greater than the draining time of buffer 1. Thus, we focus on the system after buffer 1 is drained, with an initial starting condition $x_1 = 0$. Under Π_F all fluid is routed to buffer 1 while the server allocates λ/μ_1 proportion of its time to class 1 fluid and the remainder to class 2 fluid. The essential idea of the perturbation approach is scale modification. Let $\epsilon^{-1}\tilde{x}_i = x_i, i = 1, 2$ and $\tilde{v}(\tilde{x}_1, \tilde{x}_2) = \epsilon^{-2}V(x_1, x_2)$. As a result, the HJB equation (now given in terms of time differences) can be expressed as follows:

$$0 = \min_{u_r,u_a}[\lambda u_r \Delta_{a_1}\tilde{v} + \lambda(1-u_r)\Delta_{a_2}\tilde{v} + u_a\mu_1\Delta_{d_1}\tilde{v} + (1-u_a)\mu_2\Delta_{d_2}\tilde{v}] + c_1\tilde{x}_1 + c_2\tilde{x}_2.$$

$\Delta_{a_1}\tilde{v}, \Delta_{a_2}\tilde{v}, \Delta_{d_1}\tilde{v}, \Delta_{d_2}\tilde{v}$ are respectively difference operators associated with the arrival to buffer 1, arrival to buffer 2, the departure from buffer 1 and the departure from buffer 2 processes. Formally these operators are defined as follows: $\Delta_{a_1}\tilde{v} = \tilde{v}(\tilde{x}_1+1, \tilde{x}_2) - \tilde{v}(\tilde{x}_1, \tilde{x}_2), \Delta_{a_2}\tilde{v} = \tilde{v}(\tilde{x}_1, \tilde{x}_2+1) - \tilde{v}(\tilde{x}_1, \tilde{x}_2), \Delta_{d_1}\tilde{v} = \tilde{v}(\tilde{x}_1, \tilde{x}_2) - \tilde{v}(\tilde{x}_1-1, \tilde{x}_2)$ and $\Delta_{d_2}\tilde{v} = \tilde{v}(\tilde{x}_1, \tilde{x}_2) - \tilde{v}(\tilde{x}_1, \tilde{x}_2-1)$. Note that we are interested in the case when $x_1 = 0$, therefore the optimal controls are $u_r^* = 1$ and $u_a^* = \lambda/\mu_1$. By expanding the difference operators up to the second order, the boundary conditions on x_1 and the HJB can be equivalently expressed as follows:

$$\begin{aligned} 0 &= \lambda\left(\frac{\partial V}{\partial x_1}\epsilon + \frac{\partial^2 V}{\partial^2 x_1}\frac{\epsilon^2}{2}\right) + \lambda\left(-\frac{\partial V}{\partial x_1}\epsilon + \frac{\partial^2 V}{\partial^2 x_1}\frac{\epsilon^2}{2}\right) \\ &\quad + \left(1 - \frac{\lambda}{\mu_1}\right)\mu_2\left(-\frac{\partial V}{\partial x_2}\epsilon + \frac{\partial^2 V}{\partial^2 x_2}\frac{\epsilon^2}{2}\right) + \epsilon(c_1x_1 + c_2x_2), \end{aligned}$$

with $\frac{\partial V}{\partial x_1} = 0$. Now the last step is setting $V = V_1 + \epsilon V_2$, where V_1 corresponds to the original fluid quadratic and V_2 corresponds to the linear perturbation

(lx_2). As a result the optimality equations can be expressed as:

$$-\left(1-\frac{\lambda}{\mu_1}\right)\mu_2\frac{\partial V_1}{\partial x_2}+c_1x_1+c_2x_2 \;=\; 0 \qquad (2.19)$$

$$-\mu_2\frac{\partial V_2}{\partial x_2}+2\lambda\frac{\partial V_1}{\partial^2 x_1}+\mu_2\frac{\partial V_1}{\partial^2 x_2} \;=\; 0. \qquad (2.20)$$

Note that first equation above is a first-order differential equation. We obtain $\frac{\partial V_1}{\partial^2 x_2}$ from (2.19). Next, we derive $\frac{\partial V_1}{\partial^2 x_1}$ from the original fluid quadratic equation (2.14). We then substitute these into (2.20) to yield the linear correction term:

$$\frac{c_1\lambda}{(\mu_1-\lambda)\mu_2}+\frac{c_2\mu_2}{(\mu_1-\lambda)} \;=\frac{\partial V_2}{\partial x_2} \;= l.$$

The fluid correction, l can be interpreted as adding certain constant corrections throughout the emptying time of x_1 (since x_1 empties earlier than x_2). As a result V is then $(V_1+\epsilon l x_2)$. Similar to computing the fluid slope, setting $\frac{\partial V}{\partial x_1}=\frac{\partial V}{\partial x_2}$ produces the following linear switching policy:

$$x_2 \;=\; \alpha(x_1-\tilde{o}),$$

with α defined as in the previous section, i.e., $\alpha=\frac{(c_1-c_2)\mu_2}{(\mu_1-\mu_2)c_2}$. We then obtain $\tilde{o}=\frac{c_1(2\lambda)+c_2\mu_1}{2\mu_2(c_1-c_2)}$ as the best offset among the possible linear approximations.

2.5 Asymptotics

Translation of the optimal fluid policy into an implementable discrete policy requires some care. Naive translations may cause instability (as in the case of Rybko-Stolyar network, see Maglaras [25]). Therefore it is important to analyze the asymptotic behavior of the proposed policy. We investigate asymptotics under a fluid scaling in which both the initial condition and time horizon are scaled up. Fluid-scale asymptotic optimality is used to characterize the validity of the fluid approximation in the limiting regime.

For a queueing network process $\mathbb{X}$, we define the fluid scaling as $\bar{\mathbb{X}} = r^{-1}\mathbb{X}(rt)$, for $r \geq 0$. If the queueing network process depends on r then we write

$$\bar{\mathbb{X}}^r = r^{-1}\mathbb{X}^r(rt), \tag{2.21}$$

where $\mathbb{X}^r$ is the process associated with the rth queueing network. If

$$\bar{\mathbb{X}}^N \to \bar{\mathbb{X}} \text{ a.s. u.o.c. (uniformly on compact sets).}$$

as $r \to \infty$, then the process $\bar{\mathbb{X}}$ is then called a fluid limit. Each component of the fluid limit is absolutely continuous and thus differentiable almost everywhere in $[0, \infty)$. Let $\bar{\mathbb{X}}(\omega)$ denote the set of fluid limits associated with a sample path ω. Each fluid limit satisfies fluid model dynamics (2.7) - (2.9), and furthermore the properties of the fluid limit helps us characterize stability and asymptotic optimality of the discrete network. The fluid model said to be *stable* if there exists a fixed time $\sigma > 0$ such that $\bar{\mathbb{X}}(t) = 0$, for all $t \geq \sigma$ for any fluid solution.

Definition 2.5.1. For a given state $x \in S$, let $q_k^r(t;x) = r^{-1}Q_k(rt;[rx])$ for $k = 1, 2$, $t \in \mathbb{R}_+$ and $V_r(x,T) = \mathbb{E}[\int_0^T (c_1 q_r^1(t;x) + c_2 q_r^2(t;x))dt]$, for $T \in \mathbb{R}_+$. A policy is called fluid scale asymptotically optimal, in the total cost sense, if

$$\limsup_{n\to\infty} V_r(x,T) \leq V_*(x) \text{ for } x \in S, T \leq 0$$

where $V_*(x)$ denotes the optimal value function for the fluid model.

In order to verify fluid scale asymptotic optimality for a given policy Π, one can take the fluid limit of $\bar{\mathbb{X}}$ under the policy and check whether the optimal fluid dynamics are satisfied. If so, then policy Π is asymptotically optimal in the total cost sense.

Definition 2.5.2. Let $S = \{q = (i,j) \in \mathbb{Z}_+^2 : \alpha(i - o')-j \leq 0\}$, with α and $o' = h(\tilde{o})$ where h is a function $h : \mathbb{R} \to \mathbb{Z}$ and α, $\tilde{o}$ defined as in Section 4. Recall a state of the Markov decision process is defined as (i,j,C). Then given (i,j,C), the stationary policy Π_D is then defined as follows:

- $(i,j) \notin S'$
  - if $C = C1$ then $U^r = R2_{NP}$, $U^a = A1$
  - if $C = C2$ then $U^r = R2_P$, $U^a = A1$
- $(i,j) \in S'$ then
  - if $C = C1$ & $i > 0$ then $U^r = R1_P$, $U^a = A1$
  - if $C = C2$ & $i = 0$, $j > 0$ then $U^r = R1_{NP}$, $U^a = A2$.

If the system is empty, i.e., $C = C_0$, then Π_D routes the next arrival to Queue 1.

Proposition 2.5.1. *Let stationary policy Π_D be defined as above. Then any fluid limit $\bar{\mathbb{X}}(t)$ satisfies fluid the optimality-feasibility equations with stationary* *policy Π_D. Therefore Π_D is an asymptotically optimal policy in the total cost sense.*

Proof. By definition the fluid limit $\bar{\mathbb{X}}(t)$, satisfies the fluid feasibility equations (2.7)-(2.9). If the fluid limit satisfies following optimal fluid model equations, then this implies asymptotic optimality. We prove that the following constraints hold at all time $t \geq 0$:

$$\bar{A}_1(t) \text{ cannot increase when } \alpha\bar{Q}_1(t) - \bar{Q}_2(t) > 0 \tag{2.22}$$

$$\bar{A}_2(t) \text{ cannot increase when } \alpha\bar{Q}_1(t) - \bar{Q}_2(t) \leq 0 \tag{2.23}$$

$$\bar{Y}(t) \text{ cannot increase when } \bar{Q}_1(t) + \bar{Q}_2(t) > 0 \tag{2.24}$$

$$\bar{T}_2(t) \text{ cannot increase when } \bar{Q}_1(t) > 0. \tag{2.25}$$

Let $t > 0$. Assume that $\alpha\bar{Q}_1(t) - \bar{Q}_2(t) > 0$. Since $\bar{Q}$ is a fluid limit, there exists a sample path ω and a sequence $r_n \to \infty$ such that:

$$(\bar{Q}^{r_n}(\cdot, \omega), \bar{A}^{r_n}(\cdot, \omega)) \to (\bar{Q}, \bar{A}) \text{ u.o.c.}$$

as $n \to \infty$. This suggest that $\exists \epsilon > 0$ and integer N such that $\alpha\bar{Q}_1^{r_n}(t, \omega) - \bar{Q}_2^{r_n}(t, \omega) \geq \epsilon$ for $n \geq N$. As a result, $\alpha Q_1^{r_n}(r_n t, \omega) - Q_2^{r_n}(r_n t, \omega) > r_n \epsilon$ for $n \geq N$. Consider another integer $N_2 \geq N$ and $\epsilon' \in (0, \epsilon)$ such that $r_n \epsilon \geq r_n(\alpha+1)\epsilon' + \alpha o'$ for $n \geq N_2$. Hence $\alpha Q_1^{r_n}(r_n t, \omega) - Q_2^{r_n}(r_n t, \omega) > r_n(\alpha+1)\epsilon' + \alpha o'$ for $n \geq N_2$. It then follows that:

$$\alpha(Q_1^{r_n}(r_n t, \omega) - r_n \epsilon') > Q_2^{r_n}(r_n t, \omega) + r_n \epsilon' + \alpha o',$$

for $n \geq N_2$. Now note that each component of $\mathbb{X}$ is Lipschitz continuous. This suggests that under any policy, jumps of the corresponding Markov chain are bounded. Considering the uniformized Markov chain, we then have:

$$|Q(t)–Q(t+1)| \leq 1,$$

for $t \geq 0$. For the fluid limit $\bar{Q}$, this implies that:

$$|\bar{Q}(t)–\bar{Q}(t+s)| \leq |t - s|,$$

for $t, s \geq 0$. It then follows that:

$$|Q_1^{r_n}(r_n t, \omega)–Q_1^{r_n}(r_n(t+\delta), \omega)| \leq r_n \delta$$

$$|Q_2^{r_n}(r_n t, \omega)–Q_2^{r_n}(r_n(t+\delta), \omega)| \leq r_n \delta,$$

for $\delta \geq 0$. Now let $\delta \leq \epsilon'$. We then have:

$$\alpha Q_1^{r_n}(r_n(t+\delta), \omega) > Q_2^{r_n}(r_n(t+\delta), \omega)+\alpha o',$$

for $n \geq N_2$. Therefore, $A_1^{r_n}(s, \omega)$ is flat for $s \in (r_n t, r_n(t+\delta))$ and $n \geq N_2$. Equivalently, $\bar{A}_1(s, \omega)$ is flat for $s \in (t, t+\delta)$. Letting $n \to \infty$, we have $\bar{A}_1(s)$ is flat for $s \in (t, t+\delta)$, and thus (2.22) is proved. The proof of (2.23) follows the same lines.

Now we move on to the proof of (2.24). Similarly, let $t \geq 0$ where $\bar{Q}_1(t)$ $+\bar{Q}_2(t) \geq 0$. By the continuity of $\bar{Q}$, $\exists \delta > 0$ such that $\min_{s \in (t-\delta, t+\delta)} \bar{Q}_1(t)+\bar{Q}_2(t) > 0$. There exists a sample path ω and a sequence $r_n \to \infty$ such that:

$$(\bar{Q}^{r_n}(\cdot, \omega), \bar{Y}^{r_n}(\cdot, \omega)) \to (\bar{Q}, \bar{Y}) \text{ u.o.c.}$$

as $n \to \infty$. It then follows that there exists integer N such that:

$$\inf_{s\in(t-\delta,t+\delta)} \bar{Q}_1^{r_n}(s,\omega)+\bar{Q}_2^{r_n}(s,\omega) \geq 0,$$

for $n \geq N$. Therefore,$Q_1^{r_n}(s,\omega)+Q_2^{r_n}(s,\omega) \geq 0$ for $s \in (r_n(t-\delta),(t+\delta))$ and $n \geq N$. Given that the service discipline under Π_D is non-idling, $Y^{r_n}(s,\omega)$ is flat for $s \in (r_n(t-\delta), r_n(t+\delta))$ and $n \geq N$. It then implies that $\bar{Y}^{r_n}(s,\omega)$ is flat for $s \in ((t-\delta), r_n(t+\delta))$. Let $n \to \infty$, we have $\bar{Y}(s)$ is flat for $s \in (t-\delta, t+\delta)$, proving (2.24).

Finally, we prove (2.24). Following the same steps with the previous statements, again let $t \geq 0$ where $\bar{Q}_1(t) > 0$. Then by the continuity of $\bar{Q}_1$, $\exists\delta > 0$ such that $\min_{s\in(t-\delta,t+\delta)} \bar{Q}_1(t) > 0$. Since any component of $\bar{X}$ is a fluid limit, there exists a sample path ω and a sequence $r_n \to \infty$ such that:

$$(\bar{Q}^{r_n}(\cdot,\omega), \bar{T}^{r_n}(\cdot,\omega)) \to (\bar{Q},\bar{T}) \text{ u.o.c.}$$

as $n \to \infty$. There exists integer N such that $\inf_{s\in(t-\delta,t+\delta)} \bar{Q}_1^{r_n}(s,\omega) \geq 0$ for $n \geq N$. Therefore, $Q_1^{r_n}(s,\omega) \geq 0$ for $s \in (r_n(t-\delta), r_n(t+\delta))$ and $n \geq N$. Note that under Π_D, the server gives priority to class 1 jobs. Hence, when $n \geq N$, $T_2^{r_n}(s,\omega)$ is flat for $s \in (r_n(t-\delta), r_n(t+\delta))$. It then follows that $\bar{T}_2^{r_n}(s,\omega)$ is flat for $s \in ((t-\delta),(t+\delta))$. Letting $n \to \infty$, we have $\bar{T}_2(s)$ is flat for $s \in (t-\delta, t+\delta)$, proving (2.22) and completing the proof. □ □

In this section we proved that the policy Π_D is asymptotically optimal only in the total cost sense. Although in our setting our goal is to minimize long-run average cost rate, there is no well-established theory connecting optimization of total cost in the fluid model with optimizing average cost in the discrete queueing network. The average cost optimal policy is in fact an asymptotically optimal policy in the total cost sense [28], therefore by this construction we aim to obtain a good approximation to this policy. In the next section, we evaluate the performance of the proposed discrete policy Π_D with respect to the average cost optimal policy obtained through policy iteration.

2.6 Numerical study

In this section, we provide examine the optimal policy and evaluate the performance of the asymptotically optional policy Π_D in several sample networks. The optimal discrete policy is obtained through policy iteration, on a truncated state space. Specifically, each class can have at most B customers. For large values of B, computing the optimal policy is increasingly intractable and this is a primary motivation to examine approximately optimal policies. The proposed policy Π_D has two advantages. First, it has a simple structure defined by a linear threshold, and is thus easy to implement. Second, if the proposed policy performs well then it can be used as an initial policy in *policy iteration* (if the decision-maker still requires an optimal solution) resulting in a lower number of iterations and thus less computational time. Note that the optimal allocation policy we compute in the truncated network does indeed conform to the $c\mu$ rule, confirming the structural results in Section 2.2.

Before assessing the performance of Π_D, we provide additional motivation behind this policy via visual presentations of selected average-cost optimal policies. Recall that the state definition includes the class of the job currently in service. In this section, routing policies are presented for the case when the system serves a customer of class 1. The case where the current customer is of class 2 admits similar routing controls. Figure 2.3 presents the optimal routing policy for different values of c_1, which is the holding cost for class 2 customers. The policies depicted indicate that a linear switching curve is a good approximation to the true switching curve. Also notice that increasing values of c_1 yield steeper slopes, which is consistent with the fluid slope computed in Section 2.3. For this reason, in the discussion below we only focus on comparison of routing policies.

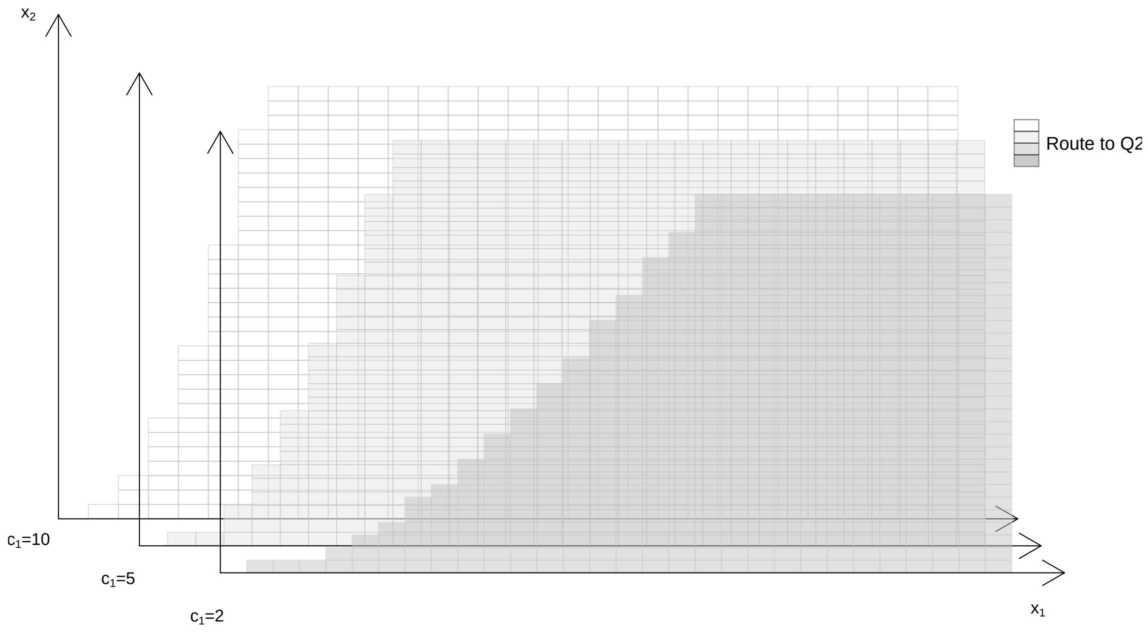


Figure 2.3: Routing policies with increasing slope : $c_2 = 1, \lambda = 0.473, \mu_1 = 0.5264, \mu_2 = 0.402$

Similar motivation can be found for the presence of an offset on the x_1 axis, corresponding to the number of class 1 jobs. Three different policies with similar curve but different offsets are presented in Figure 2.4. The offset computed via a perturbation expansion in Section 2.4 is non-increasing in μ_2 which is again consistent with the policies shown in Figure 2.4.

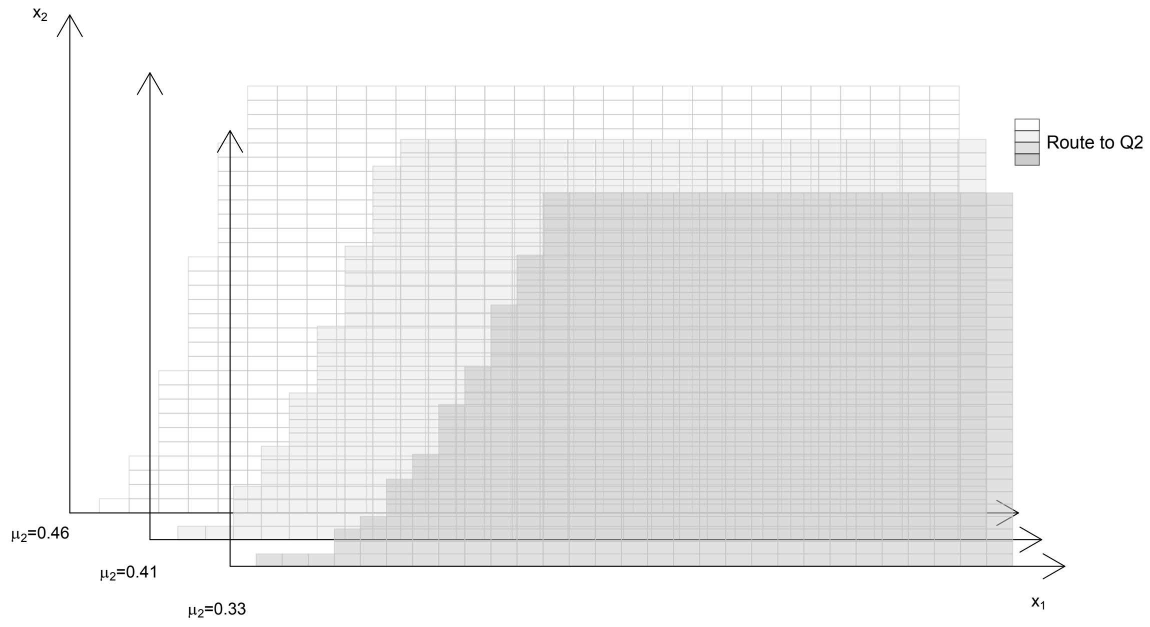


Figure 2.4: Routing policies with increasing offset : $c_1 = 10, c_2 = 1, \lambda = 0.485, \mu_1 = 0.515$

The performance of Π_D is assessed as follows: For a given set of parameters $(\lambda, \mu_1, \mu_2, c_1, c_2)$, the optimal average cost , C_{Opt} is obtained through policy iteration. We then evaluate the average cost of Π_D (C_D) and the optimality gap of Π_D is defined as: $\frac{C_D - C_{Opt}}{C_{Opt}}$. Both policies are evaluated in the system with a truncated state space.

Recall that we are interested in parameters that satisfy $\lambda > \mu_2$ and $\lambda < \mu_1$. We denote $\gamma = \mu_2/\lambda$ and $\rho = \lambda/\mu_1$. For simplicity we fix $(\lambda + \mu_1 = 1)$ and vary ρ, γ, c_1, c_2. Fixing the truncation parameter at $B = 50$, instances with the following combinations of parameters are evaluated:

$$\begin{aligned}\rho \in &\{0.95, 0.9, 0.85, 0.8, 0.75, 0.7, 0.65\}\\ \gamma \in &\{0.95, 0.9, 0.85, 0.8, 0.75, 0.7, 0.65\}.\end{aligned}$$

Recall that Π_D requires an integer term for the offset and function $h : \mathbb{R}_+ \to \mathbb{Z}_+$ that maps the offset value obtained through perturbation expansions (see Section 2.4) to an integer. First we perform an empirical evaluation of some intuitive mappings with respect to the optimal average cost policy. Three different settings are evaluated. In Setting 1, $\tilde{o}$ is rounded up to the nearest integer ($h(\tilde{o}) = \lceil \tilde{o} \rceil$). In Setting 2, $\tilde{o}$ is rounded down to the nearest integer ($h(\tilde{o}) = \lceil \tilde{o} \rceil$) and in Setting 3, we set $h(\tilde{o}) = 0$ to compare policies with and without an offset term. The instances for this set of experiments are generated with ρ and γ combinations defined as above and for $(c_1, c_2) = (10, 1)$. Figure 2.5 provides a comparison on the performance of settings 1, 2 and 3.

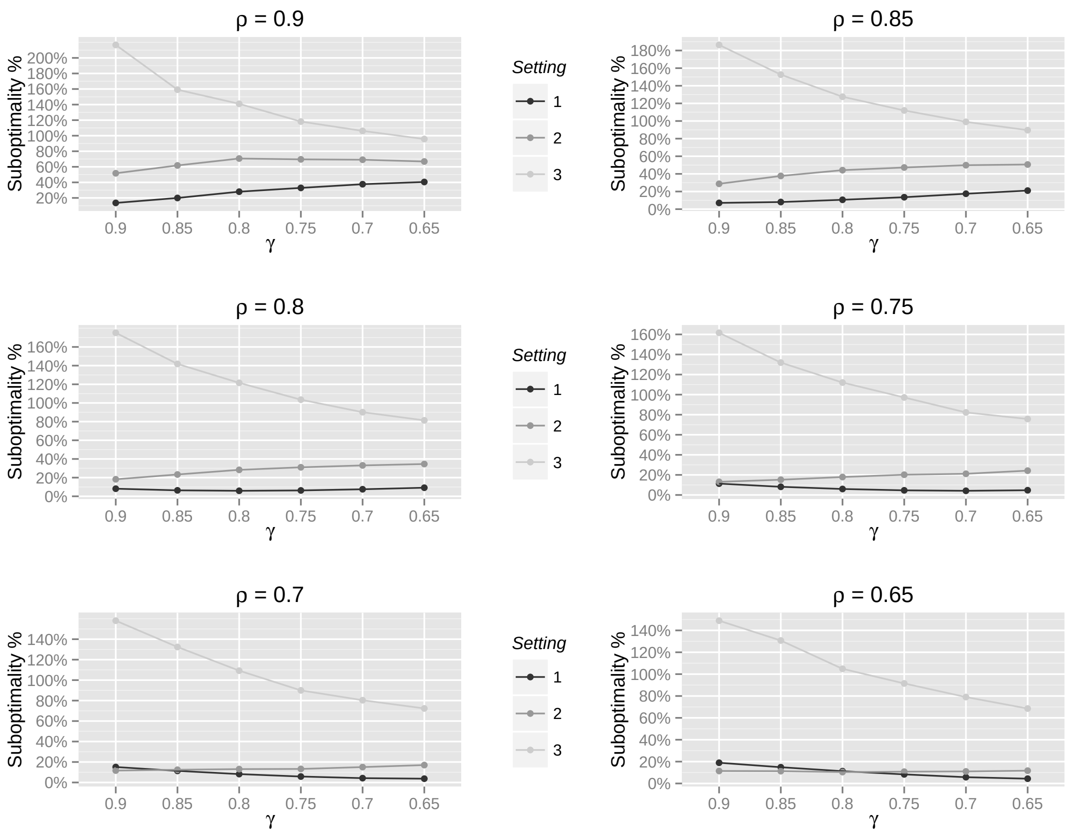


Figure 2.5: Performance of offset settings under different parameter settings

Across all instances, Figure 2.5 shows the significant benefit of using an offset term, as not doing so results in optimality gaps between 80% to 100%. It is observed that setting 1 outperforms setting 2 for all cases with the exception of a few with $\rho < 0.75$. As γ decreases we observe that setting 1 always yields better results than setting 2. These observations indicate that setting 1should be used in the definition of $h(\tilde{o})$. The next set of experiments evaluate the performance of the corresponding discrete policy Π_D in more detail. In addition to experiments on γ and ρ, this set experiments also aims to measure the effect of c_1 and c_2. Choosing $c_1 \in \{2, 5, 10\}$, $c_2 \in \{1\}$, we perform experiments on various combinations of c_1, c_2, ρ, γ. Figure 2.6 shows the performance of Π_D with respect to the average-cost optimal policy.

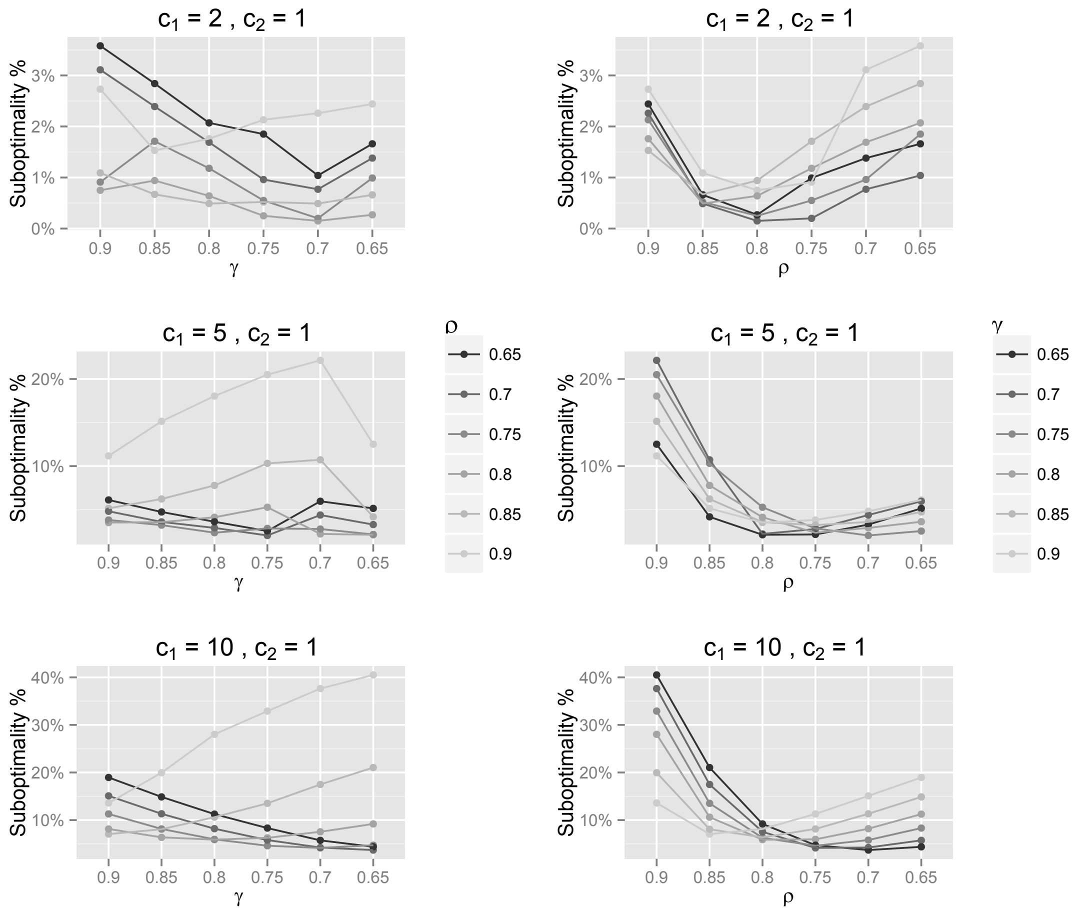
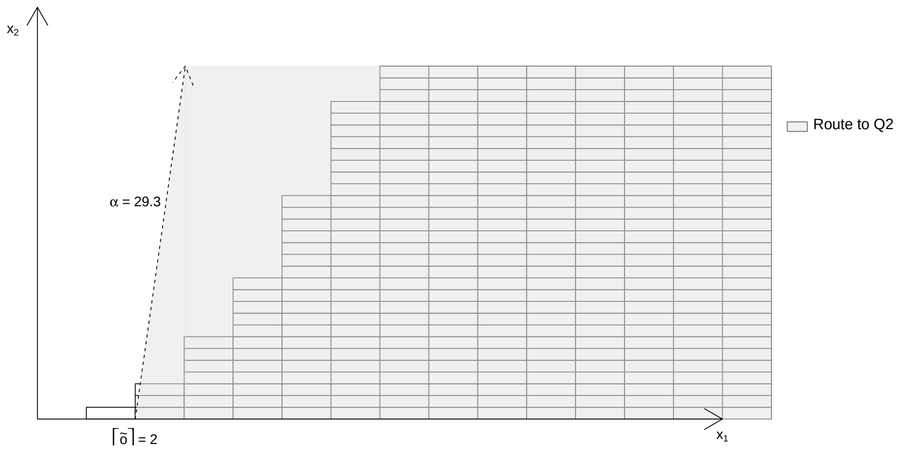


Figure 2.6: Performance of proposed policy under different parameter settings

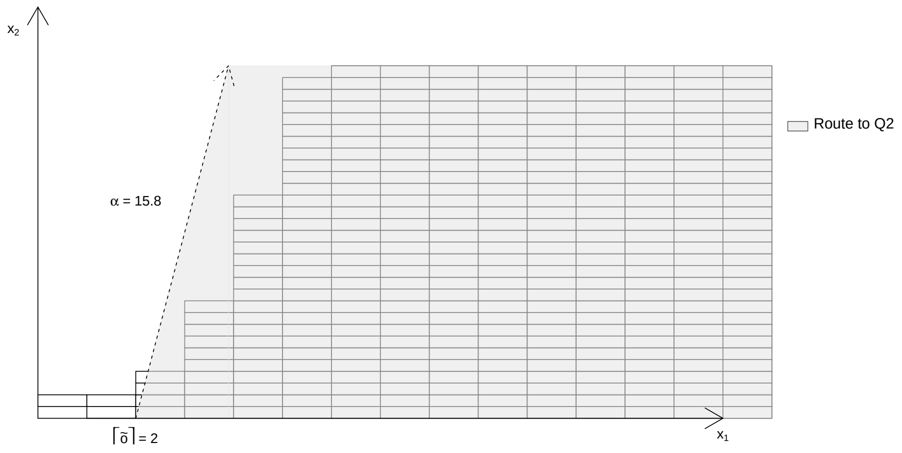
First, we observe that as c_1 approaches c_2 the optimality gap of Π_D is less than 5% under all different settings of ρ and γ. Although different trends seem to be present for $c_1 = 2$, the magnitude of the changes are very small compared to the cases with $c_1 \in (5, 10)$. The difference in average cost is expected to increase when $c_1 >> c_2$ since then one suboptimal action has a significant effect on the cost of the policy. We observe that Π_D has a better performance when ρ is less than 0.8, and the worst performance is for $\rho = 0.9$. Recall that the routing threshold α of Π_D does not depend on λ. Also recall that the offset is non-decreasing in λ. When all the other parameters are fixed, increasing λ then results in a greater number of states in which average cost optimal policy differs from Π_D. Apart from the instances with $\rho > 0.8$, decreasing γ results in better performance.

Figure 3.2 depicts the the differences between the optimal policy and Π_D in more detail. Shaded parts correspond to controls of Π_D, and rectangles correspond to controls in the average cost optimal policy. Note that we only show the routing controls as the policies do not differ in allocation control.

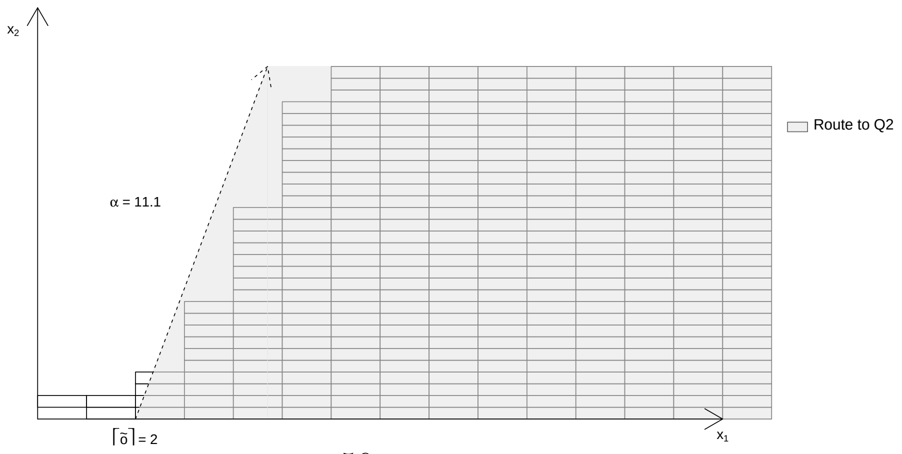
The figure indicates that Π_D more closely matches the optimal routing policy as γ increases, when $\rho = 0.85$. Class 1 customers are given priority (due to the $c\mu$ rule) therefore system spends more time processing Class 1 customers. Consequently a change in μ_2 affects the system less compared to a change in μ_1. This explains why the performance of proposed algorithm is more sensitive to changes in ρ compared to changes in γ.



(a) $\gamma = 0.65$, Suboptimality $= 21.04\%$



(b) $\gamma = 0.85$, Suboptimality $= 10.61\%$



(c) $\gamma = 0.9$, Suboptimality $= 7.06\%$

Figure 2.7: Routing controls of Π_D vs. optimal average cost policy for $c_1, c_2 = \{10, 1\}, \rho = 0.85$

Chapter 3

N-Buffer fluid model with Gurvich-type routing

The previous chapter is devoted to the analysis of a discrete stochastic (queueing) network and there the corresponding fluid model is used to get an approximation to the optimal discrete policy. In this chapter and the next, our focus is solely on fluid models. Although in Chapter 1 and 2 the motivation for the fluid models is briefly stated, we start this chapter with a big picture overview on where these (fluid) models fit in the grand scheme of optimal control.

The model in this chapter is an N-buffer parallel fluid network with a single flexible server and Gurvich-type routing. After the overview in Section 3.1, in Section 3.2 we describe the model formally. The associated linear programming model is given in Section 3.3. In Section 3.4, we outline the optimality equations. Next, we prove that the $c\mu$ rule is optimal for the scheduling control of Station 2 in Section 3.5. Further structural proofs are given in Section 3.6. To show the implications of the structural results, in Section 3.7 we explain the idea of how to compute exactly the optimal policy for a network with $N = 3$. Lastly, in Section 3.8 we generalize these ideas into a generic procedure that determines the optimal policy for a network with any given number of parallel buffers.

3.1 Overview: Fluid models and the optimal control

Optimal control theory is concerned with finding a control law for a dynamic system in order to achieve a performance goal. The evolution of the dynamic system is tracked through the state information and the decision-maker takes actions (also called as controls) which in turn affect the future state trajectory. One major distinction among control problems is on how the dynamic system evolves with respect to time. Continuous and discrete systems differ in measurement of states and placement of actions, respectively, in continuous or discrete-time steps.

From a modelling stand-point, discrete-time models are more realistic as in reality the system can only be measured and the controls can be put into effect in discrete time steps. Yet, these models come with more computational challenges than their continuous equivalents. Both in continuous and discrete-time systems, optimality equations outline the conditions for a set of controls to be optimal with use of a functional operator called the value function. The value function keeps track of the best possible value of the objective as a function of the state. The relationship between the value function and optimal controls lead to a recursive update rule which results in well-known dynamic programming in discrete-time systems. Although dynamic programming exploits the recursive structure of the problem efficiently, it becomes less practical with larger applications. As a consequence, value function approximations, approximate dynamic programming and Q-learning have been popular in the research literature. For a through review, we refer the interested reader to Powell [35] and Bertsekas [8].

Control problems also distinguish between performance goals. Finite-horizon models are concerned with optimization of a cost/reward objective on the state trajectory up to a given time horizon whereas infinite horizon models are concerned with long-run performance measures. Infinite horizon discrete-time models models are generally easier to solve compared to corresponding finite horizon ones, as the dependence on time is less explicit. Iterative algorithms as value iteration and policy iteration are popular tools in solving these problems and their idea is based on updating an estimate of value function until estimates between two consecutive iterations converge. Nevertheless these computational approaches suffer from the "curse of dimensionality", which implies that as the number of state-action pairs grow, the number of optimality equations to be satisfied become explosive, and one again must resort to approximations or good guesses of initial policies.

Continuous-time optimal control often suffers from similar computational challenges. The optimality equation, called the HJB equation, is a partial differential equation to be satisfied for all time-state pairs. In many problems, solving the HJB equation is analytically and computationally highly challenging. Computing an analytical solution may involve a guess of the optimal value function or the optimal set of controls and verifying that they are in fact optimal. Although the verification is straight-forward, without a good guess computing an analytical solution is often difficult. On the other hand evaluating the gradient of the value function is much easier. The maximum principle, also known as Pontyragin's maximum principle, provides necessary but not sufficient conditions on optimality using the gradient of the value function. Bertsekas [8] shows that if the controls lie in a polyhedral space, then in fact the maximum principle provides both sufficient and necessary conditions on optimality.

A final distinction on control problems is the effect of randomness. In deterministic control problems, a jump from one state to another only depends on the previous state trajectory and the sequence of actions. For stochastic control problems, taking an action might lead to any one of many possible states. As a result, stochastic problems are often more combinatorial. Control of queueing networks fit into the class of stochastic discrete-time optimal control problems.

Deterministic/stochastic, discrete/continuous time control problems arise in many different contexts and have a variety of applications. Now, we are going to move on to a class of deterministic continuous-time optimal control problems, namely fluid models, arising as approximations of discrete-time stochastic queueing models. Fluid models replace the stochasticity of the discrete-time control problem by replacing the random processes with their averages. The resulting control problem becomes both deterministic and continuous therefore more tractable then its discrete counterpart. A major question is: What information on the queueing network we can derive by analysing the far-simpler fluid network? The answer to this question was not clear until 1990's. In his seminal work Dai [12] shows that, under positive Harris recurrence, a discrete-network is stable if its fluid model is stable. For two station multi-class queueing networks, Dai and Vande Vate [14] outline globally stabilizing conditions in order to achieve stability.

After the link between stability of discrete and fluid networks became fairly established, the focus of research literature shifts to the optimality of networks. Specifically, can we determine the optimal policies for a given discrete network using optimal policies of the associated fluid model? The notion of asymptotic optimality helps to answer this question. Asymptotic optimality is essentially a consistency criterion between a fluid optimal policy with a discrete-policy evaluated in the discrete-stochastic network. It requires existence of the limit of queue-length vectors under a particular scaling (called fluid scaling) and consistency of these limits with the fluid optimal policy. For the total cost minimization objective in the discrete network, a strong form of solidarity between the discrete optimization problem and a related total-cost optimal control problem is established (Meyn [27]). It is shown that an optimized network possesses a fluid limit model which is itself optimal with respect to a total cost criterion. However, there is no such strong link formed with an average cost minimization problem in discrete network with a corresponding fluid model. A step toward establishing a link can be found in Meyn [30], where the author proves that a certain class of policies (MaxWeight) are approximately average cost optimal in heavy traffic, with logarithmic regret. Consideration of near-optimal control of queueing networks using fluid models are subject to other papers including Bauerle [7], Meyn [31] and Nazarathy et al. [34]. However, it must be noted that several classes of fluid models (depending on the structure of the objective) can be very difficult to solve. Hence, development of efficient solution procedures has attracted the research community and are studied in a number of papers including Fleischer et al. [15] and Weiss [40].

Another promising aspect of fluid models, which mainly motivates the current chapter, is that they provide insights on the optimal policies of the discrete model. Specifically, even under conditions where no strong tie exists between a fluid and discrete model, the fluid model can provide hints on the structure of the optimal policies and may serve as a good approximation to the discrete optimal policy. This idea is explored in Avram et al. [5], where the optimal policy of small-scale fluid networks are used in order to derive a general heuristic designed to solve the much broader class of networks. The paper puts forth a very interesting conjecture: *"A more general principle, where if a static priority scheme is optimal for the fluid model, the same policy is optimal for the stochastic model. Furthermore, whenever the fluid model predicts a linear threshold curve is optimal, the optimal policy for the stochastic model is also of threshold type, but with a nonlinear threshold curve."* This observation hints a strong link between fluid and stochastic counterparts, even though the theoretical foundation has not been fully justified.

Note that this conjecture is consistent with the analyzed model in Chapter 2. For the scheduling rule, the $c\mu$ rule - being based on static priority - is proven to be optimal for both the discrete and the fluid model. Furthermore, the optimal non-linear discrete policy is approximated by the optimal (linear) fluid policy. These observations motivate us to further investigate the fluid models on their own, as theoretical analysis of fluid models are simpler than the discrete networks, the optimal fluid policy can be translated to a discrete policy that is asymptotically optimal and finally this policy can be very insightful on the structure of the optimal policy of the corresponding discrete network. Next, we formally describe the fluid model that we focus in this chapter.

3.2 Model description

In this chapter, we focus on a fluid network composed of N parallel buffers with Gurvich type routing as presented in Figure 3.1. Fluid in buffer k is denoted as class k fluid. We assume that the system receives fluid at a time-constant rate λ and incoming fluid is distributed (or routed) among N buffers. The set of the indices of fluid classes is denoted by $\mathcal{N}$, i.e, $\{1, 2, .., N\} = \mathcal{N}$. The system is composed of a single-station and within the station, there is a flexible server. The server can allocate its capacity to process any combination of fluid classes available in buffers, however its processing rate is class-dependent, meaning that the fluid of class $k \in \mathcal{N}$ can be processed at a rate μ_k by the server. Furthermore, we assume that fluid classes are ordered, i.e., for any two fluid classes $i, j \in \mathcal{N}$, $c_i > c_j$ and $\mu_i > \mu_j$ if and only if $i < j$. The last assumption, we impose is that $\mu_1 > \lambda > \mu_2$. The objective of the fluid optimization model is to choose a set of routing and scheduling controls for every time unit $t \geq 0$ in order to minimize the total holding cost accumulated in the system until all the buffers are empty.

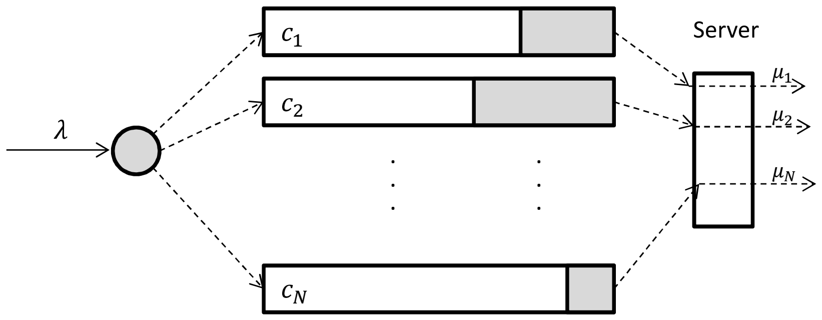


Figure 3.1: N-buffer fluid network with Gurvich-type routing

The draining time of a buffer refers to a time when the buffer level hits 0 and does not build up afterwards. This problem can be modelled as a trajectory control problem. Let T_k be the draining time of fluid k, and set $T = \max_k(T_k)$. Also let $x_k(t)$ the quantity of fluid at buffer k at time t. Let $\{u_k^r(t), u_k^a(t) \in [0, 1]\}$ be the routing and allocation controls, respectively. Given initial fluid levels at the buffers, the decision-maker then chooses

$(u_k^r(t), u_k^a(t))$ which solve the following problem:

$$\min \int_{t=0}^{T} \sum_{k \in \mathcal{N}} c_k x_k(t) dt \tag{3.1}$$

$$\dot{x}_k = \lambda u_k^r(t) - \mu_k u_k^a(t), \ \forall k \in \mathcal{N}, \forall t \geq 0 \tag{3.2}$$

$$x_k \geq 0, \ \forall k \in \mathcal{N}, \forall t \geq 0 \tag{3.3}$$

$$\text{with } x(0) = x. \tag{3.4}$$

Constraints (3.2) enforce the system dynamics. Constraints (3.3) assure that fluid levels are non-negative and lastly constraints (3.4) set the starting condition on fluid levels. Before writing down the HJB equations and outlining the conditions that optimal set of controls must satisfy, we now discuss how to numerically solve this continuous optimization problem through time discretization.

3.3 LP model

As an approximation to the original continuous-time control problem, one can formulate the corresponding linear program. The LP formulation takes a discretization parameter on time updates, and the premise of the approximation is that the smaller time intervals become, the better the approximation gets. Yet, increasing granularity comes with a computational cost as it implies increasing the size of the LP model. Note that the number of variables depend on the discretization parameter denoted as Δ.

Sets and Indices:

$$\mathcal{N} : \text{set of fluid classes;}$$

$$i \in \mathcal{N} : \text{fluid class index;}$$

$$\mathcal{T} = \{0, 1, \ldots, (\Delta)T\} : \text{set of time points;}$$

$$t \in \mathcal{T} : \text{time index;}$$

Parameters:

c_j : holding cost rate for fluid class i;

μ_j : service rate for fluid class i;

Decision Variables:

$r_{i,t}$: rate of incoming fluid routed to Buffer i at time t;

$s_{i,t}$: service rate of the server dedicated to Buffer i at time t;

$x_{i,t}$: fluid level in Buffer i at time t.

Formulation:

$$z = \min \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{N}} x_{i,t} c_{i,t} \tag{3.5}$$

$$x_{i,t+1} = x_{i,t} + \frac{1}{\Delta}(r_{i,t+1} - s_{i,t+1}), \forall t \in \mathcal{T}, \forall i \in \mathcal{N} \tag{3.6}$$

$$\sum_{i \in \mathcal{N}} r_{i,t} = \lambda, \forall t \in \mathcal{T} \tag{3.7}$$

$$\sum_{i \in \mathcal{N}} \frac{s_{i,t}}{\mu_i} \leq 1, \forall t \in \mathcal{T} \tag{3.8}$$

$$x_{i,t} \geq 0, \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{3.9}$$

$$r_{i,t} \geq 0, \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{3.10}$$

$$s_{i,t} \geq 0, \forall i \in \mathcal{N}, \forall t \in \mathcal{T}. \tag{3.11}$$

The objective (3.5) the total accumulated holding cost over the time horizon. Constraints (3.6) are on how buffer levels change over time. Constraints (3.7) specify that incoming fluid with rate λ must be distributed among buffers. Constraint sets (3.8) make sure that the capacity of server utilization is not exceeded. Finally, Constraints (3.9), (3.10) and (3.11) are non-negativity constraints imposed on the variables.

Note that Δ is a parameter of the model. To determine the value of Δ, one can start with an initial small value and gradually increase it until the solution value does not change significantly. Through this dynamic search over the values of Δ, the size of the LP can managed. Nevertheless, it must be noted that this approach is prone to discretization error and it is hence an approximation to the original problem. Furthermore, for large sized networks, the computational effort to solve these problems can be significant. One last disadvantage to any numerical solution method is that unless extensive experimental analysis is performed it is difficult to derive general conclusions on the structure of optimal policies. This is the main motivation behind an exact analysis of the model through HJB equations, as described in the next section.

3.4 Optimality equations

First, we write down the HJB equations. Then through the HJB equations, we obtain structural results on the optimal policy. With the insights obtained through the structural results, we then explicitly describe the whole policy. The Hamiltonian function H is defined as follows:

$$H(x, u^r, u^a) = \min_{u_k^r, u_k^a \in \mathbb{U}} \{cx + \sum_{k=0}^{N} p_k(\lambda u_k^r(s) - \mu_k u_k^a(s))\},$$

with p_k corresponding to Lagrange multipliers. The function $H(\cdot)$ corresponds to the value function in the discrete domain and the controls that minimize the right hand side of the equation are then optimal. Note that this equation must be satisfied for all $t \in (0, T)$ and for clarity the time index t is omitted. The complementary slackness conditions, $p_k(t)x_k(t) = 0, \forall k \in \mathcal{N}$ and the first order conditions imply: $\dot{p}_k(s) = -c_k$ for all $s \geq 0$ and $x_k(s) > 0$. For $s \geq 0$ where $x_k(s) = 0$, the structure of $\dot{p}_k$ is not yet known. In the following section, we derive the full structure of the Lagrange multipliers.

Note that at a given time point t, the Lagrange multipliers indicate the structure of the optimal policy. This relationship can be easily seen by rearranging $H(x, u^r, u^a)$ and by expressing Lagrangian multipliers as non-negative weights on controls. The following relationship between the Lagrangian multipliers and the optimal controls is established as follows:

$$\min_{k\in\mathcal{N},k\neq k'} p_k(t) > p_{k'}(t) \Leftrightarrow u^r_{k'}(t) = 1$$
$$p_{k'}(t)\mu'_k(t) > \max_{k\in\mathcal{N},k\neq k'} (p_k(t)\mu_k(t)) \Leftrightarrow u^a_{k'}(t) = 1.$$

A starting point is investigating the optimality of a static priority rule for the scheduling control. Recall that in the previous chapter for the network with $N = 2$, we proved that the $c\mu$ rule holds. Accordingly, the server gives processing priority to the class of fluid with the largest product of cost and service rate as long as its corresponding buffer is non-empty. For the optimal routing control, we observed that for $N = 2$ a linear threshold policy exists. Then for $N > 2$, are the optimal routing policies of threshold-type? If so, how can we derive them for any given parameter combination and for any N? To answer these questions, we now move on to proofs on the structure of the optimal policies.

3.5 The proof of the $c\mu$ rule

Static priority rules give the scheduling priority to a class of fluid over another, regardless of the present fluid levels in the buffers associated with the classes. Hence, in application of these rules the only information required is that whether a buffer is empty or not. Furthermore if a static priority rule is optimal, the initial state or whether the system incurred disruptions does not matter. For these reasons, systems admitting such rules as optimal policies have received significant attention in the literature.

Essentially, our goal in this section is to assess whether the optimal scheduling policy for the model is based on a strict priority rule, and if so derive its complete structure. Researchers have asked this question for different types of models, and one of the most important studies in that regard is Chen and Yao [10]. The authors define a very strong notion of optimality, namely *global optimality*, that requires optimality at every epoch throughout the entire time horizon rather than solely on a long-run cost basis. Although a globally optimal policy may not exist for every model, the authors argue that if one exists, it can be identified through a sequential procedure. The authors prove that to compute the optimal server allocation, one only needs to consider a subset of decision epochs. It is then shown that the globally optimal policy can be derived sequentially by deriving the optimal server allocation by solving a linear program and pasting the solutions together.

It is important to note that this procedure requires to solve a sequence of linear programs and that, under a globally optimal policy, the objective value of a linear program is no greater than the objective of one that is solved earlier. This poses the main challenge when one wants to apply this procedure to our model. Under the Gurvich-type routing option, the output rate of the server depends on the selected routing policy. The dependence on optimal routing controls prevents such a procedure from producing a static priority rule for scheduling control.

The sketch of the proof is as follows: First, we define a performance vector as a descriptive process that measures the performance of the fluid network. If the set of all performance vectors satisfy a set of conditions, namely *strong conservation laws*, then it is shown that the set of all feasible performance vectors define a polyhedron. Furthermore, the extreme points of the polyhedron correspond to strict index policies which are the set of policies that give strict priority to a fluid class over another based on the selected ordering of class indices. Lastly, if a linear function of the performance vector is to be minimized, then it is shown that under the optimal policy the fluid classes are given priority based on their cost rate where the higher the cost rate of a fluid class, the higher scheduling priority it receives.

For our particular problem, we first express the objective in terms of remaining work. The Skorohod problem definition and related theorems are used in proving that *strong conservation laws* hold for the model. It then follows that the policy that gives strict priority to fluid classes based on the product of their holding cost and service rates, is optimal.

Next we start by formal definitions of the processes used in the proofs as well as useful theorems in the main proof. Subsection 3.5.1 is dedicated to these preliminaries. The proof of the $c\mu$ rule is then outlined in Subsection 3.5.2.

3.5.1 Preliminaries

Let $x = (x_1, x_2, \ldots, x_N)$ denote a vector of performance measure of interest. For example in the fluid context, x can refer to the fluid levels at buffers at a given time or the amount of work left to process for each class of fluid. To denote the dependence of the performance vector on policy $\pi \in \Pi$, we use the notation x^π. Furthermore, we define a strict priority policy as a policy that gives scheduling priority to classes according to given permutation of class indices $\phi = (\phi_1, \phi_2, \ldots, \phi_N)$. For example, the strict priority policy $\pi(\phi)$ gives highest scheduling priority to fluid class ϕ_1 and the lowest scheduling priority to fluid class ϕ_N. We call an admissable policy, any scheduling policy that uses the server at full capacity as long as the system is non-empty.

Definition 3.5.1 (Green [16]). Let S be a subset of $\mathcal{N}$. The set of all performance vectors $\{x^\pi : \pi \in \Pi\}$ is said to satisfy strong conservation laws if there exists a set function $f : 2^N \to \mathbb{R}_+$ such that

$$\sum_{j \in \mathcal{N}} x_j^\pi = f(\mathcal{N}), \quad \forall \pi \in \Pi \tag{3.12}$$

$$\sum_{j \in S} x_j^{\pi(\pi(\phi))} = f(S), \quad \forall \phi \text{ such that } (\phi_1, \phi_2, \ldots, \phi_S) = S \tag{3.13}$$

$$\sum_{j \in S} x_j^\pi \geq f(S), \quad \forall \pi \in \Pi. \tag{3.14}$$

If a performance vector satisfies these strong conservation laws then it implies that the total performance over all classes is invariant under any admissable scheduling strategy. The total performance over the classes in any subset $S \subset \mathcal{N}$ is therefore invariant and furthermore is minimized by any strict-priority rule giving priority to those jobs over jobs of classes in S^C.

Theorem 3.5.1 (Green [16]). *Given a set of performance vectors $\{x^{\pi} : \pi \in \Pi\}$ that satisfies strong conservation laws, an objective of the form*

$$\min\left\{\sum_{j\in\mathcal{N}} c_j x_j^{\pi}, \pi \in \Pi\right\},$$

is optimized by the strict-priority rule that assigns priority to the job classes in the order of decreasing cost coefficients. Namely, the strict priority rule $\pi(\phi)$, where

$$c_{\phi_1} \geq c_{\phi_2} \geq \ldots \geq c_{\phi_N},$$

is optimal over all $\pi \in \Pi$.

Although we do not provide the full provide of the proof of Theorem 3.5.1, the main idea is to express the optimization problem as a linear program with an objective $\{\min \sum_{j\in N} c_j x_j\}$ and the feasible region defined by a polyhedron P defined as follows:

$$P = \left\{x \in \mathbb{R}_+ : \sum_{j\in S} x_j \geq f(S), S \subset \mathcal{N}, \sum_{j\in\mathcal{N}} x_j = f(\mathcal{N})\right\}.$$

When the strong conservation laws hold, the set of extreme points of P equals the set of performance vectors of all strict-priority rules. It then follows that the optimal policy is in fact based on a strict priority rule. For the details of the proof, we refer the interested reader to Green [16].

Definition 3.5.2 (The Skorohod Problem (Green [16])). Given $T > 0$ and $x = \{x(t), 0 \leq t \leq T\} \in D([0,T], \mathcal{R}))$ with $x(0) \geq 0$, a regulation of x over [0,T] is a pair $(z, y) \in D([0,T], \mathcal{R}) \times D([0,T], \mathcal{R})$ such that:

1. $z(t) = x(t)+y(t)$, for all $t \in [0, T]$
2. $z(t) \geq 0$, for all $t \in [0, T]$
3. $y(0) = 0$, $y(\cdot)$ is nondecreasing in $[0, T]$, $\int_0^T z(t)y(t) = 0$.

Let $D^+([0, T], S)$ denote the subset of functions in $D([0, T], S)$ that are non-decreasing. Define $Y(x)$ as follows:

$$Y(x) = \{y(\cdot) \in D^+([0, T], \mathcal{R}) : x(t)+y(t) \geq 0, t \in [0, T]\}.$$

Theorem 3.5.2. *For each $x \in D([0, T], \mathcal{R})$ such that $x(0) \geq 0$, there is a unique minimal $y \in Y(x)$, and the pair (z, y) with $z = x + y$ is the unique regulation of x over $[0, T]$, that is, the unique solution to (i), (ii) and (iii) in $D([0, T], \mathcal{R}) \times D([0, T], \mathcal{R})$.*

3.5.2 The proof

As stated earlier, the mixed nature of routing and scheduling controls makes the proof of optimal scheduling control more challenging. In this subsection, we show that the $c\mu$ rule is optimal regardless of the selected routing policy. A routing policy is a collection of routing controls $u^r(t)$ for all time $t \geq 0$. We denote a routing policy by π_r and the set of all admissible routing policies as Π_r. Note that routing controls are not restricted by time or states, therefore the set Π_r stays constant throughout all time points. The proof requires additional definitions of performance processes.

Let $W_k(t)$ denote the class-k workload process corresponding to the amount of work embodied in class k fluid. The idleness process $Y(t)$ corresponds to the total amount of time the server has been idle in $[0, t]$. We use the notation $W_k(t, \pi_r)$ to denote the vectors under routing policy $\pi_r \in \Pi_r$ and the notation $W_k^\pi(t, \pi_r)$ is used to denote the dependence on the scheduling policy $\pi \in \Pi$. The total workload is denoted by $W^\pi(t, \pi_r) = \sum_{k \in \mathcal{N}} W_k^\pi(t, \pi_r)$. The capacity of the server dedicated to process fluid-class k at a given time t under π_r is given by $u_k^{a,\pi}(t, \pi_r)$. The total utilization of the server is then $U^\pi(t, \pi_r) = \sum_{k \in \mathcal{N}} u_k^{a,\pi}(t, \pi_r)$. The output rate of the server is denoted by $O(t, \pi_r) = \sum_{k \in \mathcal{N}} \frac{\lambda}{\mu_k} u_k^r(t, \pi_r)$ where $u_k^r(t, \pi_r)$ corresponds to the routing actions taken under routing policy π_r. Furthermore let $Q^\pi(t, \pi_r)$ denote the queue length vector under scheduling policy π and routing policy π_r and $Q^\pi(t, \pi_r) = \sum_{k \in \mathcal{N}} Q_k^\pi(t, \pi_r)$ the total amount of fluid in buffers at time $t \geq 0$.

Recall that the objective of the fluid optimization problem is to minimize the total holding cost accumulated in the system until time $T \geq 0$. Furthermore, note that if a policy π^* is globally optimal then it should minimize accumulated holding cost at every decision epoch compared to any other policy $\pi \in \Pi$. That is:

$$\sum_{k \in \mathcal{N}} c_k Q_k^{\pi^*}(t, \pi_r) \leq \sum_{k \in \mathcal{N}} c_k Q_k^\pi(t, \pi_r) \text{ for } \pi_r \in \Pi_r, \pi \in \Pi, t \geq 0. \tag{3.15}$$

Now note that for all $\pi_r \in \Pi_r, \pi \in \Pi, t \geq 0$, the queue length vector can be expressed as a linear function of the workload as follows:

$$Q_k^\pi(t, \pi_r) = \mu_k W_k^\pi(t, \pi_r) \text{ for } k \in \mathcal{N}.$$

As a result, the optimality condition (3.15) can be expressed as follows:

$$\sum_{k\in\mathcal{N}} c_k\mu_k W_k^{\pi^*}(t,\pi_r) \le \sum_{k\in\mathcal{N}} c_k\mu_k W_k^{\pi}(t,\pi_r), \text{ for } \pi_r \in \Pi_r, \pi \in \Pi, t \ge 0. \quad (3.16)$$

Next, we show that the optimal policy π^* is in fact a strict priority policy. To do so, we prove that the strong conservation laws hold for the vector $W^\pi(t,\pi_r)$.

Proposition 3.5.3. *The vector* $\{W^\pi : \pi \in \Pi\}$ *satisfies strong conservation laws.*

Proof. Note that the server has no incentive to idle at a given time. It is clear that any policy that does not use the server to the full capacity while there is fluid in the system is sub-optimal. This implies that the following relation must hold under any policy $\pi \in \Pi$ and $\pi_r \in \Pi_r$:

$$\int_0^t W^\pi(s,\pi_r)dY^\pi(s,\pi_r) = 0, \text{ for all } t \ge 0.$$

This in fact corresponds to Skorohod condition, with $w(t) = x(t)+y(t)$ where $x(t)$ corresponds to the the remaining workload process if the server were to be used at full capacity. As a result of this condition, it then follows that any policy $\pi \in \Pi$ that is a candidate to be optimal must satisfy the following:

$$U^\pi(t,\pi_r) = \begin{cases} 1, & \text{for } W^\pi(t,\pi_r) > 0 \\ \min(1, O(t,\pi_r)), & \text{if } W^\pi(t,\pi_r) = 0. \end{cases}$$

Since $W^\pi(t,\pi_r)$ satisfied the Skorohod condition for all $\pi \in \Pi$, it then follows that $W^\pi(t,\pi_r)$ is invariant under $\pi \in \Pi$. Therefore we have, $\forall t \ge 0, \pi_r \in \Pi_r$:

$$\sum_k W_k^\pi(t,\pi_r) = f(\mathcal{N},\pi_r), \text{ for all } \pi \in \Pi.$$

Next, consider a subset $S \subset \mathcal{N}$. Following Definition 3.5.1, an S-policy is defined as a policy that gives scheduling priority to S-class fluid rather than classes in S^C. Let $\Pi(S)$ be the set of all $S-$policies. it then follows that:

$$\sum_{k \in S} W_k^\pi(t, \pi_r) = f(S, \pi_r), \text{ for all } \pi \in \Pi(S)$$
$$\sum_{k \in S} W_k^\pi(t, \pi_r) \geq f(S, \pi_r), \text{ for all } \pi \in \Pi.$$

Next, let $\phi = (\phi_1, \phi_2, \ldots, \phi_N)$ a permutation of integers $\{1, 2, \ldots, N\}$. Defining $S_1^\phi = \{\phi_1\}, S_2^\phi = \{\phi_1, \phi_2\}, \ldots, S_N^\phi = \{\phi_1, \phi_2, \ldots, \phi_N\} = \mathcal{N}$. The strict-priority policy $\pi(\phi)$ associated with the permutation ϕ is the unique policy $\pi \in \cap_{k=0}^N \Pi(S_k^\phi)$. That is, $\pi(\phi)$ is simultaneously an S_k^ϕ-policy for all $k = 1, 2, \ldots, N$. With this definition of strict-priority policies the vector $(W_1^\pi(t, \omega), W_2^\pi(t, \pi_r), \ldots, W_N^\pi(t, \pi_r))$ satisfies the strong conservation laws. $\square$

Now that we showed that the strong conservation laws hold for the workload vector, the next step is to just apply Theorem 3.5.1.

Proposition 3.5.4. *Consider the permutation $\phi = \{\phi_1, \phi_2, \ldots, \phi_N\}$ such that:*

$$c_{\phi_1}\mu_{\phi_1} \geq c_{\phi_2}\mu_{\phi_2} \geq \ldots \geq c_{\phi_N}\mu_{\phi_N}.$$

Then,

$$\sum_k c_k \mu_k W_k^{\pi(\phi)}(t, \pi_r) \leq \sum_k c_k \mu_k W_k^\pi(t, \pi_r) \textit{ for } \pi_r \in \Pi_r, \pi \in \Pi, t \geq 0. \quad (3.17)$$

Proof. If the strong conservation laws hold for a performance vector, it then follows from Theorem 3.5.1 that the optimal policy is of strict-priority type. This simply suggests that the fluid class associated with a higher $c\mu$ value would be given higher priority over any other class with a lower value. This corresponds to the $c\mu$ rule, completing the proof. $\square$

3.6 Other structural results

Now that the optimal scheduling policy is determined, we consider the results related to determining the structure of the optimal routing policy. The main results of this section are as follows:

- The optimal routing control is bang-bang meaning that at a given time $t \geq 0$ the incoming fluid stream is fully routed to a single buffer.
- If at time $t \geq 0$ the buffer associated with the fluid class i receives the fluid stream, then for any time $t' \geq t$, any fluid class $j > i$ does not receive incoming fluid.
- The full structure of Lagrange multipliers is determined. With this information, we can determine the optimal routing policy through a procedure explained in following sections.

Next, we discuss of the proofs.

Proposition 3.6.1. *Consider a time $t \geq 0$ at which all the corresponding buffers to fluid classes* 1 *through N*-1 *are empty. For any such t, under the* *optimal policy, all the incoming fluid is routed to the buffer of class* 1 *fluid.*

Proof. Let time $t > 0$ such that $x_k(t) = 0, \forall k \in \mathcal{N} \setminus \{N\}$. In order to fully drain the system, fluid of class N must be processed by the server, i.e. $u^a_N(t) > 0$. Note $\lambda > \mu_N$ therefore if the incoming fluid is fully routed to the buffer of class N, it would result in buffer build-up which is clearly sub-optimal. On the other hand, through the $c\mu$ rule if any buffer of class k fluid, with $k \in \mathcal{N} \setminus \{N\}$, is non-empty then this would surely imply that no fluid from buffer N is processed at the server. A policy that builds up buffers is also sub-optimal, as there exists at least one feasible policy under which buffer N drains (consider the policy that routes all incoming fluid to buffer 1, for example). As a result, the original optimization problem reduces to selecting the routing policy so that the fluid from buffer N drains fastest (note that only the fluid in this buffer incurs holding cost, as it is non-empty) while keeping the other buffers empty. The optimal routing control can be considered then as a knapsack problem with an objective of maximizing the rate of processing fluid class N. Formally the problem can be formulated as follows:

$$\min \mu_N \left(1 - \sum_{k \in \mathcal{N}, k \neq 1} \frac{u^r_k(t)\lambda}{\mu_k}\right),$$

Subject to:

$$\sum_{k \in \mathcal{N}} u^r_k = 1$$
$$u^r_k \in [0,1],\ \forall k \in \mathcal{N}.$$

Given $u_N^r(t) > 0$, the optimal solution would be then $u_1^{r*}(t) = 1$ and $u_1^{r*}(t) = 0, \forall k \in \mathcal{N}, k \neq 1$. Therefore, all the incoming fluid is surely routed to the buffer of class 1 fluid. $\square$

Proposition 3.6.2. *Lagrange multipliers admit the following structure:*

$$\dot{p}_j(t) = \begin{cases} -c_j, & \text{for } t < t_j \\ -\frac{c_{j'(t)}\mu_{j'(t)}}{\mu_j}, & \text{if } t \geq t_j, \text{ if } j'(t) = \arg\min_{j \in N, t < t_j} t_j \\ 0, & \text{otherwise.} \end{cases}$$

Proof. According to the $c\mu$ rule, for any given time the server processes fluid of class $j \in \mathcal{N}$ if and only for any fluid class $i \in \mathcal{N}$ with $i < j$, the corresponding buffer is empty. However, this does not imply that when the fluid of class j is in being processed, the processing capacity of the server is solely dedicated to this class of fluid. On the contrary, the buffer corresponding to the fluid of class i may be empty but it can receive an incoming stream of fluid through the routing control. If this happens, the server must dedicate a portion of its capacity to process fluid of class i in order to prevent the buffer from building up. In terms of controls, this case would imply $u_i^a(t), u_j^a(t) > 0$ for given $t > 0$. Recall that though HJB equation if the values of Lagrange multipliers are known at a given time, then one can determine the optimal controls. Therefore for $u_i^a(t), u_j^a(t) > 0$ to hold at time $t > 0$, the Lagrange multipliers must satisfy: $p_i(t)\mu_i = p_j(t)\mu_j$.

Now, consider time $t \in (t_{N-1}, t_N)$ at which all the corresponding buffers to fluid classes 1 through N-1 are empty. Through Proposition 3.6.1, the fluid of class 1 receives an incoming stream of fluid and thus it is processed in the server. This implies that $u_1^a(t)$ and $u_N^a(t) > 0$, therefore $p_1(t)\mu_1 = p_N(t)\mu_N$. Also by continuity of Lagrange multipliers, $p_1(t_{N-1})\mu_1 = p_{N-1}(t_{N-1})\mu_{N-1}$. If $\dot{p}_{N-1} > \frac{\dot{p}_N \mu_N}{\mu_{N-1}}$, it would imply $p_{N-1}(t)\mu_{N-1} > p_{N-1}(t)$ and thus $u_N^a(t) = 0$. On the other hand, if $\dot{p}_{N-1} < \frac{\dot{p}_N \mu_N}{\mu_{N-1}}$, then this implies $\exists t' \in (t_{N-1}, t_N)$ for which $p_{N-1}(t') = 0$ and $p_N(t') > 0$. If this is the case then, $p_{N-1}(t') < p_1(t')$, contradicting with optimal routing rule. Therefore $\dot{p}_{N-1}(t)\mu_1 = \dot{p}_N(t)\mu_N = -c_N\mu_N$, $\forall t \in (t_{N-1}, t_N)$. Backtracking from class N-1 to 1, this result is derived. □

Now that the full structure of Lagrange multipliers are known, we move on with the results relating to the optimal routing policy.

Proposition 3.6.3. *Under the optimal policy, the routing controls are bang-bang.*

Proof. The proof is by contradiction. Suppose the optimal routing policy for time $t \in (t', t'')$, is not bang-bang. This then implies that there exists at least two classes of fluid, $i, j \in \mathcal{N}$ with $i < j$, such that $p_j(t) = p_i(t)$ for $t \in (t', t'')$. As a result, $\dot{p}_j(t) = \dot{p}_i(t)$ for $t \in (t', t'')$. Note that this is not possible for $t < t_i$, since $\dot{p}_j(t) = -c_j \neq -c_i = \dot{p}_i(t)$. Similarly, for $t > t_j$ Proposition 3.6.2 implies that $\dot{p}_j(t) = \frac{\mu_i}{\mu_j}\dot{p}_i(t) \neq \dot{p}_i(t)$. Lastly, consider the time interval (t_i, t_j). Another implication is that $\dot{p}_k(t)$ is a non-decreasing function in t for $t > t_k$, $k \in \mathcal{N}$. For that reason, for any $t \in (t', t'')$ with $t' \geq t_i$ and $t'' \leq t_j$ if $p_j(t) = p_i(t)$ then as a consequence $p_i(t_j) > p_j(t_j)$ conflicting with the $c\mu$ rule. Therefore, the optimal routing controls are bang-bang. $\square$

Proposition 3.6.4. *Given that the optimal routing policy is bang-bang, let $E(t)$ correspond to the index of the fluid class receiving the arrival stream at time $t \geq 0$. If for time $t' > 0$, $E(t') = j$ then for all time $t > t'$, $E(t) \leq j$.*

Proof. Let time $t' > 0$ and $\exists j \in \mathcal{N}$ such that $E(t') = j$. Suppose that $\exists t'' \in (0, t')$ such that $E(t'') = i$ with $i < j$. This implies that $\exists t \in (t'', t')$ such that $p_i(t) = p_j(t)$. Note that for all $i, j \in \mathcal{N}$ with $i < j$, Proposition 3.6.2 indicates that $p_i(t)\mu_i = p_j(t)\mu_j$ $\forall t > t_i$, therefore $p_i(t) < p_j(t)$, $\forall t > t_i$. Also, for $t \leq t_j$, $\dot{p}_j(t) < \dot{p}_i(t)$. As $\dot{p}_j(t) = -c_j > \dot{p}_i(t) = -c_i$ for $t \leq t_i$, $\nexists t \in (t'', t')$ such that $p_i(t) = p_j(t)$. $\square$

Definition 3.6.1. Routing switch, $R(t)$, is a binary variable defined for every time $t \geq 0$ as follows:

$$R_t = \begin{cases} (j, i), & \text{if } \exists \epsilon > 0, \exists i = E(t+\epsilon), \exists j = E(t), i < j \\ \emptyset, & \text{otherwise.} \end{cases}$$

Furthermore a routing path, $P(t)$ is defined as the set of all the routing switch variables or events for all time $s \in (t, T)$. Hence, $P(t) = \{R(s) | s \in (t, T)\}$.

In order to evaluate the total cost following a particular routing path, the corresponding routing controls must be determined at every time step. Note that if at a given time the incoming fluid is routed to buffer 1 fluid then for any time later, only buffer 1 receives fluid. Therefore, at time $t \geq 0$ and state $x(t) = x$, if $u_1^r(t) = 1$ then the total cost that the system accumulates is as follows:

$$v(x) = x^T \vec{Q} x,$$

with

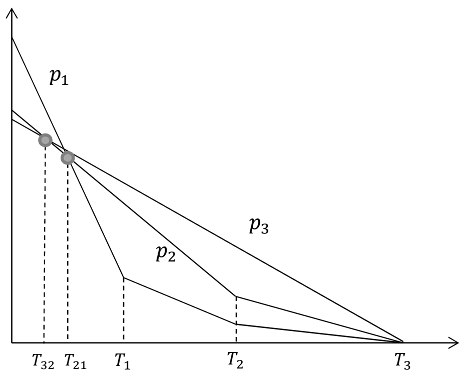
$$\vec{Q} = (2(\mu_1 - \lambda))^{-1} \begin{pmatrix} c_1 & c_2 & \dots & c_N \\ c_2 & c_2\frac{\mu_1}{\mu_2} & \dots & c_N\frac{\mu_1}{\mu_2} \\ \vdots & \vdots & \ddots & \vdots \\ c_N & c_N\frac{\mu_1}{\mu_2} & \dots & c_N\frac{\mu_1}{\mu_3} \end{pmatrix}.$$

Proposition 3.5.4 outline explicitly the structure of the optimal scheduling policy and Propositions 3.6.3 and 3.6.4 give insight on optimal routing policy. Our next goal is to derive the optimal routing policy: determining how the incoming fluid is distributed among buffers and the explicit structure of the routing path. In order to present the idea behind calculation of routing switches, we now proceed with an example network with $N = 3$ and afterwards we generalize the results for all $N \geq 0$.

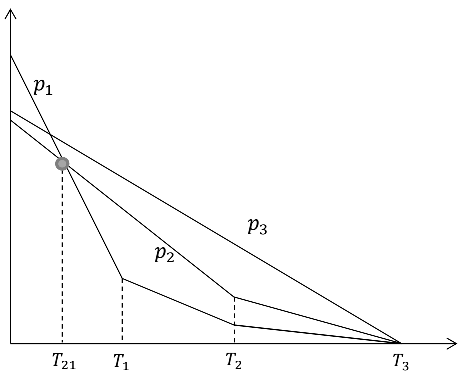
3.7 Example

To determine the optimal routing policy, the optimal routing path and the associated switching times must be calculated. Through Proposition 3.6.1, it is clear that under any starting state there exists a time when the incoming fluid is routed to the buffer of class 1 fluid. Yet, it is possible that the optimal routing policy can admit no routing switch events. In this section, we show how to derive the full routing policy for a network composed of 3 parallel buffers.

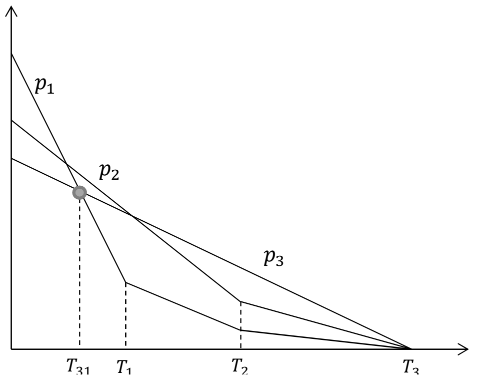
For $N_2 = 3$, the sample space of the routing path $P(t)$ for any $t > 0$ is as follows: $\{(3,2),(2,1)\}$,$\{(3,1)\}$,$\{(2,1)\}$,$\emptyset\}$. The optimal path could be easily determined if the draining times $t_k, \forall k \in \mathcal{N}$ were known. To see this, Figure 3.2 shows different routing paths under different draining times as follows:



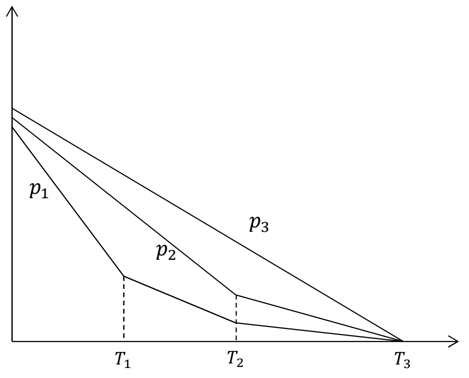
(a) $P(0) = \{(3,2),(2,1)\}$



(b) $P(0) = \{(2,1)\}$



(c) $P(0) = \{(3,1)\}$



(d) $P(0) = \{\emptyset\}$

Figure 3.2: Example of Lagrange Multipliers for different routing paths

1. $P(0) = \{(3,1)\}$

If $P(0) = \{(3,1)\}$, then there exists a routing switch for some time

$t_{31} \geq 0$. The Lagrange multipliers then satisfy $p_1(t_{31}) = p_3(t_{31})$. This relation can be expressed in terms of draining times t_1, t_2, t_3 as follows:

$$c_1(t_1 - t_{31}) + \frac{c_2\mu_2}{\mu_1}(t_2 - t_1) + \frac{c_3\mu_3}{\mu_1}(t_3 - t_2) = c_3(t_3 - t_{31}). \qquad (3.18)$$

Note that for all time $t \geq t_{31}$ there are no other routing switches and all the incoming fluid is routed to Buffer 1. Let $x' = x(t_{31})$ denote the state at time t_{31}. The draining times can be expressed as: $t_1 = t_{31} + \frac{x'_1}{\mu_1 - \lambda}$, $t_2 = t_1 + \frac{x'_2}{\mu_2(1 - \frac{\lambda}{\mu_1})}$ and $t_3 = t_2 + \frac{x'_3}{\mu_3(1 - \frac{\lambda}{\mu_1})}$. In addition, let function $L_{31}(\cdot)$ defined as follows:

$$L_{31}(x') = (c_1 - c_3)x'_1 + (\frac{c_2\mu_2}{\mu_1} - c_3)\frac{\mu_1}{\mu_2}x'_2 + (\frac{c_3\mu_3}{\mu_1} - c_3)\frac{\mu_1}{\mu_3}x'_3 = 0.$$

With this definition of $L_{31}(\cdot)$, if for a given state $x(t)$, $L_{31}(x(t)) = 0$ then Equation 3.18 is satisfied. Thus, evaluating $L_{31}(x(t))$ is sufficient to determine the routing policy. Formally, this relation can be expressed in terms of the routing controls for all $k \in N, t \geq 0$ as follows:

$$u^r_k(t) = \begin{cases} 1, & \text{for } k = 1 \text{ and } L_{31}(x(t)) \geq 0 \\ 1, & \text{for } k = 3 \text{ and } L_{31}(x(t)) < 0 \\ 0, & \text{Otherwise.} \end{cases}$$

Let $x_0 = x$ and suppose $L_{31}(x) < 0$. This suggests that $\exists t_{31} > 0$ such that $L_{31}(x(t_{31})) = 0$. The idea is then to express $x(t_{31})$ in terms of x and t_{31}, then derive t_{31} as a function of x. Note that for $t \in (0, t_{31})$ the incoming fluid is routed to Buffer 3. The scheduling policy, through the

$c\mu$ rule, gives priority to scheduling fluid from Buffer 1 if it is non-empty, and to scheduling fluid from Buffer 2, otherwise. Hence, we can define two directions d_{31}^1, d_{31}^2 to correspond to $\dot{x}$ as follows: $d_{31}^1 = (-\mu_1, 0, \lambda)$, $d_{31}^2 = (0, -\mu_2, \lambda)$. As a result $x(t_{31})$ can be expressed as:

$$x(t_{31}) = x + \min(t_{31}, \frac{x_1}{\mu_1})d_{31}^1 + \max(0, t_{31} - \frac{x_1}{\mu_1})d_{21}^2.$$

Then t_{31} can be derived by solving:

$$L_{31}(\min(t_{31}, \frac{x_1}{\mu_1})d_{31}^1 + \max(0, t_{31} - \frac{x_1}{\mu_1})d_{21}^1) = -L_{31}(x).$$

Lastly, under a starting state x and switching time t_{31}, the total cost accumulated in the system, denoted by $C_{31}(x)$, is given as follows:

$$\begin{aligned} C_{31}(x) = &- \min(t_{31}, \frac{x_1}{\mu_1})\frac{c'd_{31}^1}{2} + c'x\min(t_{31}, \frac{x_1}{\mu_1}) \\ &+ \max(0, t_{31} - \frac{x_1}{\mu_1})\frac{c'd_{31}^2}{2} + c(x - \min(t_{31}, \frac{x_1}{\mu_1})d_{31}^1) + v(x(t_{31})), \end{aligned}$$

where c is the vector associated with cost rates for each class, i.e. $c = (c_1, c_2, ..., c_n)'$.

2. $P(0) = \{(2, 1)\}$

   If there exists a time $t_{21} > 0$ when a routing switch from Buffer 2 to Buffer 1 takes place, then Lagrange multipliers should satisfy: $p_1(t_{21}) = p_2(t_{21})$. This relation can also be expressed as follows:

$$c_1(t_1 - t_{21}) + \frac{c_2\mu_2}{\mu_1}(t_2 - t_1) + \frac{c_3\mu_3}{\mu_1}(t_3 - t_2) = c_2(t_2 - t_{21}) + \frac{c_3\mu_3}{\mu_2}(t_3 - t_2), \tag{3.19}$$

where t_1, t_2, t_3 are again the draining times of buffers. Let $x' = x(t_{21})$, the draining times can be expressed as following: $t_1 - t_{21} = \frac{x'_1}{\mu_1 - \lambda}$, $t_2 = t_1 + \frac{x'_2}{\mu_2(1-\frac{\lambda}{\mu_1}})$ and $t_3 = t_2 + \frac{x'_3}{\mu_3(1-\frac{\lambda}{\mu_1})}$.

A function $L_{21}(x)$ can be then defined as:

$$L_{21}(x) = (c_1 - c_2)x_1 + (\frac{c_2\mu_2}{\mu_1} - c_2)\frac{\mu_1}{\mu_2}x_2 + c_3\mu_3(\frac{\mu_2 - \mu_1}{\mu_2\mu_3})x_3.$$

Therefore, under any state $x(t)$, it is sufficient to evaluate $L_{21}(\cdot)$ to determine the routing policy $\forall k \in N, t \geq 0$:

$$u_k^r(t) \begin{cases} 1, & \text{for } k = 1 \text{ and } L_{21}(x(t)) \geq 0 \\ 1, & \text{for } k = 2 \text{ and } L_{21}(x(t)) < 0 \\ 0, & \text{otherwise.} \end{cases}$$

Next, given an initial state $x(0) = x$ with $L_{21}(x) < 0$, we derive $t_{21} > 0$ such that $L(x(t_{21}) = 0$. Note that for Equation (3.19) to hold, $t_{21} \leq t_1$ is requires. Therefore, the direction $d_{21} = (-\mu_1, \lambda, 0)$ corresponds to $(\dot{x})$ for $t < t_{21}$. Then $x(t_{21})$ can be expressed as: $x(t_{21}) = x + t_{21}d_{21}$. It then follows that t_{21} is:

$$t_{21} = -L_{21}(x)/L_{21}(d_{21}).$$

Finally, the cost of this path under any starting state x, denoted by $C_{21}(x)$ is given as follows:

$$C_{21}(x) = -t_{21}\frac{c'd_{21}}{2} + c'xt_{21} + v(x + t_{21}d_{21}).$$

3. $P(0) = \{(3,2),(2,1)\}$

As opposed to the previous routing paths, the routing path $P(0) = \{(3,2),(2,1)\}$ involves two routing switches. This implies that there exists time $t_{32}, t_{21} \geq 0$ such that $p_2(t_{32}) = p_3(t_{32})$ and $p_1(t_{32}+t_{21}) = p_2(t_{32}+t_{21})$. These equations can be expressed explicitly as follows:

$$c_1(t_1-t_{32})+\frac{c_2\mu_2}{\mu_1}(t_2-t_1)+\frac{c_3\mu_3}{\mu_1}(t_3-t_2) = c_3(t_3-t_{32}) \tag{3.20}$$

$$c_1(t_1-t_{21}-t_{32})+\frac{c_2\mu_2}{\mu_1}(t_2-t_1)+\frac{c_3\mu_3}{\mu_1}(t_3-t_2) = c_2(t_2-t_{21}-t_{32})+\frac{c_3\mu_3}{\mu_2}(t_3-t_2), \tag{3.21}$$

where t_1, t_2, t_3 are given as functions of t_{21}, t_{32} and $x(0) = x$ as follows:

$$\begin{aligned} t_1 &= t_{32} + t_{21}+\frac{x_1-(t_{32}+t_{21})\mu_1}{(\mu_1-\lambda)} \\ t_2 &= t_1+\frac{x_2+t_{21}\lambda}{\mu_2(\mu_1-\lambda)} \\ t_3 &= t_2+\frac{x_3+t_{32}\lambda}{\mu_3(\mu_1-\lambda)}. \end{aligned}$$

We can then define a function $L_{32}(\cdot)$ such that $L_{32}(x) = 0$ such that the draining times corresponding to state x satisfy Equation (3.21). $L_{32}(\cdot)$ is defined as follows:

$$L_{32}(x) = (c_2-c_3)x_1+(c_2-c_3)\frac{\mu_2}{\mu_1}x_2+(\frac{c_3\mu_3}{\mu_1}-c_3)\frac{\mu_3}{\mu_1}x_3$$

It then follows that the switching times $t_{21}, t_{32} > 0$ should satisfy: $L_{21}(x(t_{32}+t_{21})) = 0$ and $L_{32}(x(t_{32})) = 0$. Therefore, it is sufficient to evaluate $L_{21}(\cdot), L_{32}(\cdot)$ to determine the routing policy $\forall k \in N, t \geq 0$:

$$u_k^r(t) \begin{cases} 1, & \text{for } k = 1 \text{ and } L_{21}(x) \geq 0 \\ 1, & \text{for } k = 2 \text{ and } L_{21}(x) < 0, L_{32}(x) \geq 0 \\ 1, & \text{for } k = 3 \text{ and } L_{32}(x) < 0 \\ 0, & \text{Otherwise.} \end{cases}$$

For a given inital state $x(0) = x$, the value of t_{32} can be found by solving $L_{32}(x(t_{32})) = 0$. Then plugging in t_{32}, $L_{21}(x(t_{32}+t_{21})) = 0$ gives the value for t_{21}. Therefore, the idea is to express $x(t_{32})$ and $x(t_{32}+t_{21})$ in terms of t_{21}, t_{32} and x.

Note that as a consequence of the $c\mu$ rule, $p_1(t_1)\mu_1 = p_2(t_1)\mu_2$, therefore $p_1(t_1) < p_2(t_1)$. As a result, $t_{32}, t_{21} > 0$ satisfy the following: $t_{21}+t_{32} \leq t_1$. This implies that $x(t_{32})$ can be expressed as $x(t_{32}) = x+t_{32}d_{32}$ where $d_{32} = (-\mu_1, 0, \lambda)$. Consequently, the switching times t_{32} and t_{21} are linear in state x as follows:

$$\begin{aligned} t_{32} &= -L_{32}(x)/L_{32}(d_{32}) \\ t_{21} &= -\big(L_{21}(x)+L_{21}(-L_{32}(x)/L_{32}(d_{32})\big)/L_{21}(d_{21}). \end{aligned}$$

Finally the cost of the path, $C_{321}(x)$ is given as follows:

$$C_{321}(x) = -t_{32}\frac{c'd_{32}}{2}+c'xt_{32}-t_{21}\frac{c'd_{21}}{2}+t_{21}c'(x+t_{32}d_{31})+v(x+t_{32}d_{31}+t_{21}d_{21}).$$

So far we outlined the derivation optimal trajectory under the assumption that the optimal routing path is known. It is important to note that for

$t > 0$ the sample space of routing path $P(t)$ is not conditional on the state $x(t)$. For a given state, a path may not admit a sequence of non-negative switching times. Such a path is called *infeasible.* The optimal path is then the path resulting in the lowest total cost among the feasible paths.

In order to assess the feasibility of a path, only a function evaluation of the initial state is required. For example, the path $\{2,1\}$ is infeasible if $L_{21}(x(0)) < 0$. Similarly for path $\{3,2,1\}$, feasibility requires both $L_{21}(x(0)) < 0$ and $L_{32}(x(0)) < 0$. It is important to note that under certain parameter settings, infeasibility of one path can imply infeasibility of another. For example, let $x(0) = x$ with $L_{21}(x) > 0$. Now suppose $L_{21}(d_{32}) = (c_1{-}c_2)\mu_1 + c_3\mu_2\frac{\mu_2-\mu_1}{\mu_2\mu_3}\lambda < 0$. If this is the case then surely for any $t_{32} > 0$, $L_{21}(x{+}t_{32}d_{32}) < 0$. Hence if $L_{21}(d_{32}) < 0$ then under any initial state, if path $\{2,1\}$ is infeasible then so is path $\{3,2,1\}$. This suggests that under restricted parameter settings, the search for the optimal path can follow special branching rules.

Restricted parameter settings also have implications on the the relation of the switching times to the initial state $x(0) = x$. Note that under path $\{3,1\}$, the switching time t_{31} is a non-linear function of state x. However, if $\frac{c_2\mu_2}{\mu_1} < c_3$ then Equation (3.18) requires $t_{31} \leq t_1$. Then $x(t_{31})$ can be expressed as: $x(t_{31}) = x{+}t_{31}d_{31}^1$. As a result, $t_{31} = -L_{31}(x)/L_{31}(d_{31})$ hence switching time is a linear function of state x. This observation also hints that an algorithm that exploits such special structure could be more efficient than a general one. However, for the purposes of this chapter, we provide a general algorithm that does not require any further assumptions on the parameters other then the ones imposed at the beginning of the chapter. Next, we generalize the ideas of this example into a general procedure that computes the optimal trajectory for a network composed of N buffers.

3.8 An algorithm to compute the optimal trajectory

In this section, we describe an algorithm to determine the optimal state trajectory for any given initial state. With the earlier results, the structure of Lagrange multipliers is fully known under a parameter setting. Therefore the proposed algorithm employs a procedure that checks whether conditions on Lagrange multipliers are satisfied by a candidate vector switching times for each path. This requires additional procedures such as one that projects a state to a future period, i.e. to $x(t+t')$ given $x(t)$ and $t' > 0$ while the routing policy is known for all time in $(t, t+t')$. Before describing these procedures in more detail, we introduce the following notation for the purposes of the proposed algorithm: the state $x(t) = x$ is represented by vector $\vec{x} \in \mathbb{R}^n_+$ with $\vec{x}_i = x_i$ for all $i \in \mathcal{N}$. The cost vector is given as $\vec{c} = (c_1, c_2, \ldots, c_N)$. Let $\vec{e}_i \in \mathbb{R}^N$ correspond to a unit basis vectors for all $i \in \mathcal{N}$. The vector of draining times is denoted by $\vec{T} \in \mathbb{R}^N$ where $\vec{T}_i$ is the draining time of the class i fluid. Let $\vec{1} \in \mathbb{R}^N$, the vector composed of all ones.

The set of feasible routing controls do not depend on the current state, however the set of feasible scheduling controls do. Through the $c\mu$ rule, the server dedicates its capacity to the fluid class with the lowest index (the highest

$c\mu$ value). Therefore, whenever the fluid level of a class drops to zero, then the output rate of the station varies. Therefore even if according to the routing policy fluid class j receives fluid for a given interval, $\dot{x}(t)$ can change if a buffer empties in that interval. Thus $\dot{x}(t)$ depends on $x(t)$ for time $t \geq 0$. Under fixed routing policy, the following procedure returns $\dot{x}(t)$ given $x(t)$.

```
1: procedure GETDIRECTION($\vec{x}, j$)
2:     $i = \arg\min_{i \in \mathcal{N}}(\vec{x}_i > 0)$
3:     return $(-\mu_i \vec{e}_i + \lambda \vec{e}_j)$
4: end procedure
```

The next procedure calculates a projection of the current state $x(0)$ to a future period $x(t)$, under the assumption that class j fluid receives the arrival stream. Note that this projection must take into account the changes in $\dot{x}(t)$ in case a fluid level drops to zero.

```
1: procedure UPDATESTATE($\vec{x}, t, j$)
2:     $\vec{d} =$ GETDIRECTION($\vec{x}, j$)
3:     if $\vec{x} + t\vec{d} \geq 0$ then
4:         return $\vec{x} + t\vec{d}$
5:     else
6:         $t' = \min_{\{i | \vec{d}_i < 0\}}(-\vec{x}_i / \vec{d}_i)$
7:         return UPDATESTATE($\vec{x} + t'\vec{d}, t - t', j$)
8:     end if
9: end procedure
```

Note that given $\mu_1 > \lambda$ under the optimal policy the system fully drains. If Buffer 1 receives the incoming fluid at a given time then it will do so for all consecutive time points. Therefore, for any time $t \geq 0$ and state $x(t) = x$ the draining times of each fluid can be calculated as given in the following procedure.

```
1: procedure getDrainingTimes($\vec{x}$)
2:     $\vec{T}_1 = \vec{x}_1/(\mu_1 - \lambda)$
3:     for $i \in \mathcal{N} \setminus \{1\}$ do
4:         $\vec{T}_i = \vec{T}_{i-1} + \vec{x}_i \mu_1/(\mu_i(\mu_1 - \lambda))$
5:     end for
6:     return $\vec{T}$
7: end procedure
```

In the previous section, we explained that feasiblity of a path requires conditions on Lagrange multipliers. For example, for path $\{3, 1\}$, there exists $t_{31} > 0$ if and only if $p_1(t_{31}) = p_3(t_{31})$. Therefore, a numerical search can be performed to find the root of $(p_1(t_{31}) – p_3(t_{31}))$. Note that in the case of paths composed m switching events there exists a vector of routing times, denoted as ts with $\dim(ts) = m$, that must satisfy m equations of Lagrange multipliers.

The following procedure first calculates the projection of the state x following the routing policy according to a routing path P and a candidate vector of switching times, ts. From the time of last routing switch and onward, only class 1 receives the incoming fluid therefore the draining times can be calculated. Let function $p(\cdot) : \mathbb{R}^N \to \mathbb{R}$ given by $p(t^1, t^2, \ldots, t^N) =$

$(p_1(t^1), p_2(t^2), \ldots, p_n(t^N))$. Thus $p(\cdot)$ can be evaluated to find ts.

```
1: procedure EVALUATE($\vec{x}, ts, P$)
2:     $\vec{x'} = \vec{x}; k = 0$
3:     for $(j, i) \in P$ do
4:         $\vec{d} =$ GETDIRECTION($\vec{x}, j$)
5:         $\vec{x'} =$ UPDATESTATE($\vec{x}, ts[k], j$)
6:         $k$ ++
7:     end for
8:     $\vec{T} =$ GETDRAININGTIMES($\vec{x'}$)
9:     $res = 0; k = 0$
10:    for $(j, i) \in P$ do
11:        $\vec{T}$ += $ts[k]\vec{1}$
12:        $res[k] = (-\vec{e}_i + \vec{e}_j)^T p(\vec{T})$
13:        $k$ ++
14:    end for
15:    return $res$
16: end procedure
```

Note that if routing path P and the switching times ts are known, then the states $x(t), \forall t > 0$ can be determined given state $x(0) = x$ and total cost of the state trajectory can be computed. Next procedure takes into account changes in $\dot{x}$ as a result of following the routing path and the scheduling policy, and returns the total cost accumulated in the system.

```
1: procedure COSTTRAJECTORY($\vec{x}, ts, P$)
```

```
    $cost = 0; k = 0;$
    $t' = t; \vec{x'} = \vec{x}$
    **for** $(j, i) \in P$ **do**
        $t' = ts[k]$
        **while** $t' > 0$ **do**
            $\vec{d} = \textsc{getDirection}(\vec{x'}, j)$
            **if** $\vec{x'} + t\vec{d} >= 0$ **then**
                $cost \mathrel{+}= t'\vec{c}^T x' + t'\vec{c}^T\vec{d}/2$
                $\vec{x'} = \vec{x'} + t'\vec{d}$
                $t' = 0$
            **else**
                $t'' = \min_{\{i|\vec{d}_i<0\}}(-\vec{x'}_i/\vec{d}_i)$
                $cost \mathrel{+}= t''\vec{c}^T\vec{x'} + \vec{t''}\vec{c}^T\vec{d}/2$
                $\vec{x'} = \vec{x'} + t''\vec{d}$
                $t' \mathrel{-}= t''$
            **end if**
        **end while**
        $k \mathrel{+}= 1$
    **end for**
    $cost \mathrel{+}= \vec{x'}^T\vec{Q}\vec{x'}$
    **return** cost;
**end procedure**
```

Lastly, the algorithm in its general form is given in Algorithm 1. If a path is feasible then the switching times associated with the path, ts has all positive components. Algorithm 1 performs a search on the set of the paths in the sample space, denoted as R, to assess their feasibility and then calculates their cost to find the optimal path.

Algorithm 1 Finding The Optimal Trajectory

```
Require: State $x$; $R$; $Q$; $p(\cdot)$; $\lambda$; $\mu_i, c_i, \forall i \in \mathcal{N}$
Ensure: $L^*$ optimal routing path; $ts^*$, $C^*$ the switching times and the cost
    under $L^*$
 1: $L^* = \emptyset$, $ts^* = \emptyset$
 2: $C^* = \text{costTrajectory}(x, 0, \emptyset)$
 3: for $L \in R$ do
 4:     Find $ts$ such that: $\text{evaluate}(x, ts, L) = 0$
 5:     if $\min ts > 0$ then
 6:         if $C^* > \text{costTrajectory}(x, ts, L)$ then
 7:             $C^* = \text{costTrajectory}(x, ts, L)$
 8:             $L^* = L$, $ts^* = ts$
 9:         end if
10:     end if
11: end for
```

It is important to note that this algorithm exploits the information on the structure of Lagrange multipliers. Furthermore, the procedures described in this section are easily adaptable to different classes of fluid networks if the structure of associated Lagrange multipliers is known. In fact, in the next chapter, we discuss how to adapt this algorithm for a network composed of two serially connected stations and compute the optimal policy.

Chapter 4

Tandem fluid networks with Gurvich-type routing

As stated in the introduction to Chapter 2, large-scale queueing models often require approximations due to complexity of the models. Only for some special classes of networks does the network exhibit a decomposability property that allows an analysis on a station level to be generalized to the whole system, as for Jackson or Kelly networks. Yet, attempts to upgrade these results have been limited due to the fact that explicit exact results are difficult to obtain (Nazarathy [33]).

For single station multi-class queueing networks, the $c\mu$ rule, as explained in earlier chapters, gives the optimal scheduling policy. However, even an extension of this network to one composed of two serially connected stations can admit a very different optimal scheduling policy. Figure 4.1 presents a 2-station MCQN with two different job classes. Recall that the $c\mu$ rule is a myopic rule that essentially maximizes the rate at which cost is drained from the system. In the single-station model, this myopic rule is globally optimal because after being processed in the server, the fluid leaves the system and no longer accumulates holding cost. Now, it is intuitive to think that the same logic would apply to Station 2 of this network. However, as Hordjik and Koole [19] proved, there are parameter settings under which the $c\mu$ rule is not optimal. It applies only when the holding and service rates of the fluid classes competing in Station 2 follow the condition that a class with higher holding cost also has higher service rate.

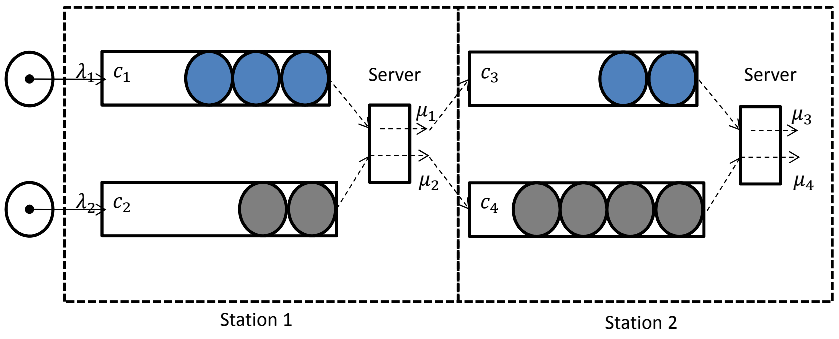


Figure 4.1: MCQN composed of 2 stations in tandem

For Station 1 of this network, under exponential service times and Poisson arrivals, Hordjik and Koole [21] prove that a variant of $c\mu$ rule is optimal in an asymptotic sense. However, for Station 1 of this network the optimal policy is of more complicated nature. Firstly, the scheduling policy of Station 1 directly affects the future buffer sizes of Station 2. This implies that the optimal scheduling policy in Station 1 must depend on the buffer lengths in Station 2 as well as in Station 1. Note that if Station 1 has a higher output rate than Station 2 then this may cause the buffers in Station 2 to build up if Station 1 works at full capacity. On the other hand, idling in Station 1 would help drain fluids from the buffers of Station 2 faster. As a result, the balance between service and cost rates of the stations is a strong indicator on the structure of the resulting policies.

In this chapter, our focus is on two-station tandem fluid networks. We analyze two different networks and each can be considered as an extension of the network provided in Chapter 3. The first model, depicted in Figure 4.2, is composed of a station with a single buffer serially connected to another station composed of N parallel buffers. Both of the stations include a single flexible server. Note that the last station of this network is the same with the network in Chapter 3. It is then interesting to investigate whether the results we obtained in Chapter 3 are valid for this model or not. When does Station 1 idle? What is the structure of the optimal policy in Station 2? Is the optimal routing policy of Station 2 bang-bang? These are some of the questions that motivate our analysis.

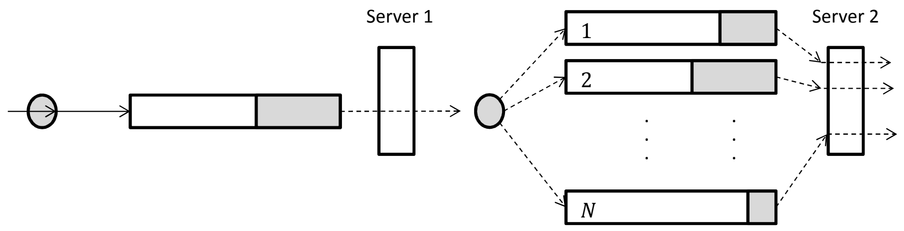


Figure 4.2: Network representation for the Tandem 1-N

The second model we consider in this chapter is one where a station composed of N parallel buffers is serially connected to a station with a single buffer. This model is presented in Figure 4.3. Note that the scheduling policy in Station 1 is much harder to determine with the additional option on routing among different buffers. Apart from the possibility of idling, even determining which class of fluid to process in the server of Station 1 is much more complicated than other networks. It is easy to find a case when the $c\mu$ rule is not optimal in Station 1. Consider a network where the buffer in Station 2 has a higher holding cost then any of the buffers in Station 1. When this is the case, processing the class of fluid with the highest service rate in Station 1 would imply decreasing the output rate of the most costly class of fluid. With these challenging cases in mind, the implications Gurvich-type routing in a network where idling is possible motivates us to analyze this network.

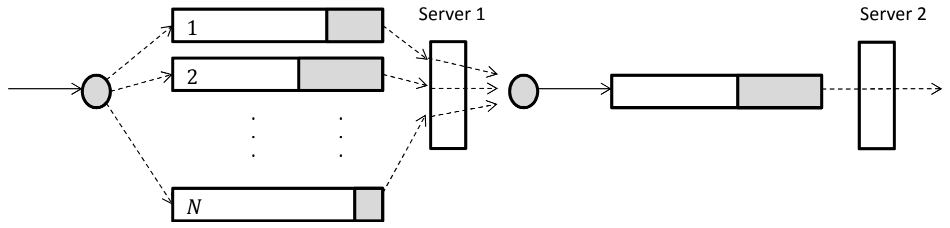


Figure 4.3: Network representation for the Tandem N-1

The tandem models analyzed in this chapter have real-life applications in various areas. Airports, for example, include intricate network topologies involving different processes such as security screening, visa application, customs inspection or flight boarding. Furthermore, the corresponding networks to these processes can be composed of multiple waiting lines in parallel. When this is the case, there is often an employee that determines to which line the next arriving passenger goes, corresponding to a routing decision. In addition, the waiting lines can have different average processing times. For example, the waiting times of passengers in TSA Pre-check security lines are significantly lower because of no requirement to remove shoes, belts, etc. Hence, airport passenger flows constitute a very suitable application domain for the tandem models we focus on this chapter.

This chapter is organized in two main sections, each focusing in detail on the two tandem network models. Section 4.1 is dedicated the the first model. After the formal model description, the associated linear programming model is given in Subsection 4.1.1. The optimality equations are given in Subsection 4.1.2. Next, we provide structural results in Subsection 4.1.3. An example on the policy computation is given in Subsection 4.1.4. Subsection 4.1.5 describes the details of an algorithm to compute the optimal policy. Section 4.2 includes the analysis for the second model. Subsection 4.2.1 includes the necessary conditions on optimality and Subsection 4.2.2 includes structural results.

4.1 Tandem fluid network 1-N

In this section, we consider a network of two serially connected (tandem) stations. For reference, this network is shown in Figure 4.4. The first station is composed of a single buffer and a single server. The output process of this station is then the input process of the second station which is composed of N parallel buffers and a single flexible server. We assume that the fluid coming out of Station 1 can be routed among the buffers of Station 2 through Gurvich type routing. Let N_s be the number of buffers in Station s

with $N_1 = 1, N_2 = N$ and $s \in S = \{1, 2\}$. The sets $\mathcal{N}_1$ and $\mathcal{N}_2$ are defined as $\mathcal{N}_1 = \{1\}$ and $\mathcal{N}_2 = \{1, 2, \ldots, N_2\}$. For notational clarity, we denote the j^{th} buffer (and fluid class) of station s by the index (s, j). Station 1 receives an external arrival stream with rate λ and the server can process fluid up to rate $\mu_{1,1}$, with $\mu_{1,1} > \lambda$ required for stability of the system. For Station 2, the fluid classes are ordered to satisfy following condition on holding costs and service rates: $\mu_{2,i} > \mu_{2,j}$ and $c_{2,i} > c_{2,j}$ if and only if $i < j$, $\forall i, j \in \mathcal{N}_2$. Again for stability the condition $\lambda < \mu_{2,1}$ is required. For this model, the decision–maker seeks to minimize the total holding cost accrued in the system until all the buffers are empty, by determining the scheduling controls of the servers in both stations as well as the routing controls in Station 2.

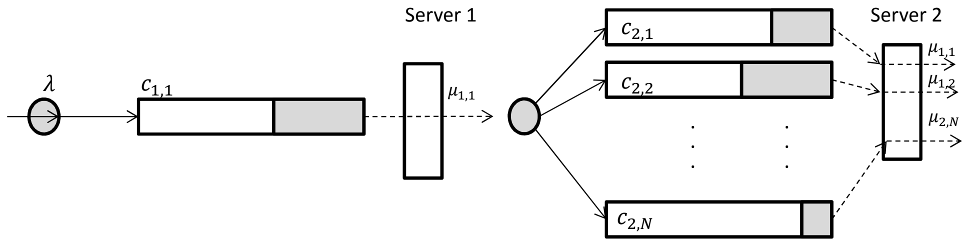


Figure 4.4: Network representation for the Tandem 1-N

Next, we formally define this optimization problem. The scheduling control refers to the extent the capacity of the server in station s is being utilized. The control $u^a_{s,j}(t)$ denotes the proportion of the server in Station s dedicated to processing fluid from buffer (s, j) at time t, with $u^a_{s,j}(t) \in (0, 1)$, $\forall t \geq 0, \forall s \in S, \forall j \in \mathcal{N}_s$ and $\sum_{j \in \mathcal{N}_s} u^a_{s,j} \leq 1$, $\forall t \geq 0, \forall s \in S$. The routing control $u^r_{s,j}(t)$ refers to the proportion of incoming fluid to Station

s being routed to buffer (s, j) at time t. We have then $u^r_{s,j}(t) \in (0, 1)$ and $\sum_{j \in \mathcal{N}_s} u^r_{s,j}(t) = 1$, $\forall t \geq 0, \forall s \in S$. Note that this implies $u^r_{1,1}(t) = 1$. Let $T_{s,j}$ be the draining time of buffer (s, j) and $T = \max_{s \in S, j \in \mathcal{N}_s} T_{s,j}$. Since $\lambda < \mu_{1,1}$ and $\lambda < \mu_{2,1}$ there exists a policy that keeps the fluid levels at zero, once it is achieved. Hence, the optimization model is only defined up to time horizon T is given as follows:

$$\min \int_{t=0}^{T} \sum_{s \in S, j \in \mathcal{N}_s} c_{s,j} x_{s,j}(t) dt \tag{4.1}$$

$$\dot{x}_{1,1}(t) = \lambda u^r_{1,1}(t) - \mu_{1,1} u^a_{1,1}(t), \qquad \forall t \geq 0 \tag{4.2}$$

$$\dot{x}_{2,j}(t) = \mu_{1,1} u^a_{1,1}(t) u^r_{2,j}(t) - \mu_{2,j} u^a_{2,j}(t), \qquad \forall j \in \mathcal{N}_2, \forall t \geq 0 \tag{4.3}$$

$$x_{s,j}(t) \geq 0, \qquad \forall s \in S, \forall j \in \mathcal{N}_s \tag{4.4}$$

$$x_{s,j}(0) = x_{s,j}, \qquad \forall s \in S, \forall j \in \mathcal{N}_s. \tag{4.5}$$

Constraint sets (4.2) and (4.3) are on the evolution of fluid levels in Stations 1 and 2, respectively. Constraints (4.4) insure that fluid levels are non−negative and finally Constraints (4.5) impose the starting condition on fluid levels. As shown in Constraints (4.3) the state is in fact a non−linear function of the controls, which in turn makes the analysis more complicated. Despite its non−linear structure, this continuous optimization problem can be approximated through a linear program formulation. This formulation is given in the next subsection.

4.1.1 LP model

We now present the LP formulation. The approach is based on expressing the the continuous problem as a linear program through discretization of the time interval. With Δ corresponding to the discretization parameter, we now outline the indices, parameters, decision variables and the constraints of this linear program.

Sets and Indices:

$$
\begin{aligned}
i \in \mathcal{N}_1 &: \text{class of job in Station1;} \\
j \in \mathcal{N}_2 &: \text{class of job in Station2;} \\
k \in \mathcal{N}_1 \cup \mathcal{N}_2 &: \text{job index;} \\
s \in 1, 2 &: \text{station index;} \\
\mathcal{T} = \{0, 1, \ldots, (\Delta)T\} &: \text{set of time points;} \\
t \in \mathcal{T} &: \text{time index;}
\end{aligned}
$$

Parameters:

$$
\begin{aligned}
c_i(c_j) &: \text{holding cost rate for fluid class } i(j); \\
\mu_i(\mu_j) &: \text{service rate for fluid class } i(j);
\end{aligned}
$$

Decision Variables:

$$
\begin{aligned}
r_{s,j,t} &: \text{rate of incoming fluid routed to Buffer}(2, j) \text{ at time } t; \\
s_{j,t} &: \text{service rate of the server dedicated to Buffer}(2, j) \text{ at time } t; \\
x_{j,t} &: \text{fluid level in Buffer } j \text{ at time } t.
\end{aligned}
$$

Formulation:

$$z = \min \sum_{t\in\mathcal{T}} \sum_{i\in N_1} x_{i,t}c_{i,t} + \sum_{t\in\mathcal{T}} \sum_{j\in\mathcal{N}_2} x_{j,t}c_{j,t} \tag{4.6}$$

$$x_{i,t+1} = x_{i,t} + \frac{1}{\Delta}(r_{i,t+1} - s_{i,t+1}), \forall t \in \mathcal{T}, \forall i \in \mathcal{N}_1 \cup \mathcal{N}_2 \tag{4.7}$$

$$\sum_{i\in\mathcal{N}_1} r_{i,t} = \lambda, \forall t \in \mathcal{T} \tag{4.8}$$

$$\sum_{j\in\mathcal{N}_2} r_{j,t} = \sum_{i\in N_1} s_{i,t}, \forall t \in \mathcal{T} \tag{4.9}$$

$$\sum_{i\in\mathcal{N}_1} \frac{s_{i,t}}{\mu_i} \leq 1, \forall t \in \mathcal{T} \tag{4.10}$$

$$\sum_{j\in\mathcal{N}_2} \frac{s_{j,t}}{\mu_j} \leq 1, \forall t \in \mathcal{T} \tag{4.11}$$

$$x_{k,t} \geq 0, \forall k \in \mathcal{N}_1 \cup \mathcal{N}_2, \forall t \in \mathcal{T} \tag{4.12}$$

$$r_{k,t} \geq 0, \forall k \in \mathcal{N}_1 \cup \mathcal{N}_2, \forall t \in \mathcal{T} \tag{4.13}$$

$$s_{k,t} \geq 0, \forall k \in \mathcal{N}_1 \cup \mathcal{N}_2, \forall t \in \mathcal{T}. \tag{4.14}$$

The objective (4.6) corresponds to the total accumulated holding cost from $t = 0$ to upper bound on draining time T. Constraint set (4.7) describe how controls affect buffer levels every unit of time. Constraints (4.8) specify that incoming fluid with rate λ must be distributed among buffers in Station 1. Constraints (4.9) assure that the input to Station 2 is equal to the output from Station 1. Constraint sets (4.10) and (4.11) make sure that the capacity of server utilization are not exceeded for Station 1 and 2 respectively. Finally, Constraints (4.12), (4.13) and (4.14) are non−negativity constraints imposed on the variables.

As stated in Section 3.3, a purely numerical solution does not give desired insights on the optimal policies. For example, a very interesting question is under which conditions Station 1 idles. Yet to answer this question, an extensive computational analysis must be performed. Still it must be kept in mind that this approach is nevertheless an approximation. For that reason, we provide an exact analytical analysis in the next subsection.

4.1.2 Optimality equations

A first step in the analysis is outlining the HJB conditions and discussing the implications in terms of optimal sets of controls. Let $p_{s,j}(t)$ be the Lagrange multiplier associated with constraints on $\dot{x}_{s,j}$ in the original optimization model. Hamiltonian function H is then defined as follows:

$$\begin{aligned} H(x,p) = \sum_{s\in S, j\in N_s} c_{s,j}x_{s,j} + \min_{(u^a_{s,j},u^r_{s,j})\in\hat{U}} & \Big\{ p_{1,1}(t)(\lambda - \mu_{1,1}u^a_{1,1}(t)) \\ & + \sum_{j\in\mathcal{N}_2} p_{s,j}(t)(\mu_1 u^a_{1,1}(t)u^r_{2,j}(t) - \mu_{2,j}u^a_{2,j}(t)) \Big\}. \end{aligned}$$

Therefore, the optimal controls at time $t \geq 0$ are defined as follows:

$$\begin{aligned} u^{*a}_{s,j}, u^{*r}_{s,j} \in \underset{(u^a_{s,j},u^r_{s,j})\in\hat{U}}{\arg\min} & \Big\{ u^a_{1,1}\mu_{1,1}\big(\textstyle\sum_{j\in\mathcal{N}_2}(u^r_{2,j}(t)p_{2,j}(t)) - p_{1,1}(t)\big) \\ & + \textstyle\sum_{j\in\mathcal{N}_2} p_{2,j}(t)(-\mu_{2,j}u^a_{2,j}(t)) \Big\}, \end{aligned}$$

where $\dot{p}_{s,j}(t) = c_{s,j}$ if $x_{s,j}(t) > 0$ $\forall s \in S, j \in N_s$ and $\forall t \geq 0$. However, $\dot{p}_{s,j}(t)$ for $t > t_{s,j}$ is not known in advance and in order to determine these rates we need insights on the optimal policies.

Through the HJB equation defined above, given values of $p_{s,j}(t)$, $\forall s \in S$ and $\forall j \in N_s$, one can fully determine the values of optimal controls $(u^r_{s,j}, u^a_{s,j})$, $\forall s \in S, \forall j \in N_s$ and $t \geq 0$. Note that if $\sum_{j \in \mathcal{N}_2}(u^r_{2,j}(t)p_{2,j}(t)) - p_{1,1}(t)) > 0$ then the optimal scheduling control of Station 1 is to idle, i.e. $u^{*a}_{1,1} = 0$. For the optimal routing control at Station 2, $u^{*r}_{2,j'} = 1$ for $j' = \arg\min_{j \in \mathcal{N}_2} p_{2,j}$ and $u^{*r}_{2,j} = 0$ for all $j \in \mathcal{N}_2$ such that $j \neq j'$. Therefore the optimal scheduling control of Station 1 reduces to $u^{*a}_{1,1} = 1$ if $\exists j \in \mathcal{N}_2$ such that $p_{2,j}(t)) > p_{1,1}(t)$ and $u^{a*}_{1,1} = 0$ otherwise. Lastly, the optimal scheduling controls of Station 2 are given as $u^{*a}_{2,j'} = 1$ for $j' \in \arg\max_{j \in \mathcal{N}_2} \mu_{2,j} p_{2,j}$ and $u^{a*}_{2,j} = 0$ for all $j \in \mathcal{N}_2$ such that $j \neq j'$. Via the propositions in the next subsection, further insights on the structure of the optimal policies are obtained.

4.1.3 Structural results

In this subsection, we provide proofs on the structure of optimal policies through arguments on value function and Lagrange multipliers. A brief summary of the results of this section is as follows:

- The optimal scheduling policy for the server at Station 2 follows the $c\mu$ rule.
- For any initial condition, there can at most be a single time interval where the server in Station 2 idles uninterruptedly.
- If at any time the server in Station 1 starts idling, the routing policy in Station 2 changes by the time idling stops.

- For some parameter settings, the scheduling policy of Station 1 is fully determined. For example given $c_{1,1} < \frac{c_{2,N}\mu_{2,N}}{\mu_{2,1}}$ and $\mu_{1,1} > \mu_{2,1}$ the server in Station 1 idles until all buffers in Station 2 are empty. As a result, the routing policy is essentially unimportant and therefore optimal policies have a much simpler structure than the model in Chapter 3. In addition in a network with $c_{1,1} > c_{2,1}$, the server in Station 1 never idles.

- Under all parameter settings, the full structure of Lagrange multipliers are determined. As in Chapter 3, the Lagrange multipliers are used to characterize optimal trajectories.

Next, we give a formal analysis of these proofs. We start with a proof on the optimal scheduling policy of the server in Station 2.

Proposition 4.1.1. *For Station 2, the optimal scheduling policy follows the $c\mu$ rule.*

Proof. The proof of the $c\mu$ rule essentially follows the same lines as the proof of the network in Chapter 3. The goal is again to prove via a global optimality condition that in fact the optimal scheduling policy of Station 2 is based on the $c\mu$ rule regardless of the input stream to Station 2. As the input to Station 2 is in fact the output from Station 1 and the scheduling policy of Station 1 does have an immediate effect on the fluid levels of buffers belonging to Station 2. We prove now that under any scheduling policy of Station 1, the scheduling policy of Station 2 follows the $c\mu$ rule.

Using the same notation as in Chapter 1, under any admissible routing policy $\pi_r \in \Pi_r$ of Station 2, a scheduling policy of Station 2, $\pi \in \Pi$, is optimal if the following holds:

$$\sum_{k\in\mathcal{N}_2} c_{2,k} Q^{\pi^*}_{2,k}(t,\pi_r) \le \sum_{k\in\mathcal{N}_2} c_{2,k} Q^{\pi}_{2,k}(t,\pi_r) \text{ for } \pi_1 \in \Pi_1, \pi_r \in \Pi_r, t \ge 0.$$

Furthermore, weighted fluid level vector can be expressed in terms of weighted workload vector as follows:

$$\sum_{k\in\mathcal{N}_2} c_{2,k} Q^{\pi^*}_{2,k}(t,\pi_1) = \sum_{k\in\mathcal{N}_2} c_{2,k}\mu_{2,k} W^{\pi}_{2,k}(t,\pi_r) \text{ for } \pi \in \Pi, \pi_r \in \Pi_r, t \ge 0.$$

Since the idleness is clearly suboptimal for the scheduling control of Station 2, with the same construction in Chapter 3, we have that the optimal scheduling policy of Station 2 is strictly priority–based with the class of fluid receiving processing priority based on the value of the product of the associated cost and service rate at Station 2. □

Proposition 4.1.2. *Under all parameter settings, for buffer $j > 1$, the Lagrange multipliers take the following form:*

$$\dot{p}_{2,j}(t) = \begin{cases} -c_{2,j}, & \textit{for } t < t_{2,j} \\ -\dfrac{\dot{p}_{2,1}\mu_{2,1}}{\mu_{2,j}}, & \textit{otherwise.} \end{cases}$$

Proof. Note that as a result of the $c\mu$ rule, there exists an ordering of draining times as follows: $t_{2,1} < t_{2,2} < \ldots < t_{2,N}$. In terms of Lagrange multipliers, this implies that $p_{2,1}(t_{2,j})\mu_{2,1} = p_{2,j}(t_{2,j})\mu_{2,j}$ for $j > 1, j \in \mathcal{N}_2$. Also note that

$p_{2,1}(T) = p_{2,j}(T)$ for $j > 1, j \in \mathcal{N}_2$. If $\exists t \in (t_{2,j}, t_{2,j+1})$ such that $p_{2,j}(t)\mu_{2,j} > p_{2,j+1}(t)\mu_{2,j+1} = p_{2,1}(t)\mu_{2,1}$ then the server is dedicated to processing fluid from Buffer $(2, j)$ while it is empty thus resulting in a sub–optimal policy. On the other hand, $p_{2,j}(t)\mu_{2,j} < p_{2,j+1}(t)\mu_{2,j+1}$ implies that if Buffer $(2, j)$ receives the incoming fluid stream, the server would have to give priority to processing Buffer $(2, j+1)$ at the expense of letting $(2, j)$ grow. This is a contradiction to the $c\mu$ rule, as it implies a strict priority order between class indexes. Hence the definition of Lagrange multipliers is consistent with the $c\mu$ rule under any routing policy. □

Proposition 4.1.3. *Given $c_{1,1} < c_{2,1}$ and $\mu_{1,1} > \mu_{2,1}$, under the optimal policy, once Buffer $(2, 1)$ is drained, it remains drained.*

Proof. To prove the proposition, we first analyze the network for $N_2 = 1$. This case reduces to a 2–station tandem network with a single buffer and server at each station. This network can be seen in Figure 4.5.

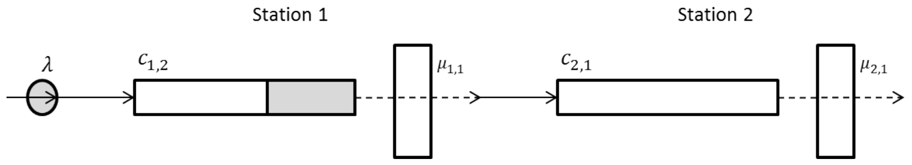


Figure 4.5: Tandem 1-N Network with $N_2 = 1$

Consider time $t > 0$ for which $x_{2,1}(t) = 0$ and $x_{1,1}(t) > 0$. Note that this network has no routing decisions, i.e., $u^r_{2,1} = 1$. Let μ denote the output of Station 1 at time t: $\mu = \mu_{1,1}u^a_{1,1}(t)$. The scheduling policy in Station 2 is then to process the incoming fluid at the maximum rate: $u^a_{2,1} = \min(\frac{\mu}{\mu_{2,1}}, 1)$. Note that given $x_{2,1}(t) = 0$, Station 1 has no incentive to idle. As a result, μ can be considered as a variable with $\mu_{2,1} \leq \mu \leq \mu_{1,1}$. For any μ with $\mu > \mu_{2,1}$, the buffer associated with class $(2,1)$ builds up. For any μ, the cost rate at which fluid accumulates at Buffer $(2,1)$ is then $(c_{2,1}\mu)$, as opposed to the (negative) cost rate of Buffer $(1,1)$, $(-c_{1,1}\mu)$. With $c_{1,1} < c_{2,1}$, the optimal μ is then $\mu = \mu_{2,1}$. This network is also analyzed in further detail in Avram [5].

What does this result imply for a network with $N_2 > 1$? One can see again the decision problem as determining the input to Buffer $(2,1)$ as a function of the output of Buffer $(1,1)$. Let the decision variable μ be $\mu = \mu_{1,1}u^a_{1,1}(t)u^r_{2,1}(t)$. Through earlier example, given $c_{1,1} < c_{2,1}$, $\mu > \mu_{2,1}$ is definitely sub–optimal. Therefore, Buffer $(2,1)$ never builds up. Note that this does pose a requirement such as $\mu = \mu_{2,1}$, since through Gurvich–type routing it can in fact be possible that $u^a_{2,1}(t) = 0$ or idling in Station 1 may be optimal: $u^a_{1,1}(t) = 0$. $\square$

Definition 4.1.1. For $t > 0$, let $j^*(t)$ correspond to the fluid class in process at Station 2 with a non–empty buffer. Furthermore, let $j'(t)$ denote the class of fluid with smallest index in Station 2 that receives incoming fluid stream (via Gurvich type routing). Formally $j^*(t)$ and $j'(t)$ are defined as follows: $j^*(t) = \arg\min_{j \in \mathcal{N}_2, t < t_{2,j}}(t_{2,j}); j'(t) = \arg\min_{j \in \mathcal{N}_2, j \neq 1}(p_{2,j})$.

Proposition 4.1.4. *Given $c_{1,1} <= c_{2,1}$, $c_{1,1} > \frac{c_{2,N}\mu_{2,N}}{\mu_{2,1}}$ and $\mu_{1,1} \geq \mu_{2,1}$:*

$$\dot{p}_{2,1}(t) = \begin{cases} -c_{2,1}, & \textit{for } t < t_{2,1} \\ -c_{1,1}, & \textit{for } t \in (t_{2,1}, t_{1,1}),\ p_{2,1}(t) \geq p_{1,1}(t) \\ p_{2,j'(t)}(t), & \textit{for } t \in (t_{2,1}, t_{1,1}),\ p_{2,1}(t) < \min(p_{1,1}(t), p_{2,j'(t)}) \\ -\frac{c_{2,j^*(t)}\mu_{2,j^*(t)}}{\mu_{2,1}}, & \textit{for } t \in (t_{2,1}, t_{1,1}),\ p_{2,j'}(t) < p_{2,1}(t) < p_{1,1}(t),\ j^*(t) \neq \emptyset \\ -\frac{c_{2,j^*(t)}\mu_{2,j^*(t)}}{\mu_{2,1}}, & \textit{for } t > \max(t_{1,1}, t_{2,1}),\ j^*(t) \neq \emptyset \\ 0, & \textit{otherwise.} \end{cases}$$

In addition, Lagrange multiplier associated with fluid class $(1,1)$ *is given as follows:*

$$\dot{p}_{1,1}(t) = \begin{cases} -c_{1,1}, & \textit{for } t < t_{1,1} \\ \dot{p}_{2,1}(t), & \textit{otherwise.} \end{cases}$$

Proof. Note that under this parameter setting, after Buffer $(2,1)$ drains, it never builds up. This has specific implications in terms of Lagrange multipliers. Consider time $t \in (t_{2,1}, t_{1,1})$, for such a time point if Buffer $(2,1)$ receives full arrival stream while the server at Station 1 works at full capacity then it would cause fluid build–up in Buffer $(2,1)$ (as $\mu_{1,1} > \mu_{2,1}$). Therefore, for that time interval, the Lagrange multipliers have to take into account the capacity of the server in Station 1 being utilized as well as the routing policy of Station 2.

It then follows that for $t \in (t_{2,1}, t_{1,1})$, if $p_{2,1}(t) < p_{1,1}(t)$ (meaning the server at Station 1 works at full capacity), and if Buffer $(2,1)$ receives fluid then it must be the case that $p_{2,1}(t) = p_{2,j'(t)}(t)$, otherwise $p_{2,1}(t)\mu_{2,1} = p_{2,j*(t)}\mu_{2,j*(t)}$ following the $c\mu$ rule, which results in $\dot{p}_{2,1}(t) = -\frac{c_{2,j^*(t)}\mu_{2,j^*(t)}}{\mu_{2,1}}$. Lastly, if Buffer

$(2,1)$ receives fluid and $p_{2,1}(t) > p_{1,1}(t)$ this implies that Station 1 idles at time $t \geq 0$. However, this indicates that processing priority in Station 2 is fully dedicated to Buffer $(2,1)$ while the same buffer is empty. Hence, idleness can not be optimal while Buffer $(2,1)$ receives fluid. In this case setting $\dot{p}_{2,1} = -c_{1,1}$ insures that the server at Station 1 does not idle, in fact it reduces its output rate to match the output of Station 2 while processing from Buffer $(2,1)$. $\square$

Proposition 4.1.5. *Given $c_{1,1} > c_{2,1}$, the server in Station 1 never idles under the optimal policy.*

Proof. Consider the original optimization problem, the objective (4.1) can be re-expressed as follows:

$$\min \int_0^T \left(c_{1,1} \int_0^t \dot{x}_{1,1}(t) ds + \sum_{j \in \mathcal{N}_2} c_{2,j} \int_0^t \dot{x}_{2,j}(s) ds \right) dt. \tag{4.15}$$

Note that the variables $\dot{x}_{s,j}(t)$ are given through Constraints (4.2) and (4.3). Hence the objective (4.15) can be expressed in terms of controls as following:

$$\int_0^T (c_{1,1} \int_0^t (\lambda - \mu_{1,1} u^a_{1,1}(s)) + \sum_{j \in \mathcal{N}_2} c_{2,j} \int_0^t (\mu_{1,1} u^a_{1,1}(s) u^r_{2,j}(s) - \mu_{2,j} u^a_{2,j}(s)) ds) dt. \tag{4.16}$$

Note that the optimization problem then selects a u that minimizes (4.16) and such that Constraints (4.4), (4.5) are not violated. The optimal scheduling controls in Station 1 are given by:

$$u^{*a}_{1,1} \in \arg\min \int_0^T \int_0^t u^a_{1,1}(s) \sum_{j \in \mathcal{N}_2} (c_{2,j} u^{*r}_{2,j}(s) - c_{1,1}) ds dt.$$

Note that for any $s > 0$, $\sum_{j\in\mathcal{N}_2} u^{*r}_{2,j}(s) = 1$. If $c_{1,1} > c_{2,1}$ this implies that $c_{1,1} > \sum_{j\in\mathcal{N}_2} c_{2,j}u^r_{2,j}(s)$ for any $s \geq 0$, therefore $u^a_{1,1}(s)$ is always to set to its upper bound (1 if $x_{1,1}(s) > 0$; otherwise $\frac{\lambda}{\mu_{1,1}}$). Therefore, the server at Station 1 never idles. $\square$

Proposition 4.1.6. *Given $c_{1,1} > c_{2,1}$, for $j \in \mathcal{N}_2$, the Lagrange multipliers take the following form:*

$$\dot{p}_{2,j}(t) = \begin{cases} -c_{2,j}, & \textit{for } t < t_{2,j} \\ -\dfrac{c_{2,j^*(t)}\mu_{2,j^*(t)}}{\mu_{2,1}}, & \textit{for } t \geq t_{2,j}, \textit{ if } j^*(t) \neq \emptyset \\ 0, & \textit{otherwise.} \end{cases}$$

In addition, Lagrange multiplier associated with fluid class $(1,1)$ is given as follows:

$$\dot{p}_{1,1}(t) = \begin{cases} -c_{1,1}, & \textit{for } t < t_{1,1} \\ \dot{p}_{2,1}(t), & \textit{otherwise.} \end{cases}$$

Proof. First, consider fluid class $(2,1)$. Through the $c\mu$ rule, the server starts processing fluid $j^*(t)$ at time $t > 0$. This implies that $p_{2,1}(t)\mu_{2,1} = p_{2,j}(t)\mu_{2,j}$ for $t > t_{j^*(t)}$. Combined with Proposition 4.1.2, for $j \in \mathcal{N}_2$, $\dot{p}_{2,j}(t)$ is given as above.

Through Proposition 4.1.5, it is clear that, $\nexists t > 0$ such that $p_{1,1}(t) < \min_{j\in\mathcal{N}_2} p_{2,j}(t)$ otherwise Station 1 would idle. Note that given $p_{2,1}(t)$ defined as above, $\exists t' > 0$ such that $p_{2,1}(t') = \min_j p_{2,j}(t')$. The Proposition 4.1.5 then holds if $p_{2,1}(t) \leq p_{1,1}(t), \forall t > 0$. Note that for $t < \min(t_{1,1}, t_{2,1})$, $\dot{p}_{1,1}(t) = -c_{1,1} < \dot{p}_{2,1}(t) = -c_{2,1}$. Hence $\dot{p}_{1,1}(t) = \dot{p}_{2,1}(t)$ for $t \geq t_{1,1}$ satisfies this condition. $\square$

Proposition 4.1.7. *Given* $c_{1,1} < \frac{c_{2,N}\mu_{2,N}}{\mu_{2,1}}$ *and* $\mu_{1,1} > \mu_{2,1}$*, the server at Station 1 idles* $\forall t < t_{2,N}$*.*

Proof. For the proof, it is sufficient to consider a state where only buffer $(2, N)$ is non–empty. Formally, let time $t \geq 0$ where $x_{1,1}(t) = 0$, $x_{2,N}(t) > 0$ and $x_{2,j}(t) = 0$ for $j \in \mathcal{N}_2, j \neq 1$. Initially, we make an assumption that for $t \in [0, t_{2,N}]$, the output rate of Station 1, stays constant. We denote this output rate as μ and solve a quadratic equation to determine the best rate. In addition, we also make an assumption that μ admits a fixed upper–bound, however later on we show that this assumption is not required. Depending on the value of μ, the draining time of Buffer $(2, N)$ changes. The total cost accumulated in the system starting at time $t = 0$ and an initial state x can be expressed as a function $v'(\mu)$ as follows:

$$v'(\mu) = \quad c_{1,1}\frac{t_{2,N}^2}{2}\left(\frac{(\lambda-\mu)^2}{\mu_{2,1}-\lambda}+(\lambda-\mu)\right)+c_{2,N}x_{2,N}\frac{t_{2,N}}{2}.$$

Given $t_{2,N} = x_{2,N}\frac{\mu_{2,1}}{\mu_{2,N}(\mu_{2,1}-\mu)}$, $v'(\mu)$ can be expressed as:

$$\begin{aligned}
v'(\mu) =& c_{1,1}t_{2,N}x_{2,N}\frac{\mu_{2,1}}{2\mu_{2,N}}\frac{(\lambda-\mu)}{\mu_{2,1}-\lambda}+c_{2,N}x_{2,N}\frac{t_{2,N}}{2}\\
v'(\mu) =& \frac{t_{2,N}x_{2,N}}{2}\left(c_{2,N}+\frac{c_{1,1}\mu_{2,1}}{\mu_{2,N}}\frac{(\lambda-\mu)}{\mu_{2,1}-\lambda}\right)\\
v'(\mu) =& \frac{tx_{2,N}}{2}\left(c_{2,N}-\frac{c_{1,1}\mu_{2,1}}{\mu_{2,N}}+\frac{c_{1,1}\mu_{2,1}(\mu_{2,1}-\mu)}{\mu_{2,N}(\mu_{2,1}-\lambda)}\right)\\
v'(\mu) =& \frac{x_{2,N}^2}{2}\frac{\mu_{2,1}}{\mu_{2,N}(\mu_{2,1}-\mu)}\left(c_{2,N}-\frac{c_{1,1}\mu_{2,1}}{\mu_{2,N}}+\frac{c_{1,1}\mu_{2,1}(\mu_{2,1}-\mu)}{\mu_{2,N}(\mu_{2,1}-\lambda)}\right)\\
v'(\mu) =& \frac{x_{2,N}^2}{2}\frac{\mu_{2,1}}{\mu_{2,N}}\left(\frac{c_{2,N}-\frac{c_{1,1}\mu_{2,1}}{\mu_{2,N}}}{\mu_{2,1}-\mu}+\frac{c_{1,1}\mu_{2,1}}{\mu_{2,N}(\mu_{2,1}-\lambda)}\right).
\end{aligned}$$

As a result, optimal μ is found as follows:

$$\mu^* \in \quad \arg\min \frac{c_{2,N} - \frac{c_{1,1}\mu_{2,1}}{\mu_{2,N}}}{(\mu_{2,1} - \mu)}.$$

Since $(c_{2,N} - \frac{c_{1,1}\mu_{2,1}}{\mu_{2,N}}) < 0$, it follows that $\mu^* = 0$ therefore the server in Station 1 idles until $t_{2,N}$. Note that μ takes the value of its lower bound (0) therefore an assumption on its upper bound is redundant. Also note that starting at any state with $x_{1,1}(t) > 0$ does not make a difference to the optimal μ. Hence, the result still holds without the assumption of a constant output rate for Station 1. □

Proposition 4.1.8. *Given* $c_{1,1} < \frac{c_{2,N}\mu_{2,N}}{\mu_{2,N}}$, *for* $j \in \mathcal{N}_2$, *the Lagrange multipliers take the following form:*

$$\dot{p}_{2,j}(t) = \begin{cases} -c_{2,j}, & \text{for } t < t_{2,j} \\ -\dfrac{c_{2,j^*(t)}\mu_{2,j^*(t)}}{\mu_{2,1}}, & \text{for } t \geq t_{2,j},\ \text{if } j^*(t) \neq \emptyset \\ -c_{1,1}, & \text{for } t \geq t_{2,j},\ j^*(t) = \emptyset \text{ and } t < t_{1,1} \\ 0, & \text{otherwise.} \end{cases}$$

In addition, the Lagrange multiplier associated with fluid class $(1,1)$ *is as follows:*

$$\dot{p}_{1,1}(t) = \begin{cases} -c_{1,1}, & \text{for } t < t_{1,1} \\ 0, & \text{otherwise.} \end{cases}$$

Proof. As a result of Proposition 4.1.7, all the buffers in Station 2 empty before the buffer in Station 1: $\max_{j\in\mathcal{N}_2} t_{2,j} = t_{2,N} < t_{1,1}$. For the server in Station 1 to idle until time $t_{2,N}$, the following condition must be satisfied: $\forall t \in (0, t_{2,N}),\ p_{1,1}(t) \leq \min_{j\in\mathcal{N}_2} p_{2,j}(t)$. In addition, by the optimality equations the Lagrange multipliers are zero at time $T = \max(t_{1,1}, t_{2,N}) = t_{1,1}$, i.e., $p_{1,1}(t_{1,1}) = p_{2,j}(t_{1,1}) = 0$, $\forall j \in \mathcal{N}_2$. Note that $\dot{p}_{1,1}(t) = \dot{p}_{2,j}(t)$, $\forall j \in \mathcal{N}_2$ for $t \in (t_{2,N}, t_{1,1})$ satisfies these requirements, since it implies $\forall j \in \mathcal{N}_2$, $p_{1,1}(t_{2,N}) = p_{2,j}(t_{2,N})$ and $p_{1,1}(t) < p_{2,j}(t)$ for $t < t_{2,1}$. Combining this with Proposition 4.1.2, yields the given Lagrange multipliers. □

Proposition 4.1.9. *The function $j'(t)$ is non–increasing in t. Furthermore, under any starting condition, $\exists t' > 0$ such that $j'(t) = 1$, for $t > t'$.*

Proof. Through Proposition 4.1.2, $p_{2,1}(t) \geq \frac{\mu_{2,j}}{\mu_{2,1}} p_{2,j}(t)$, $\forall j \in \mathcal{N}_2$, $j > 1$ and $t \geq t_{2,j-1}$. Therefore, $\exists t' > 0$ such that $j'(t) = 1$, for $t > t'$. We prove the first part of the proposition by contradiction. Consider $i, j \in \mathcal{N}_2$ with $j > i$ and a time point t and $\epsilon > 0$ such that $j'(t-\epsilon) = i$ and $j'(t) = j$. Note that $t > \max(t_j, t_i)$ is not possible since $p_{2,1}(t) \leq \min(p_{2,j}, p_{2,i})$ for $t > \max(t_j, t_i)$.

Also, $t < \min(t_j, t_i)$ is not possible either as $\dot{p}_{2,j}(t) = -c_{2,j} < \dot{p}_{2,i}(t) = -c_{2,i}$ for $t < \min(t_j, t_i)$. Therefore if $j'(t) = j$ then $j'(t-\epsilon) \neq i$. The last case to consider is for $t \in (t_{2,i}, t_{2,j})$. For $i > 1, \dot{p}_{2,i}(t)$ is non–decreasing for $t > t_{2,i}$. This then implies that $-c_{2,j} < \dot{p}_{2,i}(t)$, thus contradicting with the original assumption. □

Proposition 4.1.9 has important implications on the optimal routing policy of Station 2. The first implication is that if at some point in the policy Buffer $(2, 1)$ receives incoming fluid stream, then for the remaining time horizon the buffer keeps receiving incoming flow. Second, once the output of Station 1 is routed to the buffer of a particular fluid class, for the rest of the trajectory the buffers of higher cost classes never receive fluid. Lastly, in Proposition 4.1.4 it is clear that the optimal routing policy in Station 2 may not be bang–bang if $c_{1,1} <= c_{2,1}$, $c_{1,1} > \frac{c_{2,N}\mu_{2,N}}{\mu_{2,1}}$ and $\mu_{1,1} \geq \mu_{2,1}$. However for any other parameter settings the optimal routing policy is in fact bang–bang. Although we omit the proof, it follows the same arguments of the proof of Proposition 3.6.3.

Proposition 4.1.10. *An idle period is defined as a time interval when the server in Station 1 works at zero capacity. The optimal scheduling policy of Station 1 can admit at most one idle period.*

Proof. Consider an idle period associated with a time interval (t', t''). By definition there exists an $\epsilon > 0$ such that $p_{1,1}(t'-\epsilon) > \min_j p_{2,j}(t'-\epsilon)$ and that $p_{1,1}(t') = \min_j p_{2,j}(t')$. This implies that $\dot{p}_{1,1}(t') > \dot{\min}_{j\in\mathcal{N}_2} p_{2,j}(t')$. On the other hand, for time t'' the condition must be reversed for idling to cease: $\dot{p}_{1,1}(t'') < \dot{\min}_{j\in\mathcal{N}_2} p_{2,j}(t'')$.

Note that by the definition of draining times, the server at Station 1 can only idle before its draining time. As a result $t'' < t_{1,1}$ and therefore $\dot{p}_{1,1}(t)$ stays constant for $t < t''$. This then implies that during the idle period, $\dot{p}_{2,j'(\cdot)}(\cdot)$ does not stay constant. Existence of another idle period would require that after t'', $\dot{p}_{2,j'(\cdot)}(\cdot)$ first decreases and then increases. Yet through Proposition 4.1.9, there exists a time $\tilde{t} > 0$ such that class $(2, 1)$ receives incoming fluid stream. This implies that $\dot{p}_{2,j'(t)}(t)$ is non–increasing until time $\tilde{t}$ and non–decreasing afterwards. Therefore the optimal scheduling policy of

Station 1 can only admit at most one idle period. □

Proposition 4.1.10 has important implications. For example, consider a time when the server in Station 1 does not idle and the output of Station 1 is routed to a buffer associated with class $(2, j)$ with $j > 1$. If the server starts idling, at the time when idling is stopped, buffer $(2, j)$ will in fact receive no fluid from the output of Station 1. Another observation is that if at a given time Buffer $(2, 1)$ receives fluid from that point on Station 1 does not idle under the optimal policy. These observations show the close relationship between the optimal scheduling policy of Station 1 and the optimal routing policy of Station 2. From another perspective, we can also see that if the optimal scheduling policy of Station 1 is found, the routing policy of Station 2 is highly simplified and not all the possible routing switching times need to be computed. Although the current models seems at first glance more complex that the model in Chapter 3, in some cases optimal policies are actually easier to compute.

Definition 4.1.2. Let $E(t)$ be associated with the value $j'(t)$ for when the server in Station 1 does not idle and $\emptyset$, otherwise. Then *a routing switch*, $R(t)$, is a binary variable defined for every time $t \geq 0$ as follows:

$$R(t) = \begin{cases} (i, j), & \text{if } \exists \epsilon > 0, \exists i, j \in \mathcal{N}_2 \text{ such that } i \neq j,\ i = E(t{-}\epsilon), j = E(t) \\ (i, \emptyset), & \text{if } \exists \epsilon > 0, \exists i \in \mathcal{N}_2 \text{ such that } i = E(t{-}\epsilon) \text{ and } E(t) = \emptyset \\ (\emptyset, j), & \text{if } \exists \epsilon > 0, \exists j \in \mathcal{N}_2 \text{ such that } E(t) = \emptyset \text{ and } j = E(t). \end{cases}$$

The routing path definition then follows from Definition 3.6.1. Note that in Chapter 3, the full structure of routing policies are computed through the use of Lagrange multipliers. Apart from some parameter settings, the optimal scheduling policies of Station 1 and the optimal routing policies of Station 2 are still to be determined. After the structural results, important questions remain. For example, under any starting state, can we determine whether an idling periods exists? If one exists, when does it start and end? What is then the optimal routing policy for Station 2? The answer of these questions again require a procedure that enumerates possible paths and evaluates them using the values of Lagrange multipliers. For the model in Chapter 3 it was shown that the switching times under a given path are linear functions of the starting state x and that lead to a general procedure in determining the optimal state trajectory. For the present model, one can again use a similar idea, but there are additional challenges. For example, as we show in the next subsection, the dependence of the draining times of Station 1 and 2 imply that the draining times are no longer linear functions of the starting state, as a consequence the switching policies are of non−linear form. Next, the idea behind calculation of optimal policies is presented for a small network.

4.1.4 Example $N_2 = 2$

Let's consider a network where $N_2 = 2$ and $c_{2,1} > c_{2,2} > c_{1,1}$, $\mu_{1,1} > \mu_{2,1} > \mu_{2,2}$. The goal is to completely characterize the routing and switching policies. Note that the sample space of the routing path $P(t)$ for any $t > 0$

and starting state $x(t)$ is following: $\{(2,\emptyset),(\emptyset,1)\},\{(\emptyset,1)\},\{(2,1)\},\emptyset\}$ As for the model in Chapter 3, if the draining time $t_{s,j}$ is known for $s \in S, j \in N_s$ then optimal path can be determined.

Consider routing path, $P(0) = \{\emptyset, 1\}$ and initial state $x(t) = x$. According to the routing path, there exists time $t' > 0$ such that for interval $(0, t')$ Station 1 idles. Furthermore, there are no routing switches in (t', T) as the output of Station 1 is directed to Buffer $(2, 1)$. Then the following condition must be satisfied:

$$p_{2,1}(t') = p_{1,1}(t'),$$

which implies that,

$$p_{2,1}(t)(t_{2,1}-t')+p_{2,1}(t_{2,1})(t_{2,2}-t_{2,1}) = p_{1,1}(t)(t_{1,1}-t')+p_{1,1}(t_{1,1})\max(t_{2,2}-t_{1,1},0). \tag{4.17}$$

Note that since the structure of Lagrange multipliers through earlier results (Proposition 4.1.4), Equation (4.17) can be expresses in terms of draining times $t_{1,1}, t_{2,1}, t_{2,2}$ as follows:

$$c_{2,1}(t_{2,1}-t')+\frac{c_{2,2}\mu_{2,2}}{\mu_{2,1}}(t_{2,2}-t_{2,1}) = c_{1,1}(t_1-t')+\frac{c_{2,2}\mu_{2,2}}{\mu_{2,1}}\max(0, t_{2,2}-t_{1,1}). \tag{4.18}$$

Note that since the routing policy includes does not admit any routing switches after time t' the draining times $t_{2,1}, t_{2,2}$ and $t_{1,1}$ can be expressed as functions of state x. Yet, it is important to note that due to the connected nature of stations, the draining times are surely non−linear in x. To see this, we can express $t_{2,1}, t_{2,2}$ and $t_{1,1}$ as follows:

$$t_{1,1} = \frac{x_{1,1}+t'\lambda}{\mu_{1,1}-\lambda}$$

$$t_{2,1} = \begin{cases} \dfrac{x_{2,1}-t'\mu_{2,1}}{\mu_{2,1}-\mu_{1,1}}, & \text{if } \frac{x_{2,1}-t'\mu_{2,1}}{\mu_{2,1}-\mu_{1,1}} < t_{1,1} \\ t_{1,1}+\dfrac{x_{2,1}-t_{1,1}(\mu_{2,1}-\mu_{1,1})-t'\mu_{1,1}}{\mu_{2,1}-\lambda}, & \text{otherwise.} \end{cases}$$

$$t_{2,1} = \begin{cases} t_{2,1}+\dfrac{x_{2,2}\mu_{2,1}}{\mu_{2,2}(\mu_{2,1}-\mu_{1,1})} & \text{if } t_{2,1}+\frac{x_{2,2}\mu_{2,1}}{\mu_{2,2}(\mu_{2,1}-\mu_{1,1})} < t_{1,1} \\ t_{2,1}+\dfrac{x_{2,2}\mu_{2,1}}{\mu_{2,2}(\mu_{2,1}-\lambda)} & \text{if } t_{2,1} > t_{1,1} \\ t_{1,1}+\dfrac{x_{2,2}-(t_{1,1}-t_{2,1})\mu_{2,2}(1-\frac{\mu_{1,1}}{\mu_{2,1}})}{\mu_{2,2}(1-\frac{\lambda}{\mu_{2,1}})}, & \text{otherwise.} \end{cases}$$

Similar to the construction in Section 3.7, the value of t' can be derived by solving (4.18). Next, for a general N_2 we provide details on a procedure that computes the optimal policy.

4.1.5 Numerical procedure

In this section, we propose an algorithm to compute the optimal policy for a given state. The key idea is to perform a numerical search for the switching times that satisfy conditions on Lagrange multipliers. The algorithm explained in Section 3.8 can be adapted to the current problem and to avoid repetition, we only explain the modifications required. For consistency with the notation in Section 3.8, we introduce the following notation: let vector $x(t)$ represent the state $\vec{x}(t) = (x_{1,1}(t), x_{2,1}(t), \ldots, x_{2,n}(t))$ and $p(t) = (p_{1,1}(t), p_{2s,1}(t), \ldots, p_{2,n}(t))$. Similarly, let vector $\mu = (\mu_{1,1}(t), \ldots, \mu_{2,N_2})$ and $c = (c_{1,1}, \ldots, c_{2,N_2})$. Also, we define the set $\mathcal{N}$ to be $\mathcal{N} = \{1, 2, \ldots, N_1+N_2\}$ and $N = N_1+N_2$.

A key procedure of the algorithm is on calculating the value of state $x(t{+}t')$ based on the state $x(t)$ and the routing policy in $(t, t{+}t')$. Note that this requires calculating $\dot{x}(s)$ for $s \in (t, t{+}t')$. Suppose $u^r_{2,j}(s) = 1$ for $s \in (t, t{+}t')$ and $j \in \mathcal{N}_2$. Under this condition, $\dot{x}(s)$ can vary with s, as one (or many) of the buffers can drain and the utilization of servers can change. The next procedure takes into account these changes, and calculates $\dot{x}(s)$ for given $x(s) = \vec{x}$ and index j.

```
1: procedure GETDIRECTION($\vec{x}, j$)
2:     $i = \arg\min_{i \in \mathcal{N}\setminus\{1\}} (\vec{x}_i > 0)$
3:     if $j = \emptyset$: then
4:         return $-\mu_i\vec{e}_i{+}\vec{e}_1\lambda$
5:     else
6:         if $\vec{x}_1 > 0$ : then
7:             if $\mu_1 > \mu_2$ & $c_1 < c_2$ & $\vec{x}_2 = 0$ then
8:                 $\gamma = \mu_2$
9:             else
10:                $\gamma = \mu_1$
11:            end if
12:        else
13:            $\gamma = \lambda$
14:        end if
15:        return $-\mu_i\vec{e}_i{+}\lambda\vec{e}_1{-}\gamma\vec{e}_1{+}\gamma\vec{e}_{j+1}$
16:    end if
```

```
17: end procedure
```

Note that after the last routing switch, the policy is clear: the servers at both stations are utilized at full capacity until there is no fluid at the station, and the output of Station 1 is routed to Buffer $(2, 1)$. Thus the draining times can be calculated. Suppose after time $t > 0$, there are no routing switches. Then for $x(t) = \vec{x}$ the draining times can be calculated by considering a large time increment $t' > 0$ such that surely $x(t+t') = \vec{0}$. While calculating the value of $x(t+t')$, one needs to update $\dot{x}(s)$ for $s \in (t, t+t')$ whenever a fluid class drains. Therefore, the next procedure uses this idea to calculate draining times.

```
1: procedure drainingTimes($\vec{x}$)
2:     $t' = \max\left(\frac{\vec{x}_1}{\mu_1-\lambda}, \sum_{i\in\mathcal{N}\setminus\{1\}} \frac{\vec{x}_i}{\mu_i(1-\frac{\mu_1}{\mu_2})}\right)$
3:     $\vec{T}' = \mathbf{0}$
4:     $\vec{x}' = \vec{x}$
5:     $\tilde{t} = 0$
6:     while $\min_{i\in\mathcal{N}} \vec{x}'_i > 0$ do
7:         $\vec{d} =$ getdirection($\vec{x},' 1'$)
8:         if $\min_{i\in\mathcal{N}}(\vec{x}'_i+t'\vec{d}_i) < 0$ then
9:             $\tilde{t}+ = \vec{x}_i/\vec{d}_i;$
10:            $\vec{T}'_i = \tilde{t}$
11:            $t'- = \tilde{t}$
12:            $\vec{x}' = \vec{x}'+\tilde{t}\vec{d}$
13:            return $\vec{T}'$
```

```
14:         end if
15:     end while
16: end procedure
```

With these modifications on the two procedures, the algorithm in Section 3.8 can be used to obtain optimal state trajectory for any initial state.

4.2 Tandem fluid network N-1

In this section, we consider the network shown in Figure 4.6. The network is composed of two serially connected (tandem) stations. The first station is composed of N parallel buffers and a single flexible server and the external arrival stream (with rate λ) is routed among these buffers through Gurvich type routing. The output of this station is the sole input to Station 2 which is composed of a single buffer and a server. We follow the notation in Section 4.1 and let N_s be the number of buffers in Station s with $N_1 = N, N_2 = 1$ for $s \in S = \{1, 2\}$. We define sets $\mathcal{N}_1$ and $\mathcal{N}_2$ to be $\mathcal{N}_1 = \{1, 2, \ldots, N_1\}$ and $\mathcal{N}_2 = \{1\}$ The buffers in Station 1 are ordered as in the previous section, i.e. $\mu_{1,i} > \mu_{1,j}$, $c_{1,i} > c_{1,j}$ if and only if $i < j$, $\forall i, j \in \mathcal{N}_1$. For stability purposes, the conditions $\mu_{1,1} > \lambda$ and $\mu_{2,1} > \lambda$ are required. Again we assume that the decision–maker seeks to minimize the total holding cost accumulated in the system until all the buffers are empty. To do so, optimal scheduling controls of the servers in both stations as well as the routing controls in Station 1 are to be determined.

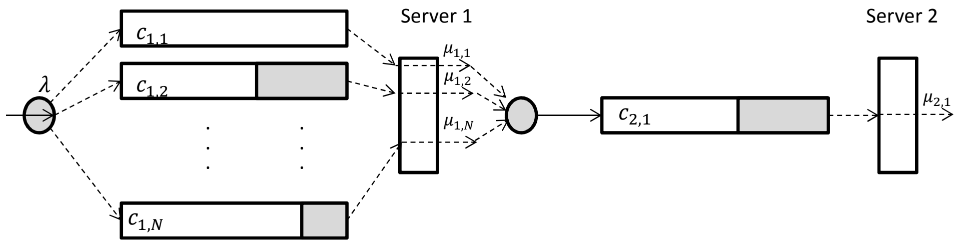


Figure 4.6: Network representation for the Tandem N-1

For the formal definition of the optimization problem we use the same definition of controls in Section 4.1. For $t \geq 0, \forall s \in S, \forall j \in N_s$, the routing and scheduling controls are denoted as $u^r_{s,j}(t)$ and $u^a_{s,j}(t)$. Note that Station 2 includes no routing option, i.e., $u^r_{2,1}(t) = 1$. With $T_{s,j}$ being the draining time of buffer (s, j) and $T = \max_{s\in S, j\in N_s} T_{s,j}$, the model is formally defined as follows:

$$\min \int_{t=0}^{T} \sum_{s\in S, j\in N_s} c_{s,j} x_{s,j}(t) dt \tag{4.19}$$

$$\dot{x}_{1,j}(t) = \lambda u^r_{1,j}(t) - \mu_{1,j} u^a_{1,j}(t), \qquad \forall j \in N_1, \forall t \geq 0 \tag{4.20}$$

$$\dot{x}_{2,1}(t) = \sum_{j\in N_1} \mu_{1,j} u^a_{1,j}(t) - \mu_{2,1} u^a_{2,1}(t), \qquad \forall t \geq 0 \tag{4.21}$$

$$x_{s,j}(t) \geq 0, \qquad \forall s \in S, \forall j \in N_s \tag{4.22}$$

$$x_{s,j}(0) = x_{s,j}, \qquad \forall s \in S, \forall j \in N_s. \tag{4.23}$$

Constraint sets (4.20) and (4.21) are on the evolution of fluid levels in Stations 1 and 2, respectively. Non−negativity of fluid levels are satisfied through Constraints (4.22) and the starting condition on fluid levels is imposed through Constraints (4.23). A linear program formulation of this continuous optimization problem can be formed through time discretization. We omit the formulation here as it is in fact the same with the one given in Subsection 4.1.1 with $\mathcal{N}_1 : \{1, 2, .., N\}$, $\mathcal{N}_2 : \{1\}$. Next, we discuss the optimality equations.

4.2.1 Optimality equations

Let $p_{s,j}(t)$ be the Lagrange multiplier associated with constraints on $\dot{x}_{s,j}$ in the above optimization model. Then, the HJB equation may be expressed as follows:

$$H(x,p) = \sum_{s \in S, j \in N_s} c_{s,j} x_{s,j} + \min_{(u^a_{s,j}, u^r_{s,j}) \in \hat{U}} \Bigg\{ \sum_{j \in N_1} p_{1,j}(t)(\lambda u^r_{1,j}(t) - \mu_{1,j} u^a_{1,j}(t)) + p_{2,1}(t)(\sum_{j \in N_1} \mu_{1,j} u^a_{1,j}(t) - \mu_{2,1} u^a_{2,1}(t)) \Bigg\},$$

The equation above implies that the optimal controls at time $t \geq 0$ are defined as follows:

$$u^{*a}_{s,j}, u^{*r}_{s,j} \in \underset{(u^a_{s,j}, u^r_{s,j}) \in \hat{U}}{\arg\min} \Bigg\{ -\mu_{2,1} p_{2,1}(t) u^a_{2,1}(t) + \lambda \textstyle\sum_{j \in \mathcal{N}_1} u^r_{1,j}(t) p_{1,j}(t) + \sum_{j \in \mathcal{N}_1} (p_{2,1}(t) - p_{1,j}(t)) u^a_{1,j}(t) \Bigg\}$$

$$u^{*a}_{s,j}, u^{*r}_{s,j} \in \underset{(u^a_{s,j}, u^r_{s,j}) \in \hat{U}}{\arg\min} \Bigg\{ \lambda \sum_{j \in \mathcal{N}_1} u^r_{1,j}(t) p_{1,j}(t) + \sum_{j \in \mathcal{N}_1} \mu_{1,j} (p_{2,1}(t) - p_{1,j}(t)) u^a_{1,j}(t) \Bigg\}$$

where $\dot{p}_{s,j}(t) = c_{s,j}$ if $x_{s,j}(t) > 0$ $\forall s \in S, j \in \mathcal{N}_s$ and $\forall t \geq 0$. However, $\dot{p}_{s,j}(t)$ for $t > t_{s,j}$ are not known in advance and in order to determine these rates we again need insights on the optimal policies.

The HJB equation has direct implications on the optimal scheduling and routing controls. For example the server at Station 1 idles for $(p_{2,1}(t)$-$\max_{j\in\mathcal{N}_1}(p_{1,j}(t))(t)) > 0$, for $t \geq 0$. For the optimal routing control at Station 1, $u^{*r}_{1,j'} = 1$ for $j' \in \arg\min_{j\in\mathcal{N}_1} p_{1,j}$ and $u^{*r}_{1,j} = 0\ \forall j \in \mathcal{N}_1$ such that $j \neq j'$. Lastly, for Station 2 there are no routing decisions and the scheduling policy is trivial: $u^{*a}_{2,1}(t) = 1$ if $x_{2,1}(t) > 0$, $u^{*a}_{2,1}(t) = \min(\frac{\sum_{j\in\mathcal{N}_1} u^{*a}_{1,j}(t)}{\mu_{2,1}}, 1)$. Insights on the the structure of the optimal policies are obtained through the results in the next subsection.

4.2.2 Structural results

Note that in the model in Section 4.1, the scheduling policy of Station 1 only requires determination of the allocation of capacity in the server. Without any fluid classes competing for service in Station 1, we proved that in fact one can determine the idling periods of the server. However, in this section, determining the optimal scheduling policy of Station 1 is much more cumbersome. The complication arises from the fact that the scheduling policy of Station 1 affects the buffer sizes of Station 2 and a greedy policy at Station 1 may not be globally optimal. As we show in the next proposition, the greedy policy, embodied by the $c\mu$ rule, is optimal only for a restricted set of parameters.

Proposition 4.2.1. *For the Tandem N-1 network with $\mathcal{N}_1 = \{1, 2\}$, under the optimal scheduling policy of Station 1, the $c\mu$ rule holds for* $(c_{1,i}-c_{2,1})\mu_{1,i} > (c_{1,i+1}-c_{2,1})\mu_{1,i+1}$, $\forall i \in \mathcal{N}_1$.

Proof. For Station 1, if the optimal scheduling policy follows the $c\mu$ rule then at any time point when Station 1 does not idle, the fluid class $(1,1)$ has processing priority over class $(1,2)$. Then the fluid of class $(1,2)$ can only be processed if there is no class $(1,1)$ fluid in the system and when Station 1 does not idle. In tems of Lagrange multipliers, this implies that there exists time point $t' > 0$ such that $\mu_{1,1}(p_{1,1}(t') - p_{2,1}(t')) \geq \mu_{1,2}(p_{1,2}(t') - p_{2,1}(t'))$ then for all time $t \leq t'$, $\mu_{1,1}(p_{1,1}(t) - p_{2,1}(t)) \geq \mu_{1,2}(p_{1,2}(t) - p_{2,1}(t))$. In other words, $\mu_{1,1}(\dot{p}_{1,1} - \dot{p}_{1,2}) + \dot{p}_{2,1}(-\mu_{1,1} + \mu_{1,2}) \leq 0$, for $t \leq t'$.

It is easy to see that such a time point $t' \geq 0$ exists, because otherwise Buffer $(1,1)$ never drains. This then implies that $\exists t' < t_{1,1}$ which gives $\dot{p}_{1,1}(t) = -c_{1,1}$ for $t \leq t'$. On the other hand, by the sufficient conditions on optimality condition, we have $\dot{p}_t \geq -c$, where $p_{s,j}(t)$ are differentiable for all $s \in S$, $j \in \mathcal{N}_s$. This then suggests that $\dot{p}_{1,2}(t) \geq -c_{1,2}$ and $\dot{p}_{2,1}(t) \geq -c_{2,1}$ for $t \leq t'$. Given the initial assumption on parameters, $\dot{p}_{2,1}(t) \geq \frac{c_{1,1}\mu_{1,1} - c_{1,2}\mu_{1,2}}{\mu_{1,2} - \mu_{1,1}}$. Combining the above observations, we have, $\mu_{1,1}(\dot{p}_{1,1}(t) - \dot{p}_{1,2}(t)) + \dot{p}_{2,1}(t)(-\mu_{1,1} + \mu_{1,2}) \leq 0$, for $t \leq t'$. As a result, fluid from Buffer $(1,2)$ is processed after Buffer $(1,1)$ drains. $\square$

The interplay of the policies of Station 1 and Station 2 not only poses challenges in examining static priority rules, but it also has surprising effects on the fluid levels of the buffer in Station 2. Reflecting back on the model in Chapter 3 or the Tandem $1-N$ model, the only case where the fluid level of a buffer increases is when it receives fluid through Gurvich–type routing. This is not surprising, since it seems intuitive that under an optimal policy, a buffer should not build up once it has drained. However, as we explain next, this intuition fails in the current model.

Remark 4.2.1. When a station is not in a idling state, it is desired to fully use its capacity. However, under some parameter settings this can require that Buffer $(2,1)$ drains first, remains empty and then builds up once more. What is even more surprising is that this phenomena even arises for networks where the holding cost of the buffer in Station 2 is the largest in the system. To provide insights on what causes this phenomena, for the rest of the section, we focus on the case where $N_1 = 2$. For the same network, we then prove results for all the possible parameter cases. We leave the analysis for general N_1 to future work.

Proposition 4.2.2. *For Tandem N−1 network with $N_1 = 2$, the Lagrange multiplier associated with class $(2,1)$ takes the following form for all $t \geq 0$:*

$$\dot{p}_{1,1}(t) = \begin{cases} -c_{1,1}, & \text{for } t < t_{1,1} \\ \frac{\dot{p}_{1,2}(t)\mu_{1,2} + \dot{p}_{2,1}(\mu_{1,1} - \mu_{1,2})}{\mu_{1,1}}, & \text{otherwise.} \end{cases}$$

Furthermore, there exists a time $t' > 0$, after which all routing is to Buffer $(1,1)$.

Proof. The first part of the proposition directly comes from the $c\mu$ rule. Essentially once the buffer of class $(2,1)$ is empty, fluid from buffer $(2,2)$ must be processed in the server, provided that idling is not optimal. Correspondingly for time $t \in (t_{1,1}, t_{2,1})$ we have $\mu_{1,1}(p_{1,1}(t) - p_{2,1}(t)) = \mu_{1,2}(p_{1,2}(t) - p_{2,1}(t))$ which leads to the Lagrange multiplier definition above.

For the second part of the proposition, consider a time point $t' > 0$ for which $x_{1,2}(t') > 0$ and $x_{1,1}(t') = 0$. If the incoming fluid to Station 1 is routed to Buffer $(2,1)$ while the server processes from the same buffer, the rate at which the buffer drains is at most $(\mu_{1,2}-\lambda)$. Whereas if the routing is to Buffer $(1,1)$ then the rate at which Buffer $(1,2)$ drains is be at most $(\mu_{1,1}(1-\frac{\lambda}{\mu_{1,2}})$, which is greater than $(\mu_{1,2}-\lambda)$. As a result there exists a time t, after which all routing is to Buffer $(1,1)$. $\square$

Proposition 4.2.3. *Given* $c_{1,1} < c_{2,1}$ *and* $\mu_{1,1} > \mu_{2,1}$, $\mu_{2,1} > \lambda+\mu_{1,2}(1-\lambda/\mu_{1,1})$, *the Buffer* $(2,1)$ *can build up once drained.*

Proof. From Proposition 4.2.2, there exists a time $t' < t_{1,1}$ such that the optimal control is to route all incoming fluid to Buffer $(1,1)$ from t' and after. Now consider a time $t \geq t'$ and $t \leq t_{1,1}$ such that $x_{2,1}(t) = 0$ and $x_{1,1}(t) \geq 0, x_{1,2}(t) > 0$.

One obvious candidate optimal policy is one that keeps the fluid level of Buffer $(2,1)$ empty while initially draining the Buffer $(1,1)$ and later on Buffer $(1,2)$. For this policy, denoted as π_1, the total holding cost accumulated until the system fully empties can be expressed through the quadratic function below:

$$v^{\pi_1}(x,t) = \frac{c_{1,1}}{2}x_{1,1}t_{1,1}^{\pi_1}+c_{1,2}x_{1,2}t_{1,1}^{\pi_1}+\frac{c_{1,2}}{2}x_{1,2}(t_{1,2}^{\pi_1}-t_{1,1}^{\pi_1})$$

where $t_{1,1}^{\pi_1}$ and $t_{1,2}^{\pi_1}$ correspond to the draining times of Buffers $(1,1)$ and $(1,2)$ respectively under policy π_1. These draining times are in fact functions of

$x_{1,1}(t)$ and $x_{1,2}(t)$ and they can be expressed as follows:

$$t^{\pi_1}_{1,1} = \frac{x_{1,1}(t)}{\mu_{2,1}-\lambda}$$
$$t^{\pi_1}_{1,2} = t^{\pi_1}_{1,1}+\frac{x_{1,2}(t)}{\mu_{1,2}(1-\frac{\lambda}{\mu_{2,1}})}.$$

Now consider an alternative policy, denoted as π_2, where instead of keeping Buffer $(2,1)$ empty the policy builds up the buffer by processing fluid from Buffer $(1,1)$ at a higher rate. Note that this is only possible for $\mu_{1,1} > \mu_{2,1}$. When Buffer $(1,1)$ empties, the output rate of Station 1 decreases. As a result, Buffer $(2,1)$ empties again. Policy π_2 is associated with the quadratic cost defined as follows:

$$\begin{aligned} v^{\pi_2}(x,t) =& \frac{c_{1,1}}{2}x_{1,1}t^{\pi_2}_{1,1}+c_{1,2}x_{1,2}t^{\pi_2}_{1,1}+\frac{c_{1,2}}{2}x_{1,2}(t^{\pi_2}_{1,2}-t^{\pi_2}_{1,1}) \\ &+\frac{c_{2,1}}{2}(\mu_{1,1}-\mu_{2,1})(t^{\pi_2}_{1,1})^2+\frac{c_{2,1}}{2}\frac{((\mu_{1,1}-\mu_{2,1})t^{\pi_2}_{1,1})^2}{\mu_{2,1}-(\mu_{1,2}(1-\frac{\lambda}{\mu_{1,1}}))}, \end{aligned}$$

where $t^{\pi_2}_{1,1}, t^{\pi_2}_{1,2}$ are given as:

$$t^{\pi_2}_{1,1} = \frac{x_{1,1}(t)}{\mu_{1,1}-\lambda}$$
$$t^{\pi_2}_{1,2} = t^{\pi_1}_{1,1}+\frac{x_{1,2}(t)}{\mu_{1,2}(1-\frac{\lambda}{\mu_{1,1}})}.$$

It is clear to see that the condition $v^{\pi}_2(x,t) < v^{\pi}_1(x,t)$ implies that the policy π_1 is not optimal. Note that for this condition to be satisfied $c_{2,1} > c_{1,2}$ and $\mu_{1,1} > \mu_{2,1} > \lambda+\mu_{1,2}(1-\lambda/\mu_{1,1})$ are required. For a network with parameters $c_{1,1} = 80, c_{1,2} = 40, c_{2,1} = 89,\ \mu_{1,1} = 6, \mu_{1,2} = 1.5, \mu_{2,1} = 4,\ \lambda = 4$ and initial state $x_{1,1}(0) = 30, x_{1,2}(0) = 1, x_{2,1}(0) = 30$ Figure 4.7 shows the optimal state trajectory resulting from the LP solution.

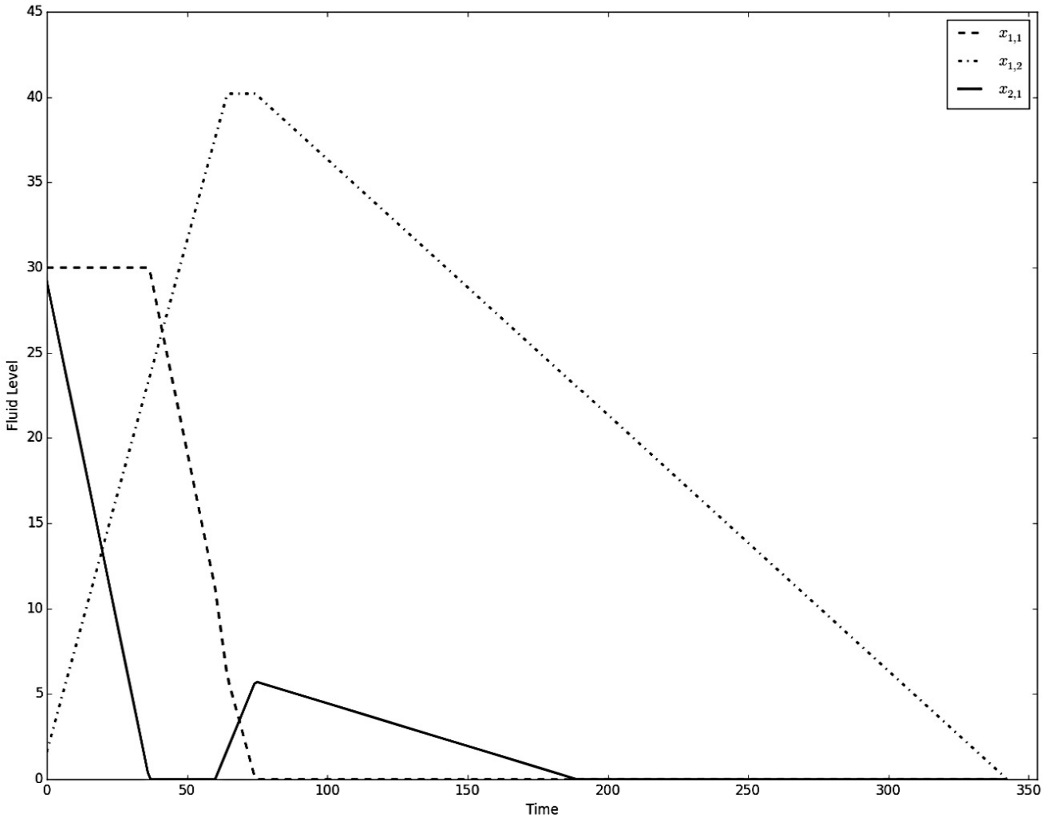


Figure 4.7: Example: The optimal state trajectory for an instance

The reason why it is optimal to build up the buffer of the class with highest cost lies in the relation between service rates. Note that in Policy π_1, in order to keep Buffer $(2, 1)$ empty, Station 1 has to work at a reduced capacity. It is only after Buffer $(1, 1)$ drains that the server at Station 1 is fully utilized. The trade–off comes from the fact that using the capacity of Station 1 at a higher rate accumulated less cost even though additional cost is incurred through letting Buffer $(2, 1)$ grow. Also, note that this result does not imply that idling in Station 1 is never optimal. In fact, for a fluid level $x_{2,1}$

high enough it can be optimal to idle the server at Station 1. Nevertheless, fluid build–up in Buffer $(2,1)$ occurs if the condition $c_{2,1} > c_{1,2}$ and $\mu_{1,1} > \mu_{2,1} > \lambda+\mu_{1,2}(1-\lambda/\mu_{1,1})$ is met regardless of whether previously the server at Station 1 had previously idled or not. □

Corollary 4.2.4. *Given $c_{1,1} < c_{2,1}$ and $\mu_{1,1} > \mu_{2,1}$,$\mu_{2,1} > \lambda+\mu_{1,2}(1-\lambda/\mu_{1,1})$, the Lagrange multiplier associated with class $(2,1)$ takes the following form:*

$$\dot{p}_{2,1}(t) = \begin{cases} -c_{2,1}, & \text{for } t < t_{2,1} \\ 0, & \text{otherwise.} \end{cases}$$

Proof. Lagrange multipliers should be in accordance with the phenomena described in Proposition 4.2.3. As a result, the definition of $p_{2,1}$ follows directly the proposition. □

Proposition 4.2.5. *Given $c_{2,1} > c_{1,1}, \lambda+\mu_{1,2}(1-\frac{\lambda}{\mu_{1,1}})$, Station 1 idles until $t_{2,1}$ and the Buffer $(2,1)$ never builds up.*

Proof. According to Proposition 4.2.3, Buffer $(2,1)$ can build up only if $c_{2,1} > c_{1,2}$ and $\mu_{1,1} > \mu_{2,1} > \lambda+\mu_{1,2}(1-\lambda/\mu_{1,1})$. It is then clear that the server at Station 1 never works at full capacity since it would result in the input rate of Station 2 exceeding its output rate. In terms of Lagrange multipliers this implies that $p_{2,1}(t) \geq \max(p_{1,1}(t), p_{1,2}(t))$, $\forall t \geq 0$.

Consider time $t = t_{2,1}$, it then follows that $p_{2,1}(t_{2,1}) \geq \max(p_{1,1}(t_{2,1}), p_{1,2}(t_{2,1}))$. For $t \leq t_{2,1}$, the fact that $\dot{p}_{2,1}(t) = -c_{2,1} < \max \dot{p}_{1,1}(t), \dot{p}_{1,2}(t)$ implies that $p_{2,1}(t) > \max p_{1,1}(t), p_{1,2}(t)$ and therefore the server at Station 1 idles until $t_{2,1}$. □

Corollary 4.2.6. *Given $c_{2,1} > c_{1,1}$, $\lambda + \mu_2(1 - \frac{\lambda}{\mu_1}) > \mu_{2,j} > \lambda$, the Lagrange multiplier associated with class $(2,1)$ takes the following form:*

$$\dot{p}_{2,1}(t) = \begin{cases} -c_{2,1}, & \textit{for } t < t_{2,1} \\ -c_{1,1}, & \textit{for } t \geq t_{2,1},\ t < t_{1,1} \\ -c_{1,2}, & \textit{for } t \geq t_{1,1},\ t < t_{1,2} \\ 0, & \textit{otherwise.} \end{cases}$$

Proof. The proof directly follows from Proposition 4.2.5. □

Proposition 4.2.7. *Given $c_{1,1} > c_{2,1} > c_{1,2}$, $\mu_{1,1} > \mu_{2,1} > \lambda + \mu_2(1 - \frac{\lambda}{\mu_1})$, idling at the server of Station 1 is only possible after Buffer $(1,1)$ drains.*

Proof. This result follows from the fact that given $c_{1,1} > c_{2,1}$, idling in Station 1 (while there is fluid in Buffer $(1,1)$) would incur holding cost at a greater rate in Station 1 than it drains at Station 2. However, this does not imply that idling in Station 1 is never optimal. In fact, after Buffer $(1,1)$ drains if under the policy that routes the incoming fluid to Buffer $(1,2)$ the station may idle. □

Corollary 4.2.8. *Given $c_{1,1} > c_{2,1} > c_{1,2}$, $\mu_{1,1} > \mu_{2,1} > \lambda + \mu_2(1 - \frac{\lambda}{\mu_1})$, the Lagrange multiplier associated with class $(1,1)$ takes the following form:*

$$\dot{p}_{1,1}(t) = \begin{cases} -c_{1,1}, & \textit{for } t < t_{1,1} \\ -c_{2,1}, & \textit{for } t > t_{1,1}, p_{1,1}(t) > p_{1,2}(t) \\ -\dfrac{c_{1,2}\mu_{1,2}}{\mu_{1,1}} + \dfrac{\dot{p}_{2,1}(\mu_{1,1} - \mu_{1,2})}{\mu_{1,1}}, & \textit{for } t > t_{1,1}, t < t_{1,2}, p_{1,1}(t) \leq p_{1,2}(t) \\ 0, & \textit{otherwise.} \end{cases}$$

Proof. This result follows from Proposition 4.2.7. $\square$

With Corollary 4.2.8 the structure of Lagrange multipliers is determined for the network under the assumption of $(c_{1,i}-c_{2,1})\mu_{1,i} > (c_{1,i+1}-c_{2,1})\mu_{1,i+1}$, $\forall i \in 1, 2$. For the model with $N_1 = 2$ our analysis showed interesting phenomena on the buffer of Station 2 draining and building–up. Furthermore, the fact that under general parameter settings, the optimal scheduling policy of Station 1 can be of dynamic nature. Therefore, a numerical procedure must evaluate a routing policy while considering the changes in scheduling priority. We hope to address these challenges in the future work.

Chapter 5

Conclusion

Complex networks have applications in many different domains such as biology, communications, production or service systems. Efficient control of such networks requires developing an accurate model of the actual system and efficient solution techniques for the associated control problem. Depending on the modeling assumptions, solving control problems can be highly challenging for large networks. In addition to determining the optimal policy, from a decision-maker's perspective, it is also important to obtain insights on why a policy is in fact optimal. For that reason, analytical results are more insightful than numerical ones and these insights become an important part in designing better systems in the future.

In that vein, a major focus of this dissertation is on obtaining analytical results on the optimal policies of particular classes of networks. Considering both queueing and fluid networks, we analyze networks composed of multiple classes of entities competing for service at the same station. Across all the models, the fraction of time that the flexible server spends on processing a particular class of entity is to be determined and the service rates depend on the particular class of entity in process. In addition, through a routing option that we call Gurvich-type routing, the class of an entity is also controlled. Without such a routing control, the networks we analyze in this dissertation correspond to multi-class queueing or fluid networks with flexible servers. For such networks, the optimal scheduling control can admit interesting properties. For example, in the single station case, the optimal scheduling control is based on a static priority-rule policy. Therefore one of the major themes in this dissertation is investigating whether similar results hold under Gurvich-type routing or not.

In Chapter 2, a queueing model composed of two parallel buffers and a flexible server is analyzed. The jobs arrive to the system according to a Poisson process and the decision-maker determines which queue to route a job upon arrival. The flexible server can process a single job at a time and the service time is assumed to be an exponential distributed random variable with a rate depending on class of job in service. It is assumed that jobs accumulate time-homogeneous class-dependent holding cost. The decision-maker is interested in minimizing the long-run average cost of the system through selecting routing and scheduling actions at each decision epoch. Through proofs based on the associated value function, structural results are obtained. The first such result is that the optimal scheduling policy follows a static priority rule, namely the $c\mu$ rule. Furthermore, it is shown that the optimal routing policy is of threshold-type. To approximate the threshold, the associated fluid network is considered. By solving the corresponding fluid optimization model and through perturbation analysis on the fluid model, we develop a threshold-policy to be implemented in the queueing model. Numerical studies show that the proposed policy is close to optimal in most of the parameter settings.

From a modeling stand-point, there is often a clear trade-off between a model's complexity and the computational effort required. Motivated by the performance of the fluid-based policy for the discrete model in Chapter 2, in Chapter 3 we consider a larger model than in Chapter 2 but only in the fluid context. The network under consideration is composed of a flexible server and N-parallel buffers with a general N. Again it is assumed that incoming fluid can be routed among N buffers, through Gurvich-type routing. Through an argument based on weighted workload, we first prove that a fluid equivalent of the $c\mu$ rule holds for the optimal scheduling control of the server. Via further proofs, insights on the structure of the optimal routing control policy is obtained. First, the optimal routing control is bang-bang. Second, the fluid classes that receive incoming fluid follow a strict order. For example, if at a given time the buffer corresponding to a class of fluid receives the incoming stream then at any time later a buffer associated with a higher holding cost can not receive any fluid. Through these insights, a procedure that computes the optimal controls and the state trajectory given an initial state is developed.

In contrast to single-station networks, in tandem networks the policy at a station has an immediate effect on downstream stations. The computation of the optimal policy then requires determining under which conditions stations idle. As a natural extension to the model in Chapter 3, in Chapter 4 we consider networks composed of two stations connected in tandem. The chapter is divided in two sections each dedicated to a different network model.

The first network we consider in Chapter 4 is composed of two stations where a flexible server is present for each station. The first station is composed of a single buffer whereas the second station is composed of N buffers in parallel. The output of Station 1 becomes the input to Station 2 and we assume that this stream of fluid is again to be routed among N buffers through Gurvich-type routing. For this model, the first important result we show is that the $c\mu$ rule is optimal for the scheduling control of Station 2. We then provide results regarding the optimal scheduling policy of Station 1 or the routing policy of Station 2. We prove that in some parameter regimes, the server at Station 1 idles until all buffers in Station 2 empty. Note that in that case no routing control decision affects the optimal state trajectory. Therefore, computing the full state trajectory can be computed easily through the static-priority policy. Also, it is shown that the server in Station 1 never idles under some parameter regimes. Except these cases, we prove that the optimal scheduling control of the server in Station 1 has a special structure. Under the optimal policy Station 1 can only admit at most one *idle period* that corresponds to a time interval where the server in Station 1 idles. With these results, we develop a procedure to compute the optimal control policies for both stations given any arbitrary initial state.

We then consider another class of tandem networks composed of two stations. The first station of this network is composed of two buffers in parallel and the second station is composed of a single buffer. Although this network is smaller than other fluid networks considered, it admits interesting properties. For example, under some parameter regimes, it is possible that the buffer in Station 2 first drains, then builds up and re-drains. More interestingly, this phenomena arises even in cases where the cost rate of the buffer in Station 2 is greater than the cost rates of the buffers in Station 1. Also for this network, the processing priority in Station 1 not only depends on terms related to station 1, but also on Station 2. We prove that a static priority rule holds for the optimal scheduling of the server in Station 1 under some restricted parameter settings. We again outline cases where the server of Station 1 idles and show how to derive the optimal policies. We conclude that chapter with a discussion of the same network with N-buffers in Station 1.

5.1 Future Work

A major focus of this dissertation is to provide theoretical proofs and insights on the structure of optimal policies for different classes of networks. Naturally, extending our results to larger classes of networks, or to networks under different topologies are future research directions. However, it must be noted that the structure of optimal policies get more complicated with consideration of multiple stations, as a result the proofs may be very challenging.

For the second network considered in Chapter 4, we proved that under restricted parameter regimes a strict index-based priority policy is in fact optimal for the scheduling control of the server in Station 1. Yet, under different parameter settings, the optimal scheduling control is indeed different.

For example, consider a case where the buffer in Station 2 is empty and it is associated with a larger service rate than the classes at Station 1. In this case, regardless of the scheduling rule of Station 1 - the buffer in Station 2 remains empty. Hence, the holding cost of the buffer in Station 2 plays no role in the scheduling control of Station 1. However, if the buffer in Station 2 is non-empty and for example, it is associated with a much higher holding cost than the ones in Station 1, surely the scheduling control of Station 1 must be affected. This simple case shows that in fact the optimal scheduling policy of Station 1 can be non-static. If that is the case, it is interesting to investigate how that affects the optimal routing policy of Station 1.

Another extension of our work is consideration of another class of two-station tandem networks where each station contains more than one buffer. This class of networks can be thought of a mixture of the networks considered in Chapter 4. For this class of network, the input to each station is routed among buffers via Gurvich-type routing. Therefore, the optimal scheduling and routing policies of each station might be highly dependent. Yet, we expect that under some parameter settings, the optimal policies might be easier to compute. For example, consider a case where the buffers in Station 1 have a higher cost rate than all of the buffers in Station 2. If that is the case, it is natural to expect that the server at Station 1 never idles. Similarly, if all the buffers have much smaller rate than the ones in Station 2 as well as high service rates, in that case, it is expected that the server at Station 1 idles until the buffers in Station 2 empty. Apart from these cases, we expect that the optimal policy might be harder to derive especially under parameter settings where the optimal scheduling rule of Station 1 is not based on a static priority rule. Investigating the interplay of policies of Station 1 and Station 2 is therefore a potential topic for future work.

Bibliography

[1] H. S. Ahn and M. E. Lewis. Flexible server allocation and customer routing policies for two parallel queues when service rates are not additive. *Operations Research*, 61(2):344–358, 2013.

[2] Z. Aksin, M. Armony, and V. Mehrotra. The modern call center: A multi-disciplinary perspective on operations management research. *Production and Operations Management*, 16(6):665–688, 2007.

[3] M. Armony. Dynamic routing in large-scale service systems with heterogeneous servers. *Queueing Systems*, 51(3-4):287–329, 2005.

[4] F. Avram. Optimal control of fluid limits of queuing networks and stochasticity corrections. In *Mathematics of Stochastic Manufacturing Systems*, volume 33, pages 1–37, 1997.

[5] F. Avram, D. Bertsimas, and M. Ricard. Fluid models of sequencing problems in open queueing networks; an optimal control approach. *Institute for Mathematics and its Applications*, 71:199, 1995.

[6] N. Bäuerle. Asymptotic optimality of tracking policies in stochastic networks. *Annals of Applied Probability*, pages 1065–1083, 2000.

[7] N. Bäuerle. Optimal control of queueing networks: an approach via fluid models. *Advances in Applied Probability*, 34(2):313–328, 2002.

[8] D. P. Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific Belmont, MA, 1995.

[9] H. Chen and A. Mandelbaum. Hierarchical modeling of stochastic networks, part ii: Strong approximations. In *Stochastic Modeling and Analysis of Manufacturing Systems*, pages 107–131. Springer, 1994.

[10] H. Chen and D. D. Yao. Dynamic scheduling of a multiclass fluid network. *Operations Research*, 41(6):1104–1115, 1993.

[11] D. R. Cox and W. Smith. *Queues*, volume 2. CRC Press, 1991.

[12] J. G. Dai. On positive Harris recurrence of multiclass queueing networks: A unified approach via fluid limit models. *The Annals of Applied Probability*, 5(1):49–77, 1995.

[13] J. G. Dai. *Stability of fluid and stochastic processing networks.* Centre for Mathematical Physics and Stochastics, University of Aarhus, 1999.

[14] J. G. Dai and J. H. Vande Vate. The stability of two-station multitype fluid networks. *Operations Research*, 48(5):721–744, 2000.

[15] L. Fleischer and J. Sethuraman. Efficient algorithms for separated continuous linear programs: the multicommodity flow problem with holding costs and extensions. *Mathematics of Operations Research*, 30(4):916–938, 2005.

[16] T. C. Green and S. Stidham. Sample-path conservation laws, with applications to scheduling queues and fluid systems. *Queueing Systems*, 36(1-3):175–199, 2000.

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

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

[19] A. Hordijk and G. Koole. The μc-rule is not optimal in the second node of the tandem queue: a counterexample. *Advances in applied probability*, pages 234–237, 1992.

[20] A. Hordijk and G. Koole. On the assignment of customers to parallel queues. *Probability in the Engineering and Informational Sciences*, 6(04):495–511, 1992.

[21] A. Hordijk and G. Koole. On suboptimal policies in multiclass tandem models. *Probability in the Engineering and Informational Sciences*, 10(01):29–39, 1996.

[22] J. Kingman. Two similar queues in parallel. *The Annals of Mathematical Statistics*, 32(4):1314–1323, 1961.

[23] G. Koole. Assigning a single server to inhomogeneous queues with switching costs. *Theoretical Computer Science*, 182(1):203–216, 1997.

[24] W. Lin and P. R. Kumar. Optimal control of a queueing system with two heterogeneous servers. *Automatic Control, IEEE Transactions on*, 29(8):696–703, 1984.

[25] C. Maglaras. Discrete-review policies for scheduling stochastic networks: Trajectory tracking and fluid-scale asymptotic optimality. *Annals of Applied Probability*, pages 897–929, 2000.

[26] A. Mandelbaum and A. L. Stolyar. Scheduling flexible servers with convex delay costs: Heavy traffic optimality of the generalized $c\mu$-rule. 52:836–855, 2004.

[27] S. Meyn. Sequencing and routing in multiclass queueing networks part i: Feedback regulation. *SIAM Journal on Control and Optimization*, 40(3):741–776, 2001.

[28] S. Meyn. Sequencing and routing in multiclass queueing networks. Part II: Workload relaxations. *SIAM Journal on Control and Optimization*, 42(1):178–217, 2003.

[29] S. Meyn. Dynamic safety-stocks for asymptotic optimality in stochastic networks. *Queueing Systems*, 50(2-3):255–297, 2005.

[30] S. Meyn. Stability and asymptotic optimality of generalized maxweight policies. *SIAM Journal on Control and Optimization*, 47(6):3259–3294, 2009.

[31] S. P. Meyn. Stability and optimization of queueing networks and their fluid models. *Lectures in applied mathematics-American Mathematical Society*, 33:175–200, 1997.

[32] J. A. Van Mieghem. Dynamic scheduling with convex delay costs: The generalized $c\mu$ rule. *The Annals of Applied Probability*, pages 809–833, 1995.

[33] Y. Nazarathy. *On control of queueing networks and the asymptotic variance rate of outputs*. PhD thesis, University of Haifa, 2008.

[34] Y. Nazarathy and G. Weiss. A fluid approach to large volume job shop scheduling. *Journal of Scheduling*, 13(5):509–529, 2010.

[35] W. B. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

[36] L. I. Sennott. *Stochastic dynamic programming and the control of queueing systems*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., New York, New York, 1999.

[37] A. Stolyar. On the stability of multiclass queueing networks. In *Proceeding of the 2nd International Conference on Telecommunication Systems–*

Modeling and Analysis, pages 23–35, Nashville, Tennessee, March 24–27 1994.

[38] T. Tezcan and J. G. Dai. Dynamic control of n-systems with many servers: Asymptotic optimality of a static priority policy in heavy traffic. *Operations Research*, 58(1):94–110, 2010.

[39] G. Weiss. On optimal draining of re-entrant fluid lines. *IMA volumes in mathematics and its applications*, 71:91–91, 1995.

[40] G. Weiss. A simplex based algorithm to solve separated continuous linear programs. *Mathematical Programming*, 115(1):151–198, 2008.

[41] W. Winston. Optimality of the shortest line discipline. *Journal of Applied Probability*, pages 181–189, 1977.