Optimal Server Allocation in Zero-Buffer Tandem Queues

Optimal Server Allocation

in

Zero-Buffer Tandem Queues

BY

MOHAMMAD HOSEIN YARMAND, M.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTING & SOFTWARE

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

PH.D.

Ph.D. (2012) McMaster University

(Computing & Software Engineering) Hamilton, Ontario, Canada

TITLE:	Optimal Server Allocation in Zero-Buffer Tandem Queues
AUTHOR:	Mohammad Hosein Yarmand
SUPERVISOR:	Dr. Douglas G. Down
NUMBER OF PAGES:	xiv, 213

To my parents,

Majid and *Masoumeh*.

Abstract

We study the server allocation problem for tandem queues in the absence of intermediate buffer space. Servers are assumed to be homogeneous and non-collaborative. We provide policies to maximize the throughput. We break down our work into four stages. First, we assume that all servers are dedicated. We propose an allocation algorithm that assigns servers to stations based on the mean service times and the current number of servers assigned to each station. The algorithm is proposed for stations with exponentially distributed service times, but where the service rate at each station may be different. We further study the impact on the proposed allocation method of including service time distributions with different coefficients of variation. Second, we consider tandem queues with both dedicated and flexible servers. We examine policies to dynamically assign flexible servers. When there is one flexible server and two stations each with a dedicated server, we completely characterize the optimal policy. We use the insights gained from applying the Policy Iteration algorithm on systems with three, four, and five stations to devise heuristics for systems of arbitrary size. Third, we study cases where flexibility is constrained such that flexible servers can only service two adjacent stations. We provide optimal policies for tandem queues with three and four stations and compare them with optimal policies for corresponding non-constrained cases. Fourth, we consider two parallel tandem queues with both dedicated servers and vertical flexible servers - servers that can move between corresponding stations of the two tandem queues. The workload allocations are the same for each line and each vertical flexible server moves only between two corresponding stations. We examine policies for dynamic allocation of these vertical flexible servers. When each tandem queue has two stations, each station possesses a dedicated server,

and a vertical flexible server exists for each pair of stations, we specify the optimal policy. For cases with more than two stations, heuristic assignments are proposed. We also analyze the throughput improvement gained from adding flexible servers within a tandem queue or between two parallel tandem queues. Numerical results are used to verify the heuristics provided in each of the stages.

Acknowledgements

I want to take this opportunity to express my thanks to Dr. Douglas Down for his generous guidance and continuous support over the years. Dr. Down has been a great role model for me and has shown me the joy of doing good research. During this research, he has taught me valuable academic skills that will certainly be beneficial to my future performance, and I am forever indebted to him for this.

Also, special gratitude to Dr. Kamran Sartipi who aided me through the initial stages of this research. I would also like to thank my Supervisory Committee members, Dr. Norm Archer and Dr. Frantisek Franek for their help and time that they put on reviewing this thesis. I also wish to thank many friends and relatives who were always there for me during the hard moments I had.

Contents

List of Tables

List of Figures

Glossary

F	The total number of flexible servers in a tandem line, 41
M	The total number of available servers, 8
N	the number of stations in a tandem line, 5
Q	The total available buffer space, $Q = \sum_{i=1}^{N} q_i$, 8
$R(s)$	The throughput in terms of the given server allocation vector (s), 14
W	The total workload, equal to the summation of the mean service times of all stations, $W = \sum_{i=1}^{N} w_i$, 8
$\bar{F}$	$\bar{F} = (f_1, \ldots, f_i, \ldots, f_N)$ is the allocation vector of the vertical flexible servers, where f_i is the number of vertical flexible servers available for the i^{th} pair of stations in two parallel tandem lines, 152
μ_i	the service rate of the i^{th} station, 5

cv $cv = (cv_1, \ldots, cv_i, \ldots, cv_N)$ is the vector of coefficients of variation, where cv_i is the coefficient of variation of the i^{th} station, 35

q $q = (q_1, \ldots, q_i, \ldots, q_N)$ is the vector of buffer allocations, where q_i is the buffer space allocated to the i^{th} station, 8

w $w = (w_1, \ldots, w_i, \ldots, w_N)$ is the vector of mean service times, where w_i is the mean service time at the i^{th} station; $w_i = \frac{1}{\mu_i}$, 8

Chapter 1

Introduction

Tandem queues are used to model scenarios in which a job needs to go through a set of consecutive steps. Jobs departing from a queue (alternatively called a station) arrive to the next queue in the line and leave the system after visiting the last queue. Tandem queues have been extensively applied in various fields. These fields include, but are not limited to, production and assembly lines, material handling systems, telecommunications (for example, packet or circuit switched networks), service centers, and healthcare systems.

Different design issues in these fields are: I) determining service times of stations, given the total service time for a job, II) allocating servers to stations, III) determining buffer sizes between stations. Researchers have studied these problems both separately and together (e.g. simultaneous allocation of servers and buffer spaces).

In this work we focus on tandem queues with no intermediate buffer spaces between stations, so called zero-buffer tandem queues. We aim to find server allocations which maximize the throughput. Our work can be broken down to the following stages:

i. Given a number of servers, we first propose an allocation algorithm, when all servers are dedicated, meaning that they stay in the station to which they are assigned (Chapter 2).

ii. We further improve the throughput by introducing flexible servers, which are servers that do not belong to any specific station and can work at all stations. We explore policies which determine how a flexible server should be dynamically allocated to maximize the throughput (Chapter 3).

iii. We then study situations where flexibility is constrained, meaning that servers can move between neighbouring stations rather than travelling through all stations. We report how the policies from stage ii should be modified to respect these additional constraints (Chapter 4).

iv. When two identical parallel lines are considered, we introduce vertical flexible servers, which are servers that can be shared among corresponding stations of the parallel lines. We specify dynamic server allocation policies that maximize the throughput (Chapter 5).

Now we explain how the stages above are related. Given a server allocation for dedicated servers (from stage i), we observe the throughput improvement, when dedicated servers are changed to flexible servers (stage ii) (Section 3.6). We then contrast the policies and throughput values when flexibility is constrained (stage iii) vs. non-constrained (stage ii) (Section 4.1.4). We also compare the throughput improvement of making dedicated servers vertically flexible (stage iv), with making servers flexible within a single line (stage ii) (Section 5.5). We use these observations to describe system design decisions in Section 7.1.

The server allocation problem in zero-buffer tandem lines is motivated by the bed management issue in hospitals. In short, bed management is the problem of assigning a number of beds to different departments of a hospital, such that patient flow is optimized. Patients need to go through these departments to complete their treatment cycle. The fact that patients must be assigned to a bed at all times, represents the zero-buffer nature of this problem. Other applications are discussed in Chapter 6.

It is possible that due to higher loads, certain departments would require more beds. In case other departments have idle beds, these idle beds can be reconfigured according to the target department requirements (if necessary) and shared between the two departments. This motivates including flexible servers among stations. Policies would be required to govern the allocation of these flexible servers to the stations.

When more than one hospital is considered, it is desirable to optimize patient flow over all involved hospitals. Hospitals with similar treatment cycles for a specific disease, could share beds between corresponding departments (which have similar beds). In the associated tandem queue model, we assume two parallel tandem lines with similar stations and allow flexible servers to move among corresponding stations between the two tandem lines.

A detailed discussion of our contributions and a literature review is given in each of the chapters of this thesis. However, here we provide a brief list of our major contributions.

- For dedicated servers, we provide an allocation algorithm for heterogeneous lines with arbitrary service time distributions, arbitrary number of stations, and arbitrary number of servers. The problems that have been considered elsewhere

are different from ours, as they have assumed one or more of the following: buffers exist, the workloads are balanced, workload allocation is also a control parameter, or that there is a limited number of extra servers to assign (assuming the line is balanced).

- For flexibility in a single line, we propose optimal policies for non-collaborative servers and in the presence of dedicated servers, such that the throughput is maximized. Other researchers have assumed one or more of the following: buffers exist, servers are collaborative, a single server exists per station, or that the goal is to minimize the cost (rather than maximizing the throughput).

- For vertical flexibility in parallel lines, given that dedicated servers are allocated, we describe optimal policies to dynamically assign servers, such that the throughput is maximized. We could not find any other work that explores a similar problem.

1.1 Reference model

Here, we present the reference model used in the chapters to follow. Consider a tandem line consisting of N stations ($N \geq 2$) where the service rate of a server assigned to station i is μ_i ($i = 1, 2, ..., N$). The service times at each station follow exponential distributions and are independent and identically distributed with rate μ_i (i.e. the rate can depend on the station). In Chapter 2, we extend the model to include stations with more generic service time distributions. The servers are capable of working at any station and can process only one job at a time. The servers are homogeneous, meaning that servers assigned to the i^{th} station each work at rate μ_i.

When more than one line is considered, the service rate of each server working at the i^{th} station in line $k \in \{1, 2\}$ is labelled μ_{ki}.

We assume there is an infinite supply of jobs in front of the first station and infinite space for jobs completed at the last station (i.e. jobs served at the last station immediately leave the system). At any given time, each station can be assigned multiple servers and each server can work only on one job, and a job may have at most one server assigned to it. The throughput of the system is equal to the departure rate from the last station. We assume the travel times for jobs to progress are negligible. Finally, we assume there are no buffers between stations.

1.2 Thesis overview

A brief description of the contents of each chapter have been discussed above, but here we provide further detail on the contents of the thesis.

In Chapter 2, we study server allocation to tandem queues in the absence of buffer spaces. Servers assigned to a station, work only at that station. We propose an allocation algorithm for tandem queues with heterogeneous mean service times that maximizes system throughput. We examine tandem lines that have service times with different coefficients of variation. Several simulations are performed to evaluate the proposed algorithm.

In Chapter 3, we investigate tandem queues in the presence of both dedicated and flexible servers. We propose policies that specify the movement of flexible servers, such that the throughput is maximized. With two stations, we show that the optimal policy is non-idling and specify it for arbitrary mean service times. With 3, 4, and 5 stations, we express the optimal policy for tandem lines with specific service times.

For tandem lines with heterogeneous service times, simulations are used to compare a number of proposed allocation policies and to indicate a near optimal policy. We examine tandem lines that have service times with different coefficients of variation.

In Chapter 4, we analyze tandem lines that have constrained flexible servers that can only move between two adjacent stations. We identify optimal policies for systems with three and four stations. We then compare these optimal policies with optimal policies for tandem lines in which flexibility is not constrained.

In Chapter 5, we look into two parallel tandem queues in the presence of both dedicated and flexible servers between the two tandem lines. With two stations, we show that the optimal allocation policy which maximizes throughput is non-idling and specify it for arbitrary mean service times. We compare the benefit gained from adding flexible servers in a tandem queue versus adding flexible servers between two tandem lines. We also discuss the order in which flexible servers should be added to tandem lines with dedicated servers only, such that the throughput improvement is maximized at each step.

In Chapter 6, we provide several real world applications of our research. We indicate how the elements of each case correspond to our model.

Chapter 7 concludes the thesis, presents several potential directions for future work, and includes a short discussion on system design decisions.

Chapter 2

Server allocation for zero-buffer tandem queues

2.1 Introduction

Consider the model introduced in Section 1.1. Assume there are M servers available to be allocated to the stations. Our problem of interest is: *given M servers, allocate them to the N stations, such that the throughput is maximized.* We could define a similar problem in terms of blocking and starvation probabilities. In that case, the goal would be to minimize an aggregated measure of these probabilities over all stations.

We propose an algorithm that has as a primary goal to roughly equalize the workloads at each of the stations, meaning that the number of servers is proportional to the mean service time at a station. However, the heterogeneous mean service times and lack of buffers introduce additional complexity beyond making the workloads equal (further discussed in Section 2.2). We use simulation to obtain insights about

the nature of the system and later to measure the performance of our algorithm on a number of configurations. In addition to exponentially distributed service times, we extend the algorithm by considering service times with coefficients of variation other than 1. *Coefficient of variation* is a normalized measure of scattering of a distribution and is defined as the ratio of its standard deviation to its mean. We illustrate that the algorithm performs well if the coefficients of variation of all stations are increased or decreased equally. The algorithm also works well on configurations where the majority of stations have service times with coefficient of variation near one and the remaining stations have service times with coefficient of variation less than one.

Optimization modeling is typically used to formulate general allocation problems in this research domain (see (Hillier and So, 1995), for example). Throughput is denoted by $R(q, s, w)$, where $q = (q_1, q_2, ..., q_N)$, $s = (s_1, s_2, ..., s_N)$, and $w = (w_1, w_2, ..., w_N)$ denote the allocation of buffers, servers, and workload to stations, respectively. Q is the total number of available buffer spaces and W is the total mean service time over all stations. The optimization problem is expressed as:

maximize $R(q, s, w)$

subject to

$$\sum_{j=2}^{N} q_j = Q,$$

$$\sum_{j=1}^{N} s_j = M,$$

$$\sum_{j=1}^{N} w_j = W,$$

$q_j \in \mathbb{N}, j \in \{2, 3, \ldots, N\}$,

$s_j \in \mathbb{N}, j \in \{1, 2, \ldots, N\}$,

$w_j > 0, j \in \{1, 2, \ldots, N\}$,

where $q, s,$ and w are decision vectors (q has entries q_j, etc.). Note that workload allocation (w) is the problem of determining the mean service time at each station, given that the mean service times sum to a fixed value.

Hillier and So (1989) aim to maximize throughput for tandem queues with equal workloads (w_i equal for all i) and small or no buffers ($q_i = 0$ or 1). They claim the optimal server allocation (s) assigns extra servers rather uniformly to the interior stations and refine this claim based on the number of servers and stations at hand. They introduce the *bowl phenomenon*: with single server stations, different service times, and equal buffers, the optimal workload allocation (w) assigns less work to the interior stations than to the end stations. It appears that the interior stations (especially the center stations) are critical in determining system performance and so should be given preferential treatment when making design decisions.

Hillier et al. (1993) define the *inverted bowl phenomenon*: when the total amount of storage space is a decision variable and workloads are equal (w_i equal for all i), the optimal buffer allocation (q) commonly follows an inverted bowl pattern. In other words, the allocation provides the stations toward the center of the line with more buffer storage space than the other stations.

Hillier and So (1996) also introduce the *L phenomenon*: simultaneous server and workload allocation (s, w) results in maximized throughput by assigning all extra servers (beyond the initial server at each station) to just one of the end stations and then adjusting the work allocation so that the end station has by far the greatest amount of work per server.

Shanthikumar and Yao (1987) consider a closed queueing network. In order to maximize throughput when allocating servers, they allocate each additional server to the station which causes the greatest increase in the difference between that station's profit (as a function of its throughput) and cost (as a function of servers allocated to it).

Spinellis et al. (2000) aim to maximize throughput in a queueing network with a fixed total service time with buffers (W, Q) and exponential service times. For a large number of stations, the server allocation (s) tends to accumulate towards the beginning of the line, and the buffer allocation (q) tends to accumulate towards the end.

Cheng and Zhu (1993) state that when assigning M heterogeneous servers to M stations with no buffer between the first two $(q_2 = 0)$ (resp. the last $(q_M = 0)$) stations and possible buffers for interior stations, it is better to allocate the slower server to the first (resp. the last) station.

Magazine and Stecke (1996) consider a three station tandem queueing system with no buffers $(q_i = 0)$. They follow the results of Hillier and So (1989) and as the number of servers increases, the unbalancing in favour of the middle station is increased. This behavior continues until the unbalancing becomes too severe. At this point, a server is taken away from the middle station and a server is added to the first and third stations. They also state that if unbalancing and distributing servers (w, s) are left to our control, both should be as balanced as possible.

Van Woensel et al. (2010) move a step further and consider any possible acyclic multi-server configuration with arbitrary service and inter-arrival time distributions.

They model the joint buffer and server allocation problem (q, s) as a non-linear optimization problem with integer decision variables. They use the Generalized Expansion Method to evaluate throughput. They further use Powell's algorithm (detailed in (Himmelblau, 1972)) for allocation purposes.

The problem we consider is different in the following respects. The models in (Cheng and Zhu, 1993; Hillier et al., 1993; Spinellis et al., 2000; Van Woensel et al., 2010) include buffers in their configurations. Hillier and So (1996) perform workload allocation together with server allocation. Shanthikumar and Yao (1987) assume certain constraints on arrival rates (the arrival rate depends on the number of jobs in the station). Hillier and So (1989) consider tandem queues with small buffers and perform simulations for the case with no buffers. They assume that workload is balanced and the numbers of servers at stations differ by at most two (i.e. there is a limited number of extra servers). In other words, starting from a balanced system, they study how to allocate extra servers. We will apply their allocation method to more generic cases to discover its potential shortcomings. The tandem line that Magazine and Stecke (1996) target is limited as all rates are equal and there are only three stations. Our problem of interest assumes a configuration with no buffers where we only have control over server allocation. There are no restrictions on the number of stations or their service rates.

Andradóttir et al. (2003) study server allocation (s) in infinite buffer settings $(q_i = \infty)$ with flexible servers using a linear programming approach. We would like to contrast the two extremes (in terms of buffer sizes) in tandem lines for allocation of dedicated servers. Namely, in Section 2.2 we compare our configuration of interest (zero-buffer) with a configuration with infinite buffers between stations.

There has also been work done on the effect of variability of service times for tandem lines. Hillier and Boling (1979); Hillier and So (1993) consider workload allocation (w) for a system with equal coefficients of variation and find that the bowl phenomenon holds but the depth of the bowl changes with coefficient of variation. El-Rayah (1979) studies the optimal arrangement of single server tandem lines (s) with no buffer spaces ($q_i = 0$) and where servers have different coefficients of variation. They discover that assigning servers with higher coefficients of variation to the exterior stations leads to higher throughput. Muth and Alkaff (1987) study the effect of independent changes in the mean service time and the service time variance on a tandem line's throughput. They study single-server tandem lines with three stations and no buffers and offer a method to compute the throughput. Heavey et al. (1993); Papadopoulos et al. (1989, 1990) examine specific production lines (with feedback or unreliable stations) by generating sparse transition matrices and solving them using the Successive Over Relaxation (SOR) method. They consider single-server tandem lines with finite buffers and Erlang or exponential service times.

Andriansyah et al. (2010) study zero-buffer multi-server general queueing networks. They use the Generalized Expansion Method to evaluate the throughput for a class of acyclic networks. They employ genetic algorithms to solve a multi-objective optimization problem to provide the trade-off between the total number of servers used and the throughput. Our work in comparison, focuses on the details of server allocation for the specific case of tandem queues. Our work focuses on the case where throughput is to be maximized (by assuming an infinite amount of work at the first station), while Andriansyah et al. (2010) focus on a system with arrivals, with better results achieved when the arrival rate is somewhat below the maximum possible

throughput.

Futamura (2000) studies the effect of service time variability in systems with and without buffers. They suggest that a station with s_i servers is similar to a station with a single server with a rate s_i times as large but with coefficient of variation closer to 1. Also, adding servers to a station adds effective buffer spaces to the station. Using these results, Futamura suggests that server allocation should follow the inverted bowl phenomenon except that more servers are assigned to stations with higher coefficient of variation to alleviate the impact of higher variance. Simulations are performed for configurations with balanced workloads and a limited number of extra servers.

This chapter is structured as follows. In Section 2.2 we model the server allocation problem and propose an algorithm to perform the allocation, given the service rates. In Section 2.3 we carry out simulations on a number of configurations and analyze the performance of the proposed algorithm. In Section 2.4 we explain how to modify the algorithm when the coefficients of variation of some or all of the stations are altered.

2.2 Server allocation method

The generic optimization problem (stated in the Introduction) refined to our specific problem becomes:

maximize $R(s)$

subject to

$$\sum_{j=1}^{N} s_j = M,$$

$q_j = 0, j \in \{2, 3, \ldots, N\}$,

$s_j \in \mathbb{N}$ and $j \in \{1, 2, \ldots, N\}$,

$w_j = \frac{1}{\mu_j}, j \in \{1, 2, \ldots, N\}$.

In order to design an allocation algorithm, we identify different parameters affecting throughput. An optimal allocation might want to 1) clear blocking, 2) avoid starvation, and/or 3) improve bottlenecks. These quantities are correlated in the following manner: blocking probability increases when the *total service rate* of a station (which is equal to the throughput of a station if it were the only station) is higher than its subsequent station; a station that causes blocking rather more frequently than other stations is a bottleneck; a blocking or bottleneck station can cause starvation in downstream stations.

The idea is to balance the total service rate for all stations. In this way, the allocation prevents a station from becoming a bottleneck. Also blocking and starvation probabilities are decreased as there are no bottlenecks in the allocation. Note that the system throughput, i.e. the throughput of the last station, is dependent on the throughput of the previous stations. Therefore its throughput could become limited by the throughput of the bottleneck station (amongst the upstream stations). For

that reason we intend to use an allocation algorithm that prevents introducing bottleneck stations by balancing total service rates. Algorithm 1 is our proposed allocation method.

First consider the following definitions. Whenever a server is allocated to a station, we say the allocation method has *visited* that station. For a given station, the total number of visits to other stations since the last visit to the given station is called the *visit distance* for that station. The visit distance that an allocation method tries to enforce for each station is called the *visit period* of that station.

In Algorithm 1, p_i is the visit period of station i, equal to the ratio of the sum of the mean service times of all stations (W) to the mean service time of station i. This equality is suggested by our simulation results. This is also compliant with our goal of balancing total service rates, as the total service rate of each station is proportional to the number of servers working at that station. The quantity l_i is the visit distance for station i. The algorithm assumes that the entries in w are not equal. The case where the entries in w are equal has previously been considered, for example in Hillier and So (1989); Magazine and Stecke (1996); Spinellis et al. (2000).

Our original conjecture was that the optimal allocation balances the total service rates ($s_i\mu_i$) of all stations. In Algorithm 1, lines 6 - 8 perform this task. The algorithm starts from a situation where one server is allocated to each station and upon allocation of the next server:

- assigns the next server to the station with the smallest value of $s_i\mu_i$ (total service rate);
- in cases where $\exists i, j.\ s_i\mu_i = s_j\mu_j \wedge i \neq j$, chooses the station with higher service rate. (If service rates are equal, follows (Hillier and So, 1989) by assigning extra

servers uniformly to the interior stations.)

Algorithm 1 Allocation Algorithm (M, N, w)

1: $W = \sum_{i=1}^{N} \frac{1}{\mu_i}$
2: $p_i = W\mu_i$
3: $\forall i, s_i = 1$
4: $\forall i, l_i = 0$
5: **while** $\sum_{i=1}^{N} s_i \leq M$ **do**
6: $t =$ indices of stations where $\forall k, k \neq t \ . \ s_t\mu_t < s_k\mu_k$
7: **if** t is not unique **then**
8: $t =$ values from line 6 that minimize $|t - \frac{N}{2}|$
9: **end if**
10: **if** t is not unique **then**
11: $t =$ a random value from line 8
12: **end if**
13: $l_t = 0$
14: $\forall i, i \neq t \ . \ l_i = l_i + 1$
15: **while** $((\exists j \ . \ l_j \geq \lfloor p_j \rfloor) \wedge (j \in$ "high priority stations"$))$ **do**
16: $s_j = s_j + 1$
17: $l_j = 0$
18: $\forall i, i \neq j \ . \ l_i = l_i + 1$
19: **end while**
20: **if** $\sum_{i=1}^{N} s_i \leq M$ **then**
21: $s_t = s_t + 1$
22: **end if**
23: **end while**

Our original conjecture ignored the effect of zero buffers, i.e. we thought in terms of a line with infinite buffers between stations. To guarantee a stable system, we need that for all stations the rate of jobs arriving to a station be equal to the rate of jobs departing from that station. However, we know that the maximum departure rate of a station is less than or equal to its total service rate. Given M servers and N stations we consider the following linear programming problem:

maximize $R(s)$

subject to

$s_i\mu_i \geq R(s),\ \forall i \in \{1, ..., N\},$

$\sum_{i=1}^{N} s_i = M.$

The linear programming problem has the solution $R^* = \max_s\{\min_i\{s_i\mu_i\}\}$. This expression is consistent with results suggested by Andradóttir et al. (2003), who consider a more general network topology and flexible servers.

This expression for the optimal throughput immediately leads to a greedy allocation algorithm that always helps the station with the smallest value of $s_i\mu_i$, the same as our original conjecture. However, simulation results revealed that this allocation method alone could lead to allocations that are far from optimal.

We observed that a method that solely balances $s_i\mu_i$ for all stations, could lead to cases where consecutive servers are assigned to a specific station. In particular, this happens when the service rate of a station is low compared to other stations. In such a case, the method allocates consecutive servers to that station to compensate for the low value of $s_i\mu_i$. Numerical results illustrating this fact are included in Section 2.3.

We simulated server allocation for a number of configurations to recognize properties of optimal allocations. We found that *each station should be visited with a certain period*. In addition, some configurations include a set of stations which we call "high priority stations" (detailed below). If such a set exists, we might need to change the order of allocation to satisfy the following constraint: *visit distances for "high priority stations" should be less than or equal to their visit periods*. In other words, it might be necessary to prioritize a station in this set by postponing visits to stations not belonging to this set, so that the above constraint is satisfied. A result of respecting this constraint is that optimal allocations avoid the behavior described in the last paragraph, i.e. servers are not assigned consecutively to stations with lower service rates. Lines 9 - 13 of the algorithm implement this constraint.

While we do not specify "high priority stations" completely, we give guidelines on how to choose them. For example, consider the configuration with mean service time vector $w = (5, 4, 2, 9, 3, 8, 7, 1, 6)$. $W = 45$ for the given configuration. The "high priority stations" are stations with mean service times 1, 2, 3, and 4. In this example we consider all stations with a mean service time less than 5 a member of the "high priority stations" set. Also, we have that the ratio of the sum of mean services times of the set's members to W is equal to $\frac{1+2+3+4}{45} = \frac{10}{45} = 0.222$. We experimentally identified that the members of the set should be chosen so that this ratio is close to 0.2. As another example, for a system with mean service time vector $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$, "high priority stations" are stations with mean service times 1, 3, 4, and 5. $W = 64$ for this configuration. We have that the required ratio is equal to $\frac{1+3+4+5}{64} = \frac{13}{64} = 0.203$. Note that if we let stations with a mean service time less than 10 belong to this set, we would have $\frac{1+3+4+5+7+9}{64} = \frac{29}{64} = 0.453$

which makes the ratio too big and hence we do not consider the second and eighth stations as "high priority stations".

For a station to belong to this set, it is necessary that its mean service time is sufficiently small, in the sense that it is less than a proportion of mean service times of stations with higher mean service times. However, this is not sufficient. The number of stations with lower and higher mean service times should be considered, in the following sense. The sum of mean service times of stations with lower mean service times over the total workload is an important proportion. If this proportion is too big, a set of "high priority stations" does not exist. An example is where all stations have the same mean service times except one station, which has a higher mean service time.

The algorithm is not particularly sensitive to the choice of the set of "high priority stations". If it is not obvious if a station belongs to the set, it does not make much difference if it is counted as a "high priority station" or not. The reason is that counting the station as a member of the set results in better allocation for some values of M and worse allocation for some other M values. For example, with $w = (5, 4, 2, 9, 3, 8, 7, 1, 6)$, it is not clear whether the first station should belong to the set or not. For instance, if this station is considered as a member of the set, the algorithm results in higher throughput for $M = 58$. If it is not considered a member, the algorithm results in higher throughput for $M = 69$.

Trying to comprehend the need to use "high priority stations", we notice the effect of having multiple servers at a station. In an infinite buffer setting we only care about the product $s_i\mu_i$ and not the individual terms, i.e. s_i and μ_i. For example, 5 servers each working at rate 2 offer the same throughput as 1 server working at

rate 10. However, in a zero-buffer setting, a larger number of servers at a station leads to higher throughputs. For example, 5 servers working at rate 2 perform better than 1 server working at rate 10. The reason is that with no buffers, servers act as buffers when blocking occurs. Therefore more servers provide artificial buffer space that helps reduce blocking.

As we intend to balance $s_i\mu_i$ among all stations, a station with higher mean service time is assigned more servers compared to a station with lower mean service time. However, as stated above, having more servers improves the throughput of a station. Therefore, in order for a low mean service time station to be able to admit multiple jobs coming from high mean service time stations, more servers should be assigned to the low mean service time station. Using the concept of "high priority stations" leads to allocations that take the multiplicity effect into account by making visit distances not bigger than visit periods for such stations.

To gain a better understanding about the multiplicity effect, we present a number of analytical results to support the belief that assigning multiple slow servers to a station leads to better performance than allocating a single fast server to the station (with equal total service rate). This belief is also consistent with the way we decide the server allocation order between stations with equal total service rate but different numbers of servers.

Note that replacing fast servers with slow servers could lead to situations where a station is working at a slower rate. More specifically, at a station with fast servers replaced by slow servers, when there are less jobs than servers, the departure rate from the station is reduced compared to the case with fast servers. However, the analysis below shows that the trade-off between the increase in buffer size and the

potential reduction in total service rate should always be resolved in favour of gaining extra buffer spaces.

Proposition 2.1. *In a tandem line with two stations and one server per station, the throughput increases if the server at the first station is replaced by a number of slow servers, preserving the total service rate at the first station.*

Proof: Assume the server at the first station (working at rate μ_1) is replaced by n servers (working at rate $\frac{\mu_1}{n}$). Let the server at the second station work at rate μ_2. We solve the Markov chain corresponding to this system. Let $p_{w^n i}$ represent the probability of being in a state with n busy servers at the first station and an idle server at the second station. Also, set $\mu = \frac{\mu_1}{\mu_2}$. We have

$$p_{w^n i} = \frac{1}{1 + \mu + \mu^2 + \sum_{j=1}^{n-1} \frac{n-j}{n} \mu^{j+2}}.$$

The throughput of the system is equal to $\mu_2(1 - p_{w^n i})$. Hence the throughput increases as n is increased.

(Proposition 2.1) □

Proposition 2.2. *In a tandem line with two stations and one server per station, the throughput increases if the server at the second station is replaced by a number of slow servers, preserving the total service rate at the second station.*

Proof: Assume the server at the second station (working at rate μ_2) is replaced by n servers (working at rate $\frac{\mu_2}{n}$). Let the server at the first station work at rate μ_1. We solve the Markov chain corresponding to this system. Let p_{bw^n} represent the

probability of being in a state with a blocked server at the first station and n busy servers at the second station. Also, set $\mu = \frac{\mu_2}{\mu_1}$. We have

$$p_{bw^n} = \frac{1}{1 + \mu + \mu^2 + \sum_{j=1}^{n-1} \frac{n-j}{n} \mu^{j+2}}.$$

The throughput of the system is equal to $\mu_1(1 - p_{bw^n})$. Hence the throughput increases as n is increased.

(Proposition 2.2) □

Proposition 2.3. *In a tandem line with three stations and one server per station, the throughput increases if any of the servers is replaced by two slower servers working at half of the rate of the original server.*

Proof: We only provide the proof for the case where the server at the third station is replaced by two slower servers. When there is a single server at each station, the states are:

$$\{wii, wwi, wiw, www, wbw, bwi, bww, bbw\},$$

where w, i, and b stand for a working, idling, and blocked server, respectively. The throughput is equal to $T = \mu_3(1 - (p_{wii} + p_{wwi} + p_{bwi}))$. When the third station's server is replaced by two slower servers, the states are:

$$\{wiii, wwii, wiwi, wwwi, wiww, wwww, bwii, bwwi, wbww, bbww, bwww\},$$

where the last two letters of each state correspond to the two servers at the third station. The throughput is equal to:

$T' = \mu_3(p_{wiww} + p_{wwww} + p_{wbww} + p_{bwww} + p_{bbww}) + \mu_3(p_{wiwi} + p_{wwwi} + p_{bwwi})/2.$

Evaluating $T' - T$ simplifies to:

$\mu_1^4\mu_2^3\mu_3^3(\mu_1+\mu_2)(\mu_1^2+\mu_1\mu_2+\mu_2^2)^2(\mu_1+\mu_2+2\mu_3)(4\mu_1^6+12\mu_1^5\mu_2+20\mu_1^5\mu_3+12\mu_1^4\mu_2^2+$
$48\mu_1^4\mu_2\mu_3+41\mu_1^4\mu_3^2+4\mu_1^3\mu_2^3+44\mu_1^3\mu_2^2\mu_3+84\mu_1^3\mu_2\mu_3^2+44\mu_1^3\mu_3^3+26\mu_1^2\mu_2^3\mu_3+77\mu_1^2\mu_2^2\mu_3^2+$
$78\mu_1^2\mu_2\mu_3^3+26\mu_1^2\mu_3^4+16\mu_1\mu_2^4\mu_3+52\mu_1\mu_2^3\mu_3^2+65\mu_1\mu_2^2\mu_3^3+37\mu_1\mu_2\mu_3^4+8\mu_1\mu_3^5+4\mu_2^5\mu_3+$
$16\mu_2^4\mu_3^2+25\mu_2^3\mu_3^3+19\mu_2^2\mu_3^4+7\mu_2\mu_3^5+\mu_3^6),$

which is clearly positive, allowing us to conclude $T' > T$. The cases where the first or second stations are replaced with slower servers are proved similarly.

(Proposition 2.3) □

We considered several configurations with two stations and more than one server at each station (e.g. $s = (2, 3)$). The multiplicity effect holds for them. However, we found it difficult to show the effect for an arbitrary number of servers at the two stations. When the number of stations is increased (in a single-server setting), validating the multiplicity effect becomes a complex algebraic exercise. However we believe this effects holds in general.

Hillier and So (1989) adapt results for the single server setting to analyze multi-server settings. They state that using s_i servers working at rate μ_i at a station is almost equivalent to employing a single fast server working at rate $s_i\mu_i$ with $s_i - 1$ buffer spaces. The multiplicity effect is consistent with this argument. Having multiple servers introduces buffer spaces which in turn increases throughput by reducing blocking.

2.3 Simulation results

We have simulated a number of configurations with different numbers of stations and servers. As the numbers of servers and stations increase, the number of possible allocations grows so dramatically that it becomes impractical to find the optimal allocation by simulation. Given M and N and assuming each station has at least one server, the number of possible combinations is N^{M-N}, which illuminates the exponential nature of the search space. In order to be able to simulate the problem, we need to reduce the size of the search space to a manageable size, i.e. filter the set of combinations that we consider. Each run of the simulation consists of tracking the system until a certain number of departures has occurred and then calculating system throughput.

Assume that the throughput values for all combinations of a configuration with given N, M, and w are known. Now we want to simulate the system for $M+1$ servers. Call the combinations leading to the highest throughput values *top combinations*. In other words, if combinations are ordered based on their throughput values, those appearing at the top of the list are the *top combinations*. We track the top combinations with M servers and record the minimum value that each station takes (s_i^{min}) in this set. The s_i^{min} values for top combinations with $M+1$ servers are greater than or equal to s_i^{min} for top combinations with M servers. If a station in the configuration with $M+1$ servers were to have a lower s_i^{min} value compared to the configuration with M servers, this station becomes a bottleneck as its total service rate would be less than other stations, which in turn could reduce the throughput. This is consistent with Algorithm 1 and reinforced by simulations. Note that we do not claim that the optimal allocation for $M+1$ servers has s_i values that are greater

than or equal to those in the optimal allocation for M servers. Our claim is weaker as it targets a range of s_i values and combinations rather than a specific s_i value.

We used this property to limit our search space when simulating $M + 1$ servers using simulation results for M servers. For example, if a station is assigned 9, 10, or 11 servers within the top combinations with 80 servers, for simulating 81 servers we only consider a range around these numbers (say 7 to 13). In practice, the s_i values in top configurations tend to be equal or differ by at most two servers, which supports our choice of the numbers of servers (s_i) to consider.

In order to determine the order in which servers are allocated during the simulation, we used the following criterion: whenever all s_i values for a station are greater than or equal to a certain number (x) in the top combinations, we let that station have x servers. We recorded the order in which stations satisfied the above criterion.

For example, consider a case with $N = 9$ and $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$, with all values of M between 10 and 98. The simulation results are shown in Table 2.2. Analyzing this table helped us refine our original conjecture in the algorithm. In the table, the header is w, the first column is s_i, and the table entries represent the order in which servers are allocated to stations. To measure the performance of our algorithm, we applied Algorithm 1 to the above configuration. The allocation order generated by the algorithm is shown in Table 2.3.

To gain a better understanding of how the algorithm performs, we summarize the results in Tables 2.4 and 2.6. In these tables, the optimal allocation together with the relative error of the throughput of the allocation generated by the algorithm are shown (for $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$). If the allocation suggested by the algorithm is not optimal, the optimal allocation is also presented below the algorithm's allocation.

s_i	**12**	**7**	**13**	**3**	**5**	**4**	**1**	**10**	**9**
1	11	14	10	24	17	19	52	12	13
2	16	23	15	42	29	33	94	18	20
3	22	32	21	60	39	48		25	27
4	28	40	26	81	54	63		31	35
5	34	50	30		66	77		37	43
6	38	59	36		76	92		44	47
7	45	69	41		90			49	58
8	51	80	46					57	64
9	55	86	53					65	71
10	62	96	56					70	79
11	68		61					74	85
12	73		67					83	93
13	78		72					89	
14	84		75					97	
15	88		82						
16	95		87						
17			91						
18			98						

Table 2.2: Empirical allocation order for $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$

The average relative error of the allocation provided by the algorithm for $M = 10$ to 98 servers is 0.80%. For the same configuration, if we delay increasing the value of l_i for "high priority stations" even for at most 3 visits, i.e. skipping lines 9 to 13 in the algorithm, experimental results are greatly changed and the average relative error increases to 2.00%. This is in fact the result of the greedy algorithm suggested for the infinite buffer setting (and our original conjecture).

s_i	**12**	**7**	**13**	**3**	**5**	**4**	**1**	**10**	**9**
1	11	14	10	26	17	21	48	12	13
2	16	23	15	45	29	35		18	19
3	22	32	20	63	40	49		24	27
4	28	41	25	82	52	62		31	34
5	33	51	30		64	77		37	42
6	38	60	36		76	91		44	47
7	42	69	39		88			53	56
8	50	78	46					58	65
9	55	87	54					66	70
10	59	96	57					71	80
11	67		61					79	85
12	72		68					84	93
13	75		72					90	
14	83		74					97	
15	89		81						
16	94		86						
17			92						
18			95						

Table 2.3: Algorithm 1 allocation order for $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$

We have simulated a number of other configurations to verify the proposed algorithm. We describe two of them here. Assume a nine station line with the following mean service times: $w = (1, 2, 3, 4, 5, 6, 7, 8, 9)$. For $M = 45$ through 70 the average relative error is 0.93%. Consider another nine station configuration with the following mean service times: $w = (5, 4, 2, 9, 8, 3, 8, 7, 1, 6)$. For $M = 45$ through 82 the average

relative error is 0.46% which suggests that the algorithm is performing well.

Comparing our allocations with the optimal allocations for $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$, Algorithm 1 always assigns more servers to stations 1, 3, and 9; it always assigns less servers to stations 4, 5, and 6; station 2 often has the same number of servers except for a few cases where the algorithm assigns less servers. For $w = (5, 4, 2, 9, 8, 3, 8, 7, 1, 6)$, the algorithm always assigns more servers to stations 1, 4, and 9; it always assigns less servers to stations 3, 5, and 8.

Hillier and So (1989) define $n = \lfloor \frac{M}{N} \rfloor$ and $E = M - nN$ for systems with equal workloads. They state that when $N - E = 1$ the optimal allocation assigns extra servers to every station except station 1; when $N - E = 2$ it allocates extra servers to every station except stations 1 and N; when $N - E > 2$ they could not characterize an optimal solution. However, in general terms, they suggest to spread the extra servers rather uniformly over the interior stations. Applying their method to $w = (5, 4, 2, 9, 8, 3, 8, 7, 1, 6)$ when $N - E = 1$, then $M = 53$ and the average relative error is 0.41%. When $N - E = 2$, then $M = 52$ and the average relative error is 0.24%. When $N - E > 2$, then we choose $M = 46$ through 50, resulting in an average relative error of 6.05% (our algorithm results in 1.31% average relative error). Note that their method is silent on allocations in the range of $M = 54$ through 89, as the workloads become equal again at $M = 90$.

Similarly, if we apply their guideline to $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$ when $N - E = 1$, then $M = 72$ and the average relative error is 0.06%. When $N - E = 2$, then $M = 71$ and the average relative error is 0.07%. When $N - E > 2$ we choose $M = 65$ through 69, the average relative error is 2.07%. Their method is silent on allocations in the range of $M = 73$ through 128. Notice that their guideline performs better on

the latter configuration compared to the former configuration, as it is a case where "high priority stations" are also interior stations.

M	Allocation	Throughput	Rel Error
10	(1, 1, 2, 1, 1, 1, 1, 1, 1)	0.0562	
11	(2, 1, 2, 1, 1, 1, 1, 1, 1)	0.0630	
12	(2, 1, 2, 1, 1, 1, 1, 2, 1)	0.0784	
13	(2, 1, 2, 1, 1, 1, 1, 2, 2)	0.0866	
14	(2, 2, 2, 1, 1, 1, 1, 2, 2)	0.0984	
15	(2, 2, 3, 1, 1, 1, 1, 2, 2)	0.1085	
16	(3, 2, 3, 1, 1, 1, 1, 2, 2)	0.1133	6.94%
	(2, 2, 3, 1, 2, 1, 1, 2, 2)	0.1218	
17	(3, 2, 3, 1, 2, 1, 1, 2, 2)	0.1305	
18	(3, 2, 3, 1, 2, 1, 1, 3, 2)	0.1426	
19	(3, 2, 3, 1, 2, 1, 1, 3, 3)	0.1494	2.86%
	(3, 2, 3, 1, 2, 2, 1, 3, 2)	0.1538	
20	(3, 2, 4, 1, 2, 1, 1, 3, 3)	0.1593	3.1%
	(3, 2, 4, 1, 2, 2, 1, 3, 2)	0.1644	
21	(3, 2, 4, 1, 2, 2, 1, 3, 3)	0.1758	
22	(4, 2, 4, 1, 2, 2, 1, 3, 3)	0.1860	1.03%
	(3, 3, 4, 1, 2, 2, 1, 3, 3)	0.1880	
23	(4, 3, 4, 1, 2, 2, 1, 3, 3)	0.1971	1.39%
	(3, 3, 4, 2, 2, 2, 1, 3, 3)	0.1999	
24	(4, 3, 4, 1, 2, 2, 1, 4, 3)	0.2070	2.62%
	(4, 3, 4, 2, 2, 2, 1, 3, 3)	0.2126	
25	(4, 3, 5, 1, 2, 2, 1, 4, 3)	0.2181	3.97%
	(4, 3, 4, 2, 2, 2, 1, 4, 3)	0.2271	
26	(4, 3, 5, 2, 2, 2, 1, 4, 3)	0.2387	
27	(4, 3, 5, 2, 2, 2, 1, 4, 4)	0.2484	0.71%
	(4, 3, 5, 2, 3, 2, 1, 4, 3)	0.2502	
28	(5, 3, 5, 2, 2, 2, 1, 4, 4)	0.2581	1.22%
	(4, 3, 5, 2, 3, 2, 1, 4, 4)	0.2613	
29	(5, 3, 5, 2, 3, 2, 1, 4, 4)	0.2754	
30	(5, 3, 6, 2, 3, 2, 1, 4, 4)	0.2858	

Table 2.4: Algorithm 1 allocation for $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$ - Part I

M	Allocation	Throughput	Rel Error
31	(5, 3, 6, 2, 3, 2, 1, 5, 4)	0.2986	
32	(5, 4, 6, 2, 3, 2, 1, 5, 4)	0.3108	
33	(6, 4, 6, 2, 3, 2, 1, 5, 4)	0.3172	2.8%
	(5, 4, 6, 2, 3, 3, 1, 5, 4)	0.3264	
34	(6, 4, 6, 2, 3, 2, 1, 5, 5)	0.3262	3.04%
	(6, 4, 6, 2, 3, 3, 1, 5, 4)	0.3364	
35	(6, 4, 6, 2, 3, 3, 1, 5, 5)	0.3482	
36	(6, 4, 7, 2, 3, 3, 1, 5, 5)	0.3605	
37	(6, 4, 7, 2, 3, 3, 1, 6, 5)	0.3727	0.17%
	(6, 4, 7, 2, 4, 3, 1, 5, 5)	0.3733	
38	(7, 4, 7, 2, 3, 3, 1, 6, 5)	0.3803	1.44%
	(6, 4, 7, 2, 4, 3, 1, 6, 5)	0.3859	
39	(7, 4, 8, 2, 3, 3, 1, 6, 5)	0.3888	2.25%
	(6, 5, 7, 2, 4, 3, 1, 6, 5)	0.3978	
40	(7, 4, 8, 2, 4, 3, 1, 6, 5)	0.4074	0.62%
	(6, 5, 7, 3, 4, 3, 1, 6, 5)	0.4099	
41	(7, 5, 8, 2, 4, 3, 1, 6, 5)	0.4176	0.96%
	(7, 5, 7, 3, 4, 3, 1, 6, 5)	0.4217	
42	(7, 5, 8, 2, 4, 3, 1, 6, 6)	0.4297	1.33%
	(7, 5, 7, 3, 4, 3, 2, 6, 5)	0.4355	
43	(8, 5, 8, 2, 4, 3, 1, 6, 6)	0.4347	2.88%
	(7, 5, 8, 3, 4, 3, 2, 6, 5)	0.4476	
44	(8, 5, 8, 2, 4, 3, 1, 7, 6)	0.4470	3.66%
	(7, 5, 8, 3, 4, 3, 2, 6, 6)	0.4639	
45	(8, 5, 8, 3, 4, 3, 1, 7, 6)	0.4706	1.01%
	(7, 5, 8, 3, 4, 3, 2, 7, 6)	0.4754	
46	(8, 5, 9, 3, 4, 3, 1, 7, 6)	0.4809	1.53%
	(8, 5, 8, 3, 4, 3, 2, 7, 6)	0.4884	
47	(8, 5, 9, 3, 4, 3, 1, 7, 7)	0.4886	2.64%
	(8, 5, 9, 3, 4, 3, 2, 7, 6)	0.5018	

Table 2.5: Algorithm 1 allocation for $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$ - Part II

M	Allocation	Throughput	Rel Error
48	(8, 5, 9, 3, 4, 3, 2, 7, 7)	0.5103	1.16%
	(8, 5, 9, 3, 4, 4, 2, 7, 6)	0.5163	
49	(8, 5, 9, 3, 4, 4, 2, 7, 7)	0.5257	0.37%
	(8, 6, 9, 3, 4, 4, 2, 7, 6)	0.5276	
50	(9, 5, 9, 3, 4, 4, 2, 7, 7)	0.5358	0.86%
	(8, 6, 9, 3, 4, 4, 2, 8, 6)	0.5404	
51	(9, 6, 9, 3, 4, 4, 2, 7, 7)	0.5489	0.74%
	(8, 6, 9, 3, 5, 4, 2, 7, 7)	0.5531	
52	(9, 6, 9, 3, 5, 4, 2, 7, 7)	0.5642	0.01%
	(8, 6, 10, 3, 4, 4, 2, 8, 7)	0.5643	
53	(9, 6, 9, 3, 5, 4, 2, 8, 7)	0.5790	
54	(9, 6, 10, 3, 5, 4, 2, 8, 7)	0.5957	
55	(10, 6, 10, 3, 5, 4, 2, 8, 7)	0.6037	0.57%
	(9, 6, 10, 4, 5, 4, 2, 8, 7)	0.6071	
56	(10, 6, 10, 3, 5, 4, 2, 8, 8)	0.6137	0.49%
	(9, 6, 10, 4, 5, 4, 2, 9, 7)	0.6186	
57	(10, 6, 11, 3, 5, 4, 2, 8, 8)	0.6271	0.42%
	(9, 7, 10, 4, 5, 4, 2, 9, 7)	0.6297	
58	(10, 6, 11, 3, 5, 4, 2, 9, 8)	0.6403	0.19%
	(10, 6, 11, 4, 5, 4, 2, 8, 8)	0.6415	
59	(11, 6, 11, 3, 5, 4, 2, 9, 8)	0.6479	1.38%
	(10, 6, 11, 4, 5, 4, 2, 9, 8)	0.6570	
60	(11, 7, 11, 3, 5, 4, 2, 9, 8)	0.6594	1.88%
	(10, 7, 11, 4, 5, 4, 2, 9, 8)	0.6721	
61	(11, 7, 12, 3, 5, 4, 2, 9, 8)	0.6695	2.38%
	(10, 7, 11, 4, 5, 5, 2, 9, 8)	0.6858	
62	(11, 7, 12, 3, 5, 5, 2, 9, 8)	0.6867	1.53%
	(10, 7, 11, 4, 6, 5, 2, 9, 8)	0.6974	
63	(11, 7, 12, 4, 5, 5, 2, 9, 8)	0.7078	0.19%
	(11, 7, 11, 4, 6, 5, 2, 9, 8)	0.7091	

Table 2.6: Algorithm 1 allocation for $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$ - Part III

M	Allocation	Throughput	Rel Error
64	(11, 7, 12, 4, 6, 5, 2, 9, 8)	0.7216	0.08%
	(11, 7, 12, 4, 5, 5, 2, 10, 8)	0.7222	
65	(11, 7, 12, 4, 6, 5, 2, 9, 9)	0.7367	0.04%
	(11, 7, 12, 4, 6, 5, 2, 10, 8)	0.7370	
66	(11, 7, 12, 4, 6, 5, 2, 10, 9)	0.7508	
67	(12, 7, 12, 4, 6, 5, 2, 10, 9)	0.7611	0.33%
	(11, 8, 12, 4, 6, 5, 2, 10, 9)	0.7636	
68	(12, 7, 13, 4, 6, 5, 2, 10, 9)	0.7760	0.01%
	(11, 8, 13, 4, 6, 5, 2, 10, 9)	0.7761	
69	(12, 8, 13, 4, 6, 5, 2, 10, 9)	0.7876	0.18%
	(11, 8, 13, 4, 6, 5, 2, 11, 9)	0.7891	
70	(12, 8, 13, 4, 6, 5, 2, 10, 10)	0.7995	0.43%
	(12, 8, 13, 4, 6, 5, 2, 11, 9)	0.8029	
71	(12, 8, 13, 4, 6, 5, 2, 11, 10)	0.8134	0.12%
	(12, 8, 13, 4, 7, 5, 2, 11, 9)	0.8144	
72	(13, 8, 13, 4, 6, 5, 2, 11, 10)	0.8223	0.56%
	(12, 8, 13, 5, 6, 5, 2, 11, 10)	0.8270	
73	(13, 8, 14, 4, 6, 5, 2, 11, 10)	0.8360	0.52%
	(12, 8, 14, 4, 7, 5, 2, 11, 10)	0.8404	
74	(13, 8, 15, 4, 6, 5, 2, 11, 10)	0.8451	0.91%
	(12, 8, 14, 5, 6, 6, 2, 11, 10)	0.8529	
75	(14, 8, 15, 4, 6, 5, 2, 11, 10)	0.8519	1.72%
	(13, 8, 14, 4, 7, 6, 2, 11, 10)	0.8661	
76	(14, 8, 15, 4, 7, 5, 2, 11, 10)	0.8679	1.28%
	(13, 8, 14, 5, 7, 6, 2, 11, 10)	0.8791	
77	(14, 8, 15, 4, 7, 6, 2, 11, 10)	0.8847	0.79%
	(13, 8, 14, 5, 7, 6, 2, 12, 10)	0.8917	
78	(14, 9, 15, 4, 7, 6, 2, 11, 10)	0.8943	1.35%
	(13, 9, 14, 5, 7, 6, 2, 12, 10)	0.9066	
79	(14, 9, 15, 4, 7, 6, 2, 12, 10)	0.9139	0.49%
	(13, 9, 15, 5, 7, 6, 2, 12, 10)	0.9185	

Table 2.7: Algorithm 1 allocation for $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$ - Part IV

M	Allocation	Throughput	Rel Error
80	(14, 9, 15, 4, 7, 6, 2, 12, 11)	0.9286	0.46%
	(13, 9, 15, 5, 7, 6, 2, 12, 11)	0.9329	
81	(14, 9, 16, 4, 7, 6, 2, 12, 11)	0.9397	0.68%
	(14, 9, 15, 5, 7, 6, 2, 12, 11)	0.9462	
82	(14, 9, 16, 5, 7, 6, 2, 12, 11)	0.9571	0.24%
	(14, 9, 15, 5, 7, 6, 3, 12, 11)	0.9595	
83	(15, 9, 16, 5, 7, 6, 2, 12, 11)	0.9661	0.61%
	(14, 9, 16, 5, 7, 6, 2, 13, 11)	0.9721	
84	(15, 9, 16, 5, 7, 6, 2, 13, 11)	0.9835	0.16%
	(14, 9, 16, 5, 7, 6, 3, 13, 11)	0.9851	
85	(15, 9, 16, 5, 7, 6, 3, 13, 11)	0.9985	
86	(15, 9, 16, 5, 7, 6, 3, 13, 12)	1.0085	0.13%
	(15, 9, 16, 5, 8, 6, 3, 13, 11)	1.0098	
87	(15, 9, 17, 5, 7, 6, 3, 13, 12)	1.0213	0.06%
	(15, 10, 16, 5, 8, 6, 3, 13, 11)	1.0219	
88	(15, 10, 17, 5, 7, 6, 3, 13, 12)	1.0352	0.21%
	(15, 10, 16, 5, 8, 6, 3, 13, 12)	1.0374	
89	(15, 10, 17, 5, 8, 6, 3, 13, 12)	1.0521	
90	(15, 10, 17, 5, 8, 7, 3, 13, 12)	1.0652	
91	(16, 10, 17, 5, 8, 7, 3, 13, 12)	1.0783	
92	(16, 10, 17, 5, 8, 7, 3, 14, 12)	1.0913	
93	(16, 10, 18, 5, 8, 7, 3, 14, 12)	1.1060	
94	(16, 10, 18, 5, 8, 7, 3, 14, 13)	1.1174	
95	(17, 10, 18, 5, 8, 7, 3, 14, 13)	1.1298	0.09%
	(16, 11, 18, 5, 8, 7, 3, 14, 13)	1.1309	
96	(17, 10, 19, 5, 8, 7, 3, 14, 13)	1.1413	0.32%
	(16, 11, 18, 6, 8, 7, 3, 14, 13)	1.1450	

Table 2.8: Algorithm 1 allocation for $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$ - Part V

2.4 Impact of service time variance

A natural extension to the above problem is considering service time distributions which are not exponential. More specifically, we modify the distributions so that the coefficients of variation (cv) are not equal to 1. In simulations, an Erlang distribution is used for coefficient of variation less than one and a Hyper-exponential distribution for coefficient of variation greater than one. An Erlang distribution is the distribution of the sum of k independent exponential variables with mean μ and probability density function $f(x;k,\lambda) = \frac{\lambda^x x^{k-1} e^{-\lambda x}}{(k-1)!}$ for $x, k \geq 0$. A Hyper-exponential distribution is the distribution of a mixure of k exponential distributions such that the probability density function of a random variable X is given by $f_X(x) = \sum_{i=1}^{k} f_{Y_i}(x) p_i$ where Y_i is an exponentially distributed random variable with rate parameter λ_i and p_i is the probability that X will take on the form of the exponential distribution with rate λ_i. In such systems, allocations depend on the position of stations in the tandem line, service rates, and coefficients of variation. We try to study the effect of these parameters separately and together. We let cv_j represent the coefficient of variation of the service time distribution of station j.

For the following cases, Algorithm 1 works well: coefficients of variation are approximately one except for a small number of stations which are less than one; coefficients of variation are less than one with equal values; coefficients of variation are greater than one with equal values. For the remaining cases: coefficients of variation are approximately one except for a small number of stations where they are greater than one; coefficients of variation less than one with different values; coefficients of variation greater than one with different values, we provide some guidelines for how to modify Algorithm 1, but are unable to provide a complete picture.

In a tandem line including stations with $cv_j = 1$ and $cv_j > 1$, the guideline is to follow Algorithm 1 but expedite visits to stations with $cv_j > 1$. Employing Algorithm 1 alone with no modifications could lead to results that are far from optimal. The amount that visit periods should be advanced depends on the coefficients of variation, the p_j values, and the position of stations. Notice, however, that visit periods are still mainly dependent on service rates. Therefore the change in visit periods is not very large compared to p_j. During our simulations, we have observed that a 100% increase in the coefficient of variation of a station changes its visit period by not more than 20%.

For the $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$ configuration with $cv_5 = 1.61, cv_8 = 1.31$ and $cv_j = 1$ for the remaining stations, the average relative error of Algorithm 1 is 1.61%. As stated, p_5 and p_8 should be decreased (we do not know the exact amount). However, we only need to perform simulations for a couple of M values to calculate new visit periods. Choosing $M = 13, 14, 19$, we inferred that we should change p_5 from 12.8 to 10.75 and p_8 from 6.4 to 6.25. Employing Algorithm 1 with the modified visit periods results in an average relative error of 0.94%. Table 2.9 presents the optimal allocations for this example for $M = 43$ through 69.

In a tandem line where for all stations $cv_j > 1$ with different values, visits to stations with higher mean service times are expedited which can result in an increase in visit periods of lower mean service time stations. Consider the $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$ configuration with $cv_j > 1$ for all stations. More specifically, let $cv = (1.22, 1.76, 1.14, 2.68, 2.12, 2.37, 2.45, 1.38, 1.50)$. Algorithm 1 with the original visit periods results in an average relative error of 5.53%. We performed simulations for a small number of M values ($M = 15, 17, 20$) and determined the

following values for visit periods: $(6.11, 8, 5.4, 15.66,\ 10.8, 13, 35, 7, 7.28)$. With these adjusted visit periods the average relative error becomes 0.67%. Comparing optimal allocations with the allocations computed from Algorithm 1 adjusted with the above visit periods, the optimal allocations assign more servers to stations with higher mean service times.

In a tandem line where for all stations $cv_j > 1$ with equal values, Algorithm 1 provides good results (i.e. low average relative error). For example for $w = (3, 2, 1, 2, 1, 1, 1, 2)$ and $cv_j = 2.15$ for all stations, for $M = 9$ through 50 Algorithm 1 leads to an average relative error of 0.93%.

In a tandem line including stations with $cv_j = 1$ and $cv_j < 1$, Algorithm 1 leads to reasonable results. However, the optimal allocation tends to postpone visits to stations with $cv_j < 1$ in comparison with p_j calculated in Algorithm 1. In other words, the optimal policy assigns less servers to stations with lower coefficients of variation. For the $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$ configuration with $M = 10$ through 69, $cv_5 = cv_8 = 0.5$, and $cv_j = 1$ for the rest, the average relative error of the allocations provided by Algorithm 1 is 0.79%. Table 2.9 presents the optimal allocations for this example for $M = 43$ through 69.

In a tandem line where for all stations $cv_j < 1$ with equal values, Algorithm 1 provides good results. The optimal allocation assigns more servers to stations with higher mean service times and to stations at the two ends of the tandem line. For the same configurations with $M = 10$ through 69 and $cv_j = 0.5$ for all stations, Algorithm 1 results in an average relative error of 0.52%.

In a tandem line where for all stations $cv_j < 1$ with different values, less servers are assigned to stations with higher coefficient of variation. For $w = (3, 2, 1, 2, 1, 1, 1, 2)$

and $cv = (0.7, 0.7, 0.25, 0.25, 0.7, 0.25, 0.25, 0.25)$ with $M = 9$ through 50, Algorithm 1 results in an average relative error of 1.95%. The average relative error for the same configuration with $cv_j = 1$ is 0.58%. This suggests that the visit periods in Algorithm 1 should be adjusted. Simulating for a small number of M values ($M = 19, 25, 26, 27$), we inferred that the adjusted visit periods should be $(4.75, 6.33, 12, 6.16, 12, 12.5, 12, 6.9)$. Employing Algorithm 1 with the adjusted visit periods results in an average relative error of 0.58%.

In order to study the effect of coefficient of variation independent of mean service times, we consider a configuration with 4 stations, each having a mean service time of 1. We choose $cv_j = 0.5, 1, 1.34$ for the third station and let $cv_j = 1$ for the other stations. As expected, the inverted bowl phenomenon is followed, i.e. the second and third stations are prioritized. While visit periods remain the same when the coefficient of variation is changed, their ordering changes. For $cv_j = 1$, the allocation priority swings between the second and third station so that balance is achieved in the long run. For $cv_j \neq 1$, although allocation priority swings between the second and third station, for $cv_j < 1$ the second station is prioritized and for $cv_j > 1$ the third station is prioritized. This observation is consistent with the above claims.

Futamura (2000) states that the optimal server allocation follows the inverted bowl phenomenon but assigns more servers to stations with higher coefficient of variation. As the number of stations is increased, the optimal allocation tends to put servers at stations with higher coefficient of variation over stations in the middle. The systems under consideration have equal workload and all coefficients of variation are 1 except some stations which have higher coefficients of variation. We observe that a more comprehensive analysis of such systems would suggest that increasing the coefficient

of variation of some stations changes the allocation order but leaves visit periods unchanged. Table 2.9 illustrates this fact. In Table 2.9, while the fifth station with $cv_5 > 1$ takes servers 5 and 6 sooner compared to the case $cv_5 = 1$, the visit periods for both cases are the same. This is consistent with our earlier observation that changing the coefficient of variation has an effect on the server assignment, but not to the degree that there is a large difference in the number of servers assigned (over the exponential case).

In the next chapter, we let a number of flexible servers be shared among stations. We provide allocation policies that maximize the throughput when both dedicated and flexible servers are present. We also discuss how the throughput is improved, when flexible servers are used.

M	$cv_j = 1$	$cv_5 = cv_8 = 0.5$	$cv_5 = 1.61, cv_8 = 1.31$
43	(7, 5, 8, 3, 4, 3, 2, 6, 5)	(7, 5, 8, 3, 4, 3, 1, 6, 6)	(7, 5, 8, 3, 4, 3, 1, 7, 5)
44	(7, 5, 8, 3, 4, 3, 2, 6, 6)	(7, 5, 8, 3, 4, 3, 2, 6, 6)	(7, 5, 8, 3, 4, 3, 1, 7, 6)
45	(7, 5, 8, 3, 4, 3, 2, 7, 6)	(8, 5, 8, 3, 4, 3, 2, 6, 6)	(7, 5, 8, 3, 5, 3, 1, 7, 6)
46	(8, 5, 8, 3, 4, 3, 2, 7, 6)	(8, 5, 8, 3, 4, 3, 2, 7, 6)	(8, 5, 8, 3, 5, 3, 1, 7, 6)
47	(8, 5, 9, 3, 4, 3, 2, 7, 6)	(8, 5, 9, 3, 4, 3, 2, 7, 6)	(8, 5, 8, 3, 5, 3, 2, 7, 6)
48	(8, 5, 9, 3, 4, 4, 2, 7, 6)	(8, 5, 9, 3, 4, 4, 2, 7, 6)	(8, 5, 9, 3, 5, 3, 2, 7, 6)
49	(8, 6, 9, 3, 4, 4, 2, 7, 6)	(8, 6, 9, 3, 4, 4, 2, 7, 6)	(8, 5, 9, 3, 5, 3, 2, 8, 6)
50	(8, 6, 9, 3, 4, 4, 2, 8, 6)	(8, 6, 9, 3, 4, 4, 2, 7, 7)	(8, 6, 9, 3, 5, 3, 2, 8, 6)
51	(8, 6, 9, 3, 5, 4, 2, 7, 7)	(8, 6, 10, 3, 4, 4, 2, 7, 7)	(8, 6, 9, 3, 5, 4, 2, 8, 6)
52	(8, 6, 10, 3, 4, 4, 2, 8, 7)	(9, 6, 10, 3, 4, 4, 2, 7, 7)	(9, 6, 9, 3, 5, 4, 2, 8, 6)
53	(9, 6, 9, 3, 5, 4, 2, 8, 7)	(9, 6, 10, 3, 4, 4, 2, 8, 7)	(9, 6, 9, 3, 5, 4, 2, 8, 7)
54	(9, 6, 10, 3, 5, 4, 2, 8, 7)	(9, 6, 10, 3, 5, 4, 2, 8, 7)	(9, 6, 10, 3, 5, 4, 2, 8, 7)
55	(9, 6, 10, 4, 5, 4, 2, 8, 7)	(9, 6, 11, 3, 5, 4, 2, 8, 7)	(9, 6, 10, 3, 5, 4, 2, 9, 7)
56	(9, 6, 10, 4, 5, 4, 2, 9, 7)	(9, 6, 11, 3, 5, 4, 2, 9, 7)	(9, 6, 10, 3, 6, 4, 2, 9, 7)
57	(9, 7, 10, 4, 5, 4, 2, 9, 7)	(10, 6, 11, 3, 5, 4, 2, 9, 7)	(9, 6, 11, 3, 6, 4, 2, 9, 7)
58	(10, 6, 11, 4, 5, 4, 2, 8, 8)	(10, 6, 11, 4, 5, 4, 2, 9, 7)	(10, 6, 11, 3, 6, 4, 2, 9, 7)
59	(10, 6, 11, 4, 5, 4, 2, 9, 8)	(10, 6, 11, 4, 5, 4, 2, 9, 8)	(10, 6, 11, 3, 6, 4, 2, 9, 8)
60	(10, 7, 11, 4, 5, 4, 2, 9, 8)	(10, 7, 11, 4, 5, 4, 2, 9, 8)	(10, 6, 11, 4, 6, 4, 2, 9, 8)
61	(10, 7, 11, 4, 5, 5, 2, 9, 8)	(10, 7, 11, 4, 5, 5, 2, 9, 8)	(10, 7, 11, 4, 6, 4, 2, 9, 8)
62	(10, 7, 11, 4, 6, 5, 2, 9, 8)	(10, 7, 12, 4, 5, 5, 2, 9, 8)	(10, 7, 11, 4, 6, 4, 2, 10, 8)
63	(11, 7, 11, 4, 6, 5, 2, 9, 8)	(10, 7, 12, 4, 5, 5, 2, 10, 8)	(10, 7, 12, 4, 6, 4, 2, 10, 8)
64	(11, 7, 12, 4, 5, 5, 2, 10, 8)	(11, 7, 12, 4, 5, 5, 2, 10, 8)	(10, 7, 12, 4, 6, 5, 2, 10, 8)
65	(11, 7, 12, 4, 6, 5, 2, 10, 8)	(11, 7, 12, 4, 5, 5, 2, 10, 9)	(11, 7, 12, 4, 6, 5, 2, 10, 8)
66	(11, 7, 12, 4, 6, 5, 2, 10, 9)	(11, 7, 13, 4, 5, 5, 2, 10, 9)	(11, 7, 12, 4, 6, 5, 2, 10, 9)
67	(11, 8, 12, 4, 6, 5, 2, 10, 9)	(11, 8, 12, 4, 6, 5, 2, 10, 9)	(11, 7, 13, 4, 6, 5, 2, 10, 9)
68	(11, 8, 13, 4, 6, 5, 2, 10, 9)	(11, 8, 13, 4, 6, 5, 2, 10, 9)	(11, 8, 13, 4, 6, 5, 2, 10, 9)
69	(11, 8, 13, 4, 6, 5, 2, 11, 9)	(11, 8, 13, 4, 6, 5, 2, 11, 9)	(11, 8, 13, 4, 6, 5, 2, 11, 9)

Table 2.9: Comparison of coefficient of variation modifications for $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$

Chapter 3

Dynamic server allocation for zero-buffer tandem queues with dedicated and flexible servers

3.1 Introduction

Consider the model introduced in Section 1.1. Let $M \geq 1$ dedicated servers and $F \geq 1$ flexible servers be present in the system. Assume that the dedicated servers are already assigned to the stations. We are interested in *determining the dynamic assignment policy for flexible servers that maximizes the long-run average throughput.* Long-run average throughput is defined as $\lim_{t\to\infty} \frac{\text{the number of departures by time } t}{t}$. For simplicity, we assume the travel times for flexible servers to move between stations are negligible.

We introduce the concept of "hand-off" for flexible servers. Hand-off happens when a flexible server passes the job it is serving to a dedicated server at the same

station. Although it is possible to perform hand-off at any time, we let it occur only in the following two cases. When a station has a busy flexible server and a free dedicated server, the flexible server can pass its job to the dedicated server and become free. When a station has a busy flexible server and a blocked dedicated server, jobs can be swapped between the two servers. In either of the two cases, we say a hand-off has taken place.

Any allocation policy should define appropriate actions when blocking or starvation occurs. In cases where there are multiple blocked or starved servers, policies should prioritize resolving blocking or starvation of the involved servers. We will show that when workloads (i.e. the ratio of the mean service time to the number of dedicated servers for a station) are equal, a near-optimal policy is one that clears blocking from the end of the line to the beginning and avoids starving servers. With different workloads, a near-optimal policy prioritizes clearing blocking for stations with higher mean service times.

An instance of the application of the problem described above is the bed management problem from the healthcare domain. Bed management deals with optimizing patient flow into and out of beds, so that waiting times are reduced (e.g., see (Department of Health and Aged Care, 1999)). A patient arriving at a hospital might need to go through a sequence of units. For example, the patient might go through, in order, the emergency, express, medicine, surgery, recovery, and complex units. The role of the express unit is to accommodate the patient until a bed becomes available in the medicine unit. A number of beds, called express beds, are shared between the emergency, express, and medicine units to move patients between these units. However, sharing is performed in an ad-hoc manner and is limited to these three units only.

Several questions arise, including: how much is the throughput improvement, if beds are shared among the units? How should the shared beds be allocated dynamically to the units? If we need to take a number of dedicated beds from the units and share them, which units should be chosen? While we do not model this application directly, our results are a first step in an analytic approach to such problems.

There is a rich literature on the assignment of flexible servers in tandem queues. Most of the papers in this domain focus on minimizing holding costs. Iravani (1997) considers tandem queues attended by a moving server with holding and switching costs. As a basic model, he shows that for two-stage tandem queues, the policy which minimizes the total discounted and long run average holding and switching costs is a greedy and exhaustive policy in the second stage. The first stage could then follow static, gated-limited, or double-threshold policies.

Ahn et al. (2002) consider the optimal control of two parallel servers in a two-stage tandem queueing system with two flexible servers and holding costs. They examine both collaborative and non-collaborative cases. In the collaborative case, servers may collaborate to work on the same job at the same time, while in the non-collaborative case each job can be served by at most one server at a time. They provide simple conditions under which it is optimal to allocate both servers to station 1 or 2 in the collaborative case. In the non-collaborative case, they show that the same condition as in the collaborative case guarantees the existence of an optimal policy that is exhaustive in station 1. However the condition for exhaustive service at station 2 to be optimal does not carry over.

Pandelis (2008) considers a two-station tandem line with both dedicated and flexible servers where all servers have time-varying rates. Servers can work collaboratively.

With a given probability, jobs can leave the system after completion in the first station. The optimal policy to minimize holding costs is described.

For a two-class queueing system with one dedicated server, one flexible server, and no exogenous arrivals, Ahn et al. (2004) characterize the server assignment policy that minimizes the expected total holding cost incurred until all jobs initially present in the system have departed.

Andradóttir et al. (2007) consider tandem queues having both flexible and dedicated servers with finite or infinite buffers between stations. They study the dynamic assignment of servers, in the collaborative case, such that the long-run average throughput is maximized. They assume that the service requirements of jobs at stations are exponentially distributed. For 2 stations and 3 servers (both flexible and dedicated) the allocation of the flexible servers is of threshold type. Multiple thresholds are used to express allocation policies and the thresholds are specified.

Wu et al. (2006) consider the notion of dedicated and flexible servers to determine the allocation of flexible servers that minimizes holding cost in a clearing system with two queues in tandem and in which dedicated servers are subject to failure. Wu et al. (2008) extend the work by Wu et al. (2006) to more general serial lines with external arrivals under discounted and average cost criteria.

Andradóttir et al. (2001, 2003), consider the dynamic assignment of servers to maximize the long-run average throughput in queueing networks with flexible servers. They consider tandem queues with two stations and flexible servers with a finite buffer between the stations. Servers work collaboratively. They use Policy Iteration to show that with less than three servers, a policy which avoids blocking at the first station and starvation at the second station, is optimal. Andradóttir and Ayhan (2005) study

the same system with three servers. Assuming server 1(2) is more efficient than server 2(3) for serving jobs at the second station, they prove that the optimal policy has three mandates: (i) server 1 works at station 1, unless the station is blocked; (ii) server 3 works at station 2, unless it is starved; (iii) server 2 is a roving server, it works at station 1 if the buffer is low and works at station 2 when the buffer level is high.

Hasenbein and Kim (2011) also consider the system introduced in the last paragraph. They prove that the conjecture proposed by Andradóttir and Ayhan (2005) for generic numbers of servers, is indeed true. They use general properties of the bias of the optimal policy to show that a threshold policy on buffer levels exists that relies on ordering of server efficiency at stations. They further determine the threshold values. They also prove that restricting the buffer size decreases the maximum achievable throughput.

Kırkızlar et al. (2012) analyze a tandem line which is understaffed, i.e. there are more stations than servers. They consider tandem queues with three stations, two servers, different flexibility structures, and either deterministic service times and arbitrary buffers or exponential service times and small buffers. In the deterministic setting, they prove that the best possible production rate with full server flexibility and infinite buffers can be attained with partial flexibility and zero buffers.

In contrast to (Ahn et al., 2002, 2004; Iravani, 1997; Pandelis, 2008) which consider holding costs, we study throughput in our work. The problem considered by Andradóttir and Ayhan (2005) differs from our work as it studies the collaborative case and includes buffers between stations. The work by Andradóttir et al. (2001, 2007); Hasenbein and Kim (2011) encompasses buffer spaces, collaborative servers,

assumes heterogeneous service rates for servers, and allows a single server per station. Note that in a zero-buffer setting, server allocation in tandem lines with collaborative servers is a different problem than server allocation with non-collaborative servers. In the work by Andradóttir et al. (2003), jobs in the system belong to different classes and service rates are heterogeneous based on server and job class. In this chapter, we determine optimal policies for dynamic allocation of flexible servers in zero-buffer tandem lines with heterogeneous service rates for stations. We study the non-collaborative case and our goal is to maximize throughput. In addition, we report the throughput improvement gained from making dedicated servers flexible.

This chapter is organized as follows. In Section 3.2, the optimality of hand-off and non-idling for tandem lines with two stations is shown. In Section 3.3 we use Markov Decision Process theory to derive the optimal policy for tandem lines with two stations, an arbitrary number of dedicated servers, and one flexible server. We further show how to employ the Policy Iteration algorithm to construct the optimal policy. In Section 3.4 we apply Policy Iteration to larger instances (tandem lines with 3, 4, and 5 stations) and describe a more generic near-optimal policy. Using the insights gained from Section 3.4, in Section 3.5 we provide heuristics for allocation policies for systems of arbitrary size and with heterogeneous mean service times. We also study configurations with service times that are not exponentially distributed. Section 3.6 studies the effects of increasing the number of flexible servers on the throughput.

3.2 Optimal Policy Properties

In this section we introduce and prove two properties of the optimal policy for tandem lines with two stations. By optimal we mean that the policy maximizes throughput, which we equivalently consider that the policy leads to the highest number of departures (at every point in time) from the first station. These properties are the optimality of hand-off and of non-idling of flexible servers. As a result, the state and action spaces when modelling the system with Markov Decision Processes are constrained so that any states or actions which do not perform hand-off or idle flexible servers can be excluded. This fact becomes helpful when search spaces are big, i.e. for systems with a large number of stations or servers. In Theorems 3.1 and 3.2 there is one dedicated server at each station and one flexible server that can work at either station. In Section 3.2.5, Corollary 3.1 shows that Theorems 3.1 and 3.2 hold for systems with arrivals and for clearing systems. Also, Corollary 3.2 shows that the results hold for systems with arbitrary numbers of dedicated and flexible servers.

3.2.1 Hand-off property

In this subsection, we state and prove Theorem 3.1 on the optimality of hand-off.

Theorem 3.1. *The optimal policy performs hand-off whenever possible.*

Proof: There are only two scenarios under which hand-off is possible:

1. A dedicated server becomes blocked in the same station where a flexible server is working (for $N = 2$, this can only happen in the first station). Hand-off is used to clear blocking.

2. A dedicated server becomes available in the same station where a flexible server is working (for $N = 2$, this can only happen in the second station). Hand-off is used to avoid a flexible server working at a station where there is an idle dedicated server.

Let ω and ω' be policies that always perform hand-off, with the only exception that ω' does not perform a hand-off at one of the times when a hand-off is possible.

We define the property $ahead(n)$ as follows. Comparing the system under policies ω and ω' when n jobs have departed from the first station under ω, $ahead(n)$ holds if any of the following hold for each job at a station: i) the same job is being served with identical time spent in the system; ii) the same job is being served with less residual service time under ω; iii) ω is serving a job that ω' has not yet admitted. Note that when there are two jobs in a station, more than one of the above situations (i.e. i), ii), and iii)) can hold. Both stations must satisfy these conditions.

The proof is structured as follows:

For the first scenario, Lemma 3.1 states that if the n^{th} departure happens sooner under ω than ω' and $ahead(n)$ holds, then the $n{+}1^{st}$ departure under ω' will not occur sooner under ω.

For the second scenario, Lemma 3.2 states that if the n^{th} departure happens sooner under ω than ω' and $ahead(n)$ holds, then the $n{+}1^{st}$ departure under ω' will not occur sooner under ω.

For both scenarios, Lemma 3.3 completes the proof by showing that given a system with $m > 1$ departures from the first station that satisfies $ahead(m)$, Lemmas 3.1 or 3.2 can be applied iteratively on the system to satisfy $ahead(m + 1)$.

Given Lemmas 3.1, 3.2, and 3.3, a simple induction argument follows. Define $D^{\omega}(n)$ to be the time at which the n^{th} departure from the first station occurs under ω. We use induction to show $\forall n, m.(D^{\omega}(n) \leq D^{\omega'}(n) \wedge m > n) \Rightarrow (D^{\omega}(m) \leq D^{\omega'}(m))$.

(Theorem 3.1) $\square$

Lemma 3.1. *For the first scenario,* $\forall n \ . \ (D^{\omega}(n) \leq D^{\omega'}(n) \wedge \mathit{ahead}(n)) \Rightarrow (D^{\omega}(n+1) \leq D^{\omega'}(n+1))$.

Lemma 3.2. *For the second scenario,* $\forall n \ . \ (D^{\omega}(n) \leq D^{\omega'}(n) \wedge \mathit{ahead}(n)) \Rightarrow (D^{\omega}(n+1) \leq D^{\omega'}(n+1))$.

Lemma 3.3. $\forall m > 2 \ . \ \mathit{ahead}(m) \Rightarrow \mathit{ahead}(m+1)$.

3.2.2 Proof of Lemmas 3.1, 3.2, and 3.3

In all the lemmas below, the following labelling convention is used. The labelling *Case* $l_1.\ldots.l_{k-1}.l_k$ means that all of the assumptions made in cases l_1 to $l_1.\ldots.l_{k-1}$ hold for this case, in addition to the assumption introduced in l_k. *Case* $l_1.\ldots.l_{k-1}.l_k$ and *Case* $l_1.\ldots.l_{k-1}.l'_k$ where $l_k \neq l'_k$ refer to two different branches of $l_1.\ldots.l_{k-1}$ that differ only in the assumptions introduced in l_k and l'_k.

The following notation is used throughout the following lemmas. Let $X^{p,i}_{n,t}$ be the residual service time of job n at time t served by server X that works in station $i \in \{1,2\}$ under policy $p \in \{\omega, \omega'\}$. X can be replaced by d or f, representing a dedicated or a flexible server, respectively. The residual service time for a job is the time remaining to complete the job's service. Dropping any of the indices of $X^{p,i}_{n,t}$

means that the dropped index can take any value. Also note that the service time of job n is independent of its admission time, the server being dedicated/flexible, and the policy used, i.e. $X_n^i = d_{n,t}^{p,i} = f_{n,t'}^{\omega',i}$. To clarify, $d_{n,t}^{\omega,i}$ is the residual service time of job n at time t which is being served by a dedicated server at station i under policy ω. Finally, to refer to a server working at station i, we use σ_X^i, where $X \in \{d, f\}$ as above.

In all figures in the lemmas below (e.g. Figure 3.1), the top label is the time stamp of the configuration, ovals represent stations, the first row shows ω and the second shows ω'. Within each oval are job numbers, in each station the top job is served by a dedicated server and the bottom job by a flexible server. The notation below the stations shows the residual service time of a job appearing in the same station under both policies when residual service times are different. In some cases, a hand-off is shown by two configurations in a row separated by a vertical line.

In Lemmas 3.1 and 3.2, let t_0 be the hand-off time at which the two policies make a different choice.

Lemma 3.1: *For the first scenario,* $\forall n.(D^{\omega}(n) \leq D^{\omega'}(n) \wedge ahead(n)) \Rightarrow (D^{\omega}(n+1) \leq D^{\omega'}(n+1))$.

Proof: The two primary steps of the proof (basis and inductive steps) are shown in the following.

Basis step:

If time t_0 is not reached, both systems remain the same and there is nothing to prove. Otherwise, start the system from the empty state and let the first departure

from the first station ($D^{\omega}(1) = D^{\omega'}(1)$) occur. If $d^1_{2,t_0} < \min\{f^1_{3,t_0}, d^2_{1,t_0}\}$, the system follows the first scenario. σ^1_d is serving job 2, σ^1_f is serving job 3, and σ^2_d is serving job 1. Up until time t_0, the two policies are the same. When time t_0 is reached, the following cases are possible.

Case 1: Assume $d^2_{1,t_0} > f^1_{3,t_0}$, meaning that the flexible server completes its service before the dedicated server. Let $t_1 = f^1_{3,t_0}$. ω' waits t_1 time units until σ^1_f completes job 3 and sends the server to the second station with $f^{\omega',2}_{3,t_0+t_1}$ as residual service time. ω performs a hand-off at t_0 and sends the flexible server to the second station with a residual service time of $f^{\omega,2}_{2,t_0}$. Therefore at $t_0 + t_1$, $D^{\omega}(2) < D^{\omega'}(2)$.

Case 2: Assume $d^2_{1,t_0} \leq f^1_{3,t_0}$, meaning that the dedicated server completes its service before the flexible server. Let $t_1 = d^2_{1,t_0}$. ω' waits t_1 time units until σ^2_d becomes available. So at time $t_0 + t_1$, the blocked job goes to the second station with $d^{\omega',2}_{2,t_0+t_1}$ as residual service time. ω performs a hand-off at t_0 and sends the flexible server to the second station with a residual service time of $f^{\omega,2}_{2,t_0}$ (here the residual service time is equal to the service time). Therefore at $t_0 + t_1$, $D^{\omega}(2) < D^{\omega'}(2)$.

Inductive step:

If t_0 has occurred in the basis step, Lemma 3.3 shows that for each case considered in the inductive step, the system respects the *ahead* property. Hence ω' will not have departures occurring sooner than ω. If time t_0 is not reached, the two systems remain the same and there will be nothing to prove at this stage.

If time t_0 occurs after the $n-1^{st}$ departure but before the n^{th} departure, consider this configuration: a dedicated blocked server (serving job k) and a flexible busy server (serving job $k+1$) at the first station and a dedicated busy server (serving job

p) at the second station.

Case 1: Assume $d^2_{p,t_0} > f^1_{k+1,t_0}$, meaning that the flexible server completes its service before the dedicated server. Let $t_1 = f^1_{k+1,t_0}$, assume σ^1_d is holding the k^{th} job, and $n-1$ departures from the first station have occurred. ω' waits t_1 time units until σ^1_f completes service and sends the server to the second station with $f^{\omega',2}_{k+1,t_0+t_1}$ as residual service time. (To simplify the argument, we assume that at t_0+t_1 a hand-off occurs and the k^{th} job is sent to the second station instead of the $k+1^{st}$ job and hence σ^2_f should serve a job with service time f^2_{k,t_0+t_1}. This can be done as the jobs are indistinguishable.) ω performs a hand-off at t_0 and sends the flexible server to the second station with a service time of $f^{\omega,2}_{k,t_0}$. Therefore at t_0+t_1, $D^{\omega}(n) < D^{\omega'}(n)$. Considering the system at t_0+t_1, the following cases can happen. Note that σ^1_d is blocked at t_0+t_1 under both policies, holding job $k+1$.

Case 1.1: Assume $f^{\omega,2}_{k,t_0} - t_1 < d^2_{p,t_0+t_1} < f^{\omega',2}_{k,t_0+t_1}$. Then $D^{\omega}(n+1) = t_0+t_1+f^{\omega,2}_{k,t_0}-t_1$ and $D^{\omega'}(n+1) = t_0+t_1+d^{\omega',2}_{p,t_0+t_1}$. Therefore $D^{\omega}(n+1) < D^{\omega'}(n+1)$. Figure 3.1 illustrates this case.

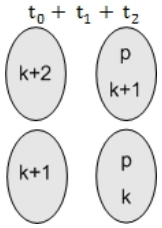

Figure 3.1: Theorem 3.1 - Case 1.1

Case 1.2: Assume $d^2_{p,t_0+t_1} < f^{\omega,2}_{k,t_0} - t_1$. Let $t_2 = d^2_{p,t_0+t_1}$. Therefore $D^{\omega}(n+1) = D^{\omega'}(n+1) = t_0+t_1+t_2$. At $t_0+t_1+t_2$, σ^1_d is serving job $k+2$ and σ^2_d is serving job $k+1$ under both policies. Under ω the residual service time of job k is $f^{\omega,2}_{k,t_0} - t_1 - t_2$ and under ω' it is $f^{\omega',2}_{k,t_0+t_1} - t_2$. Figure 3.2 illustrates this case.

We further divide this case.

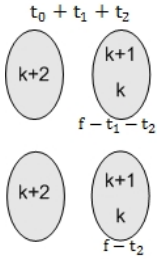

Figure 3.2: Theorem 3.1 - Case 1.2

Case 1.2.1: Assume $d^1_{k+2,t_0+t_1+t_2} < \min\{d^2_{k+1,t_0+t_1+t_2}, f^{\omega,2}_{k,t_0} - t_1 - t_2\}$. We divide this case further.

Case 1.2.1.1: Further assume $f^{\omega,2}_{k,t_0} - t_1 - t_2 < d^2_{k+1,t_0+t_1+t_2}$. Assumption 1.2.1 implies that job $k+2$ is blocked at time $(t_0+t_1+t_2+d^1_{k+2,t_0+t_1+t_2})$. Therefore $D^{\omega}(n+2) < D^{\omega'}(n+2)$.

Case 1.2.1.2: Now assume $d^2_{k+1,t_0+t_1+t_2} < f^{\omega,2}_{k,t_0} - t_1 - t_2$. Let $t_3 = d^2_{k+1,t_0+t_1+t_2}$. Therefore $D^{\omega}(n+2) = D^{\omega'}(n+2) = t_0 + t_1 + t_2 + t_3$. At $t_0 + t_1 + t_2 + t_3$, under both policies, σ^1_d is serving job $k+3$ and σ^2_d is serving job $k+2$. Under ω the residual service time of job k is $f^{\omega,2}_{k,t_0} - t_1 - t_2 - t_3$ and under ω' it is $f^{\omega',2}_{k,t_0+t_1} - t_2 - t_3$. Figure 3.3 illustrates this case. This is similar to Case 1.2. Although this reference seems circular, the fact that job $k+2$ eventually leaves the system avoids this.

Case 1.2.1.3: Now assume $d^2_{k+1,t_0+t_1+t_2} = f^{\omega,2}_{k,t_0} - t_1 - t_2$. This is similar to Case 1.2.2.1, discussed below.

Case 1.2.2: Assume $d^1_{k+2,t_0+t_1+t_2} > \min\{d^2_{k+1,t_0+t_1+t_2}, f^{\omega,2}_{k,t_0} - t_1 - t_2\}$. We divide this case further.

Case 1.2.2.1: Further assume $f^{\omega,2}_{k,t_0} - t_1 - t_2 < d^2_{k+1,t_0+t_1+t_2}$. Let $t_3 =$

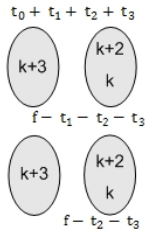

Figure 3.3: Theorem 3.1 - Case 1.2.1.2

$f^{\omega,2}_{k,t_0} - t_1 - t_2$. Consider the system at $t_0 + t_1 + t_2 + t_3$. Under ω we have $d^1_{k+2,t_0+t_1+t_2+t_3}$, $f^{\omega,1}_{k+3,t_0+t_1+t_2+t_3}$, and $d^2_{k+1,t_0+t_1+t_2+t_3}$ as residual service times for jobs $k+2$, $k+3$, and $k+1$, respectively. Under ω' we have $d^1_{k+2,t_0+t_1+t_2+t_3}$, $f^{\omega',2}_{k,t_0+t_1+t_2+t_3}$, and $d^2_{k+1,t_0+t_1+t_2+t_3}$ as residual service times for jobs $k+2$, $k+3$, and $k+1$, respectively. Figure 3.4 illustrates this case. We divide this case further.

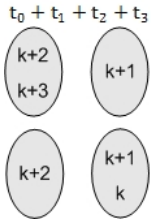

Figure 3.4: Theorem 3.1 - Case 1.2.2.1

Case 1.2.2.1.1: Define $t_4 = f^{\omega',2}_{k,t_0+t_1+t_2+t_3}$ and assume that job k has the smallest residual service time. At $t_0 + t_1 + t_2 + t_3 + t_4$, σ^1_d is serving job $k+2$ and σ^2_d is serving job $k+1$ under both policies. Under ω the residual service time of job $k+3$ is $f^{\omega,2}_{k+3,t_0+t_1+t_2+t_3} - t_4$ and under ω' it is $f^{\omega',2}_{k+3,t_0+t_1+t_2+t_3+t_4}$. This is similar to Case 3.2, discussed below. Figure 3.5 illustrates this case.

Case 1.2.2.1.2: Assume job $k+3$ has the smallest residual service

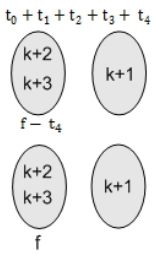

Figure 3.5: Theorem 3.1 - Case 1.2.2.1.1

time. Then $D^{\omega}(n+2) < D^{\omega'}(n+2)$.

Case 1.2.2.1.3: Assume job $k+1$ has the smallest residual service time. This is similar to Case 1.2.2.1.1.

Case 1.2.2.1.4: Assume job $k+2$ has the smallest residual service time. ω performs a hand-off as job $k+2$ is served and therefore $D^{\omega}(n+2) < D^{\omega'}(n+2)$.

Case 1.2.2.1.5: Assume the residual service times of jobs $k+2$ and $k+3$ are equal to or less than the residual service time for job $k+1$. This is similar to Case 1.2.2.1.4.

Case 1.2.2.1.6: Assume the residual service times of jobs $k+1$ and $k+2$ are equal to or less than the residual service time for job $k+3$. This is similar to Case 1.2.2.1.1.

Case 1.2.2.2: Further assume $d^2_{k+1,t_0+t_1+t_2} < f^{\omega,2}_{k,t_0} - t_1 - t_2$. Let $t_3 = d^2_{k+1,t_0+t_1+t_2}$. Consider the system at time $t_0+t_1+t_2+t_3$. Jobs $k+2$ and $k+3$ are being served at the first station under both policies. Under ω the residual service time of job k is $d^{\omega,2}_{k,t_0} - t_1 - t_2 - t_3$. Under ω' the residual service time of job k is $d^{\omega',2}_{k,t_0+t_1} - t_2 - t_3$. Figure 3.6 illustrates this case. We have $D^{\omega}(n+2) = D^{\omega'}(n+2)$.

Case 1.2.2.3: Further assume $d^2_{k+1,t_0+t_1+t_2} = f^{\omega,2}_{k,t_0} - t_1 - t_2$. This is

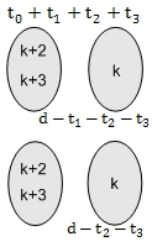

Figure 3.6: Theorem 3.1 - Case 1.2.2.2

similar to Case 1.2.2.2.

Case 1.2.3: Assume $d^1_{k+2,t_0+t_1+t_2} = d^2_{k+1,t_0+t_1+t_2} = f^{\omega,2}_{k,t_0} - t_1 - t_2$. This is similar to Case 1.2.2.1.

Case 1.3: Assume $f^{\omega',2}_{k,t_0+t_1} \leq d^2_{p,t_0+t_1}$. Then $D^{\omega}(n+1) = t_0 + t_1 + f^{\omega,2}_{k,t_0} - t_1$ and $D^{\omega'}(n+1) = t_0 + t_1 + f^{\omega',2}_{k,t_0+t_1}$. Therefore $D^{\omega}(n+1) < D^{\omega'}(n+1)$.

Case 1.4: Assume $d^2_{p,t_0+t_1} = f^{\omega,2}_{k,t_0} - t_1$. This is similar to Case 1.2.2.1.

Case 2: Assume $d^2_{p,t_0} \leq f^1_{k+1,t_0}$, meaning that the dedicated server completes its service before the flexible server. Let $t_1 = d^2_{p,t_0}$, assume σ^1_d is holding the k^{th} job, and $n-1$ departures from the first station have occurred. Under ω', t_1 time units elapse before σ^2_d becomes available. So at $t_0 + t_1$, the blocked job goes to the second station with residual service time $d^{\omega',2}_{k,t_0+t_1}$. ω performs a hand-off at t_0 and sends the flexible server to the second station with a service time of $f^{\omega,2}_{k,t_0}$. At t_0+t_1, σ^2_d becomes available, and the flexible server hands off its job to σ^2_d with residual service time $d^{\omega,2}_{k,t_0} - t_1$. Therefore at $t_0 + t_1$, $D^{\omega}(n) < D^{\omega'}(n)$. As the servers are indistinguishable, without loss of generality, we can assume that under ω' at time t_0+t_1 the two servers at the first station swap their jobs. Figure 3.7 illustrates this case.

Consider the system at time $t_0 + t_1$ and let $t_2 = \min\{d^1_{k+1,t_0+t_1}, f^1_{k+2,t_0+t_1}\}$.

Case 2.1: Assume $d^{\omega,2}_{k,t_0} - t_1 < t_2 < d^{\omega',2}_{k,t_0+t_1}$. At time $t_0+t_1+t_2$, ω sends σ^1_d's job

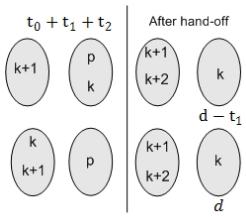

Figure 3.7: Theorem 3.1 - Case 2

(job k+1) to the second station and admits the $k + 3^{rd}$ job. At time $t_0+t_1+t_2$, ω' performs a hand-off between σ_d^1 and σ_f^1 (jobs k+1 and k+2) and the flexible server is sent to the second station. Therefore $D^{\omega}(n+1) = D^{\omega'}(n+1) = t_0+t_1+t_2$. Let $t_3 = \min\{d^{\omega',2}_{k,t_0+t_1} - t_2, f^2_{k+1,t_0+t_1+t_2}\}$. Figure 3.8 illustrates this case. We divide this case further.

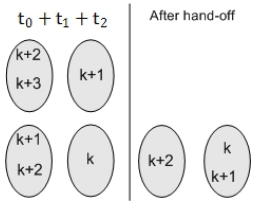

Figure 3.8: Theorem 3.1 - Case 2.1

Case 2.1.1: Assume $\min\{f^{\omega,1}_{k+3,t_0+t_1+t_2}, d^1_{k+2,t_0+t_1+t_2}\} < t_3$. This means that under ω a service completion occurs in the first station earlier than a service completion in the second station under ω'. Therefore $D^{\omega}(n+2) = t_0 + t_1 + t_2 + t_3 < D^{\omega'}(n+2)$. Figure 3.9 illustrates this case.

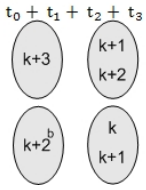

Figure 3.9: Theorem 3.1 - Case 2.1.1

Case 2.1.2: Assume $\min\{f^{\omega,1}_{k+3,t_0+t_1+t_2}, d^{1}_{k+2,t_0+t_1+t_2}\} > t_3$. This case is similar to Case 1.2.1.1. Figure 3.10 illustrates this case.

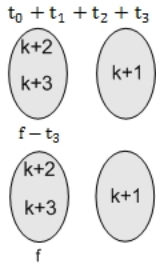

Figure 3.10: Theorem 3.1 - Case 2.1.2

Case 2.1.3: Assume $\min\{f^{\omega,1}_{k+3,t_0+t_1+t_2}, d^{1}_{k+2,t_0+t_1+t_2}\} = t_3$. This is similar to Case 2.1.2.

Case 2.2: Assume $t_2 < d^{\omega,2}_{k,t_0} - t_1$. This is similar to Case 1.2. Figure 3.11 illustrates this case.

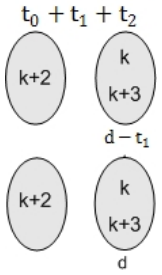

Figure 3.11: Theorem 3.1 - Case 2.2

Case 2.3: Assume $d^{\omega',2}_{k,t_0+t_1} \leq t_2$. Then at $t_0+t_1+d^{\omega',2}_{k,t_0+t_1}$ the system is the same under both policies. We have $D^{\omega}(n+1) = D^{\omega'}(n+1) = t_0+t_1+t_2$. Figure 3.12 illustrates this case.

Case 2.4: Assume $t_2 = d^{\omega,2}_{k,t_0} - t_1$. We have $D^{\omega}(n+1) = t_0+t_1+t_2 < D^{\omega'}(n+1)$.

The *ahead* property is required when we make cross references between cases. For

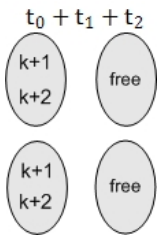

Figure 3.12: Theorem 3.1 - Case 2.3

example, it is stated that Case 1.2.2.1.1 is similar to Case 3.2. In order to be able to adapt Case 3.2, we need to ensure all the assumptions made in Cases 3.2 and 3 hold. In other words, if Case 3.2 is applied to a system in which the *ahead* property does not hold, it is possible to have earlier departures under ω'. For example if Case 3.2 is applied to a situation similar to Figure 3.5, with the exception that job $k+3$ has smaller residual service time under ω', the *ahead* property would be violated and departures would happen earlier under ω'.

(Lemma 3.1) □

Lemma 3.2: *For the second scenario* $\forall n.(D^{\omega}(n) \leq D^{\omega'}(n) \wedge ahead(n)) \Rightarrow (D^{\omega}(n+1) \leq D^{\omega'}(n+1))$.

Proof: The proof is by induction.

Basis step:

If time t_0 is not reached, both systems remain the same and there is nothing to prove. Otherwise, start the system from the empty state and let the first departure from the first station happen ($D^{\omega}(1) = D^{\omega'}(1)$). If the flexible server serves the departed task in the second station, the system follows the second scenario. σ_d^1 is serving job 2, σ_f^2 is serving job 1, and σ_d^2 is free. Up until time t_0, the two policies

are the same. When time t_0 is reached, the following cases are possible.

Case 3: Assume $d^1_{2,t_0} < f^2_{1,t_0}$, meaning that the dedicated server completes its service before the flexible server. Let $t_1 = d^1_{2,t_0}$. Under policy ω', σ^1_d completes its service after t_1 time units. So at $t_0 + t_1$, the completed job is sent to the second station with corresponding residual service time d^2_{2,t_0+t_1}. ω performs a hand-off at time t_0 and sends the flexible server to the first station with residual service time $f^{\omega,1}_{3,t_0}$. At time $t_0 + t_1$, σ^1_d completes its service, and the flexible server hands off its job to σ^1_d with residual service time $d^{\omega,1}_{3,t_0} - t_1$. The flexible server goes to the second station with residual service time f^2_{2,t_0+t_1}. Hence $D^{\omega}(2) = D^{\omega'}(2) = t_0 + t_1$.

Case 4: Now assume $d^1_{2,t_0} > f^2_{1,t_0}$, meaning that the flexible server completes its service before the dedicated server. Let $t_1 = f^2_{1,t_0}$. Under ω', σ^2_f becomes available after t_1 time units. So at $t_0 + t_1$, the flexible server is sent to the first station with residual service time $f^{\omega',1}_{3,t_0+t_1}$ and σ^2_d is free. ω performs a hand-off at t_0 and sends the flexible server to the first station with residual service time $f^{\omega,1}_{3,t_0}$. Now if $f^{\omega,1}_{3,t_0} - t_1 < d^1_{2,t_0+t_1}$, $D^{\omega}(2) < D^{\omega'}(2)$. Otherwise $D^{\omega}(2) = D^{\omega'}(2)$.

Inductive step:

If t_0 has occurred in the basis step, Lemma 3.3 shows that for each case considered in the inductive step, the system respects the *ahead* property. Hence, departures under ω' will not occur sooner than under ω. If time t_0 is not reached, the two systems remain the same and there will be nothing to prove at this stage.

If time t_0 occurs after the $n-1^{st}$ departure but before the n^{th} departure, consider the following configuration: a busy dedicated server at the first station (job k) and a free dedicated server and a busy flexible server (job p) at the second station. Up

until time t_0, the two policies are the same. When time t_0 is reached, the following cases are possible.

Case 3: Assume $d^1_{k,t_0} < f^2_{p,t_0}$, meaning that the dedicated server completes its service before the flexible server. Let $t_1 = d^1_{k,t_0}$. Under policy ω', σ^1_d completes its service after t_1 time units. So at $t_0 + t_1$, the completed job is sent to the second station with residual service time d^2_{k,t_0+t_1}. There is also an admission at the first station with residual service time $d^{\omega',1}_{k+1,t_0+t_1}$. ω performs a hand-off at t_0 and sends the flexible server to the first station with residual service time $f^{\omega,1}_{k+1,t_0}$. At $t_0 + t_1$, σ^1_d completes its service, and the flexible server hands off its job to σ^1_d with residual service time $d^{\omega,1}_{k+1,t_0} - t_1$. The flexible server goes to the second station with residual service time f^2_{k,t_0+t_1}. Figure 3.13 illustrates this case.

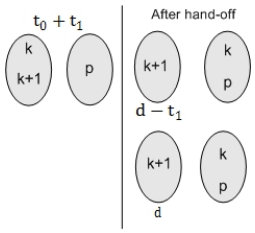

Figure 3.13: Theorem 3.1 - Case 3

Let $t_2 = \min\{d^2_{k,t_0+t_1}, f^2_{p,t_0+t_1}\}$.

Case 3.1: Assume $d^{\omega',1}_{k+1,t_0+t_1} \leq t_2$. Then $D^{\omega}(n) = D^{\omega'}(n) = t_0 + t_1 + t_2$.

Case 3.2: Assume $d^{\omega,1}_{k+1,t_0} - t_1 > t_2$. We make a simplifying assumption that job p is served before job k (which is a valid assumption considering homogeneity of servers and possibility of hand-off). At $t_0 + t_1 + t_2$, σ^1_d is serving job $k+1$ and σ^2_d is serving job k under both policies. The residual service time of job $k+1$ is $d^{\omega,1}_{k+1,t_0} - t_1 - t_2$ under ω and is $d^{\omega',1}_{k+1,t_0+t_1} - t_2$ under ω'. This is similar to Case 1.2.2.1.1. Figure 3.14 illustrates this case. We further divide this case.

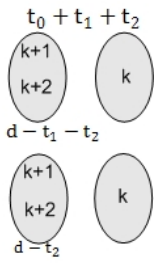

Figure 3.14: Theorem 3.1 - Case 3.2

Case 3.2.1: Assume $\min\{d^{\omega,1}_{k+1,t_0} - t_1 - t_2, f^1_{k+2,t_0+t_1+t_2}\} < d^2_{k,t_0+t_1}$. We further divide this case.

Case 3.2.1.1: Further assume $d^{\omega,1}_{k+1,t_0} - t_1 - t_2 < f^1_{k+2,t_0+t_1+t_2} < d^{\omega',1}_{k+1,t_0+t_1} - t_2$. Then $D^{\omega}(n+1) < D^{\omega'}(n+1)$.

Case 3.2.1.2: Assume $f^1_{k+2,t_0+t_1+t_2} \leq d^{\omega,1}_{k+1,t_0} - t_1 - t_2$. This is similar to Case 3. Figure 3.15 illustrates this case. Although this reference seems circular, the fact that jobs k and $k+2$ eventually leave the system avoids this.

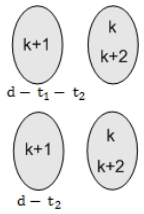

Figure 3.15: Theorem 3.1 - Case 3.2.1.2

Case 3.2.1.3: Assume $d^{\omega',1}_{k+1,t_0+t_1} - t_2 \leq f^1_{k+2,t_0+t_1+t_2}$. Then $D^{\omega}(n+1) < D^{\omega'}(n+1)$.

Case 3.2.2: Assume $\min\{d^{\omega,1}_{k+1,t_0} - t_1 - t_2, f^1_{k+2,t_0+t_1+t_2}\} \geq d^2_{k,t_0+t_1}$. This is similar to Case 3.2.1.

Case 3.3: Assume $d^{\omega,1}_{k+1,t_0} - t_1 < t_2 < d^{\omega',1}_{k+1,t_0+t_1}$. This is similar to Case 3.2.

Case 3.4: Assume $t_2 = d^{\omega,1}_{k+1,t_0} - t_1$. This is similar to Case 3.2.1.1.

Case 4: Now assume $d^1_{k,t_0} > f^2_{p,t_0}$, meaning that the flexible server completes its service before the dedicated server. Let $t_1 = f^2_{p,t_0}$. Under policy ω', σ^2_f becomes available after t_1 time units. So at t_0+t_1, the flexible server is sent to the first station with residual service time $f^{\omega',1}_{k+1,t_0+t_1}$ and σ^2_d is free. ω performs a hand-off at t_0 and sends the flexible server to the first station with residual service time $f^{\omega,1}_{k+1,t_0}$. Figure 3.16 illustrates this case.

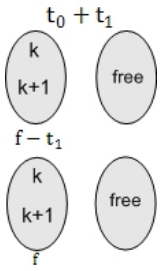

Figure 3.16: Theorem 3.1 - Case 4

Let $t_2 = d^1_{k,t_0+t_1}$.

Case 4.1: Assume $f^{\omega,1}_{k+1,t_0} - t_1 < t_2$. Then $D^{\omega}(n) = t_0 + t_1 + f^{\omega,1}_{k+1,t_0} - t_1 < D^{\omega'}(n) = t_0 + t_1 + t_2$. This is similar to Case 1.2.2.1.1.

Case 4.2: Assume $f^{\omega,1}_{k+1,t_0} - t_1 > t_2$. This is similar to Case 3.2. Figure 3.17 illustrates this case.

Case 4.3: Assume $t_2 = f^{\omega,1}_{k+1,t_0} - t_1$. Then $D^{\omega}(n) = D^{\omega}(n+1) = t_0 + t_1 + t_2 < D^{\omega'}(n+1)$.

(Lemma 3.2) □

The following lemma shows that after each departure, the system state corresponds

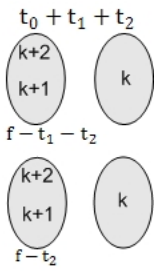

Figure 3.17: Theorem 3.1 - Case 4.2

to one of the cases described in Lemmas 3.1 or 3.2.

Lemma 3.3: $\forall m > 2. ahead(m) \Rightarrow ahead(m+1)$.

Proof: To show this we should connect the output of each case in Lemmas 3.1 and 3.2 (the configuration after $n+1$ departures) to the assumed initial state for some other case. However, instead of performing an exhaustive evaluation, we find that we can group cases together in a generic manner. We identify these generic cases and describe how they connect.

Consider the following generic cases: *g1*: different jobs in the first station and no blocking in the future (until the next departure under ω); *g2*: different jobs in the first station with blocking in the future (until the next departure under ω); *g3*: identical jobs with identical residual service times in the first station and identical jobs in the second station; *g4*: identical jobs with identical residual service times in the first station and different jobs in the second station; *g5*: identical jobs with different residual service times in the first station and identical jobs in the second station; *g6*: identical jobs with identical residual service times in the first station and identical jobs with different residual service times in the second station.

Note that in all of the generic cases, $ahead(m)$ holds. Starting from a generic case,

when a departure occurs from the first station under ω, the system either navigates to another generic case or it becomes the same under both policies (shown below). In either case, $ahead(m+1)$ holds, as both the generic cases and the situation when systems are the same respect the *ahead* property.

Assume ω' is serving a job in the first station that is already served by ω (e.g. Case 1.1 or 1.2.1.1 in Lemma 3.1). The worst case scenario (for keeping the system ahead under ω) is when ω' serves its job in the first station and admits the job ω is currently serving in the first station (label this job k). Now the residual service time of this job is t time units less under ω compared to ω'. *g1*: if there is no blocking, job k departs sooner under ω than ω' and the system stays in *g1*. *g2*: otherwise if job k is blocked under ω, the same would happen under ω' up to t time units later, as the same jobs (or jobs previously served at the second station by ω) will be served at the second station under ω' (Cases 1.3 and 1.2.2.1.4 in Lemma 3.1 are examples). From *g2*, the system can go to either of the following generic cases (i.e. *g3-g6*).

For *g3-g6*, assume that a specific job is being served in the first station under both policies. Let the residual service times of the job at the first station be identical. *g3*: if the states of the jobs are identical (i.e. the same jobs with identical residual service times) in the second station under both policies, then the policies behave the same (e.g. Case 2.3 in Lemma 3.1). *g4*: if ω has admitted new jobs in the second station, ω is already ahead in the number of departures and ω' needs to go through the same steps (e.g. Case 1.2.2.1.2 in Lemma 3.1). The system stays in *g4*. *g6*: if there are identical jobs at the second station with smaller residual service time under ω, jobs depart earlier from the second station under ω (e.g. Cases 1.2.1.2 and 1.2.2.2 in Lemma 3.1). The system either stays in *g6* or becomes the same under both policies.

Now assume the residual service time under ω is less in the first station. *g5*: if jobs are in the same state (i.e. the same jobs with identical residual service times) in the second station under both policies, ω can lead to earlier departures from the first station (e.g. Cases 1.2.2.1.1 and 3.2 in Lemmas 1 and 2, respectively). The system can go to *g1* or *g2*.

(Lemma 3.3) □

3.2.3 Non-idling property

In Theorem 3.1 we proved that an optimal policy uses hand-off whenever possible. Now in Theorem 3.2, we aim to show that an optimal policy avoids idling the flexible server. In the proof of Theorem 3.2, we will use the results of Theorem 3.1, i.e. the two policies used in the proof perform hand-offs whenever possible. Having said that, we only claim non-idling to be a property of an optimal policy if hand-off is employed. Otherwise there might be cases where non-idling leads to better results.

For example, assume at time t_0, jobs k and $k+1$ are being served by dedicated servers at the first and second stations, respectively, and that policies do not perform hand-off. At t_0, a non-idling policy assigns the flexible server to the first station to serve job $k+2$. An idling policy lets the flexible server idle at t_0. Now assume σ_d^1 completes serving its job sooner than the other servers. At $t_0 + d^1_{k,t_0}$, under the non-idling policy σ_d^1 (job k) becomes blocked. Under the idling policy the flexible server is assigned to the second station to admit job k and σ_d^1 admits job $k+2$. Hence, assuming there have been $n-1$ departures from the first station for both policies, $D^{idling}(n) < D^{non\text{-}idling}(n)$ which means idling the flexible server leads to better results. As a direct result of such a mechanism, our proof of Theorem 3.2 below would not go through.

Theorem 3.2. *The optimal policy is non-idling.*

Proof: Let π' and π be non-idling policies, i.e. whenever the flexible server becomes idle, the policy assigns it to:

1. the second station, if the first station is blocked;

2. the first station, otherwise.

The only exception is that π' idles the flexible server once for only t time units starting from t_0.

Only two other non-idling policies exist. The first is a policy that assigns the idle flexible server to the first station, if the first station is blocked. But as hand-off is used, the flexible server performs a hand-off and takes the blocked job to the second station, resulting in the same behavior as π. The second policy assigns the idle flexible server to the second station, if the first station is not blocked. Again, as hand-off is used, the flexible server passes its job to the free dedicated server in the second station and admits a new job at the first station, leading to the same behavior as π. Therefore we only need to consider π and π' throughout the rest of the proof. There are three scenarios under which idling is possible:

1. a blocked server at the first station and a busy server at the second station
2. two busy servers, one at each station
3. one busy server at the first station

For the third scenario, once the job at the first station is completed, everything would be the same as the second scenario. Therefore we do not consider the third scenario separately. Define $D^{\pi}(n)$ to be the time at which the n^{th} departure from the first station occurs under π. We need to show $\forall n, m.\ (D^{\pi}(n) < D^{\pi'}(n) \wedge m > n) \Rightarrow (D^{\pi}(m) \leq D^{\pi'}(m))$.

The rest of the proof is structured as follows:

For the first scenario, Lemma 3.4 states that if the n^{th} departure happens sooner under π than π' and *ahead(n)* holds, then the $n{+}1^{st}$ departure under π' will not occur sooner under π.

For the second scenario, Lemma 3.5 states that if the n^{th} departure happens sooner under π than π' and *ahead(n)* holds, then the $n{+}1^{st}$ departure under π' will not occur sooner under π.

For both scenarios, Lemma 3.6 completes the proof by showing that given a system with $m > 1$ departures from the first station that satisfies *ahead(m)*, Lemmas 3.4 or 3.5 can be applied iteratively on the system to satisfy *ahead(m+1)*.

Given Lemmas 3.4, 3.5, and 3.6, a simple induction argument follows.

(Theorem 3.2) □

Lemma 3.4. *For the first scenario,* $\forall n \;.\; (D^{\pi}(n) \leq D^{\pi'}(n) \wedge ahead(n)) \Rightarrow (D^{\pi}(n+1) \leq D^{\pi'}(n+1))$.

Lemma 3.5. *For the second scenario,* $\forall n \;.\; (D^{\pi}(n) \leq D^{\pi'}(n) \wedge ahead(n)) \Rightarrow (D^{\pi}(n+1) \leq D^{\pi'}(n+1))$.

Lemma 3.6. $\forall m > 2 \;.\; ahead(m) \Rightarrow ahead(m+1)$.

A natural extension of Theorem 3.2 is as follows: introducing idling to any non-idling policy decreases the throughput, assuming hand-off is used. If hand-off is not used, a non-idling policy can lead to better results, as discussed at the beginning of Section 3.2.3. Theorem 3.2 shows that after the idling period ($[t_0, t_0 + t]$), the system

is *ahead* under the policy which is always non-idling. It also shows that if a non-idling policy is applied to the system, the *ahead* property still holds.

To investigate the proposed extension, we could follow an approach similar to the proof of Theorem 3.2. Given two idling policies, we make one of them non-idling for a period of time. Theorem 3.2 shows that after this period, the system is *ahead* under the changed policy. We believe that if an idling policy is applied to the system, the *ahead* property still holds and hence the proof follows. However we are unable to provide a proof for this belief.

3.2.4 Proof of Lemmas 3.4, 3.5, and 3.6

In this subsection, we provide the proofs of Lemmas 3.4, 3.5, and 3.6. In Lemmas 3.4 and 3.5, let t_0 be the time at which the two policies make a different choice.

Lemma 3.4: *For the first scenario,* $\forall n.(D^{\pi}(n) < D^{\pi'}(n) \wedge ahead(n)) \Rightarrow (D^{\pi}(n+1) \leq D^{\pi'}(n+1))$.

Proof: The two primary steps of the proof (basis and inductive steps) are shown in the following.

Basis step:

If time t_0 is not reached, both systems remain the same and there is nothing to prove. Otherwise, start the system from the empty state and let the first departure from the first station ($D^{\pi}(1) = D^{\pi'}(1)$) occur. If $d^1_{2,t_0} < d^2_{1,t_0}$, the system follows the first scenario. σ^1_d is blocked and is holding job 2 and σ^2_d is serving job 1. Up until time t_0, the two policies are the same. When time t_0 is reached, π employs the flexible

server to clear blocking by sending the flexible server to the second station to serve job 2 and admitting job 3 at the first station. Hence, $D^{\pi}(2) < D^{\pi'}(2)$.

Inductive step:

If t_0 has occurred in the basis step, Lemma 3.6 shows that for each case considered in the inductive step, the system respects the *ahead* property. Hence π' will not have departures occurring sooner than π. If time t_0 is not reached, the two systems remain the same and there will be nothing to prove at this stage.

If time t_0 occurs after the $n-1^{st}$ departure but before the n^{th} departure, consider this configuration: a dedicated blocked server at the first station and a busy dedicated server at the second station and an idle flexible server. At t_0, π sends the idle flexible server to the second station with $f^{\pi,2}_{k,t_0}$ as service time and therefore allows for an admission at the first station with $d^{\pi,1}_{k+1,t_0}$ as service time. We have $D^{\pi}(n) = t_0$. Assume π' idles the flexible server for t time units. Let $t_1 = d^2_{p,t_0}$.

Case 1: Assume $t < t_1$. At $t_0 + t$, π' sends the flexible server to the second station. We have $D^{\pi}(n) = t_0 < D^{\pi'}(n) = t_0 + t$. At $t_0 + t$, we are in a situation similar to Case T3.1-1.2 (case T3.1-*num* refers to Case *num* of Theorem 3.1).

Case 2: Now assume $t > t_1$. Figure 3.18 illustrates the situation at $t_0 + t_1$. We further divide this case.

Case 2.1: Assume $\min\{f^{\pi,1}_{k+1,t_0} - t_1, d^{\pi,1}_{k+2,t_0+t_1}\} \leq (t - t_1)$. Then a departure from the first station occurs sooner under π than π'. Therefore $D^{\pi}(n+1) < D^{\pi'}(n+1)$.

Case 2.2: Assume $\min\{f^{\pi,1}_{k+1,t_0} - t_1, d^{\pi,1}_{k+2,t_0+t_1}\} > (t - t_1)$. Then at $t_0 + t$, under π we have $d^{\pi,1}_{k+2,t_0+t_1} - (t - t_1)$ and $f^{\pi,1}_{k+1,t_0} - t$ as residual service times for jobs $k+2$ and $k+1$, respectively. Under π' we have $d^{\pi',1}_{k+2,t_0+t}$ and $f^{\pi',1}_{k+1,t_0+t_1} - (t - t_1)$

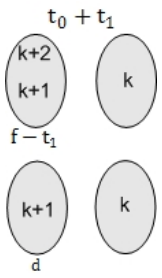

Figure 3.18: Theorem 3.2 - Case 2

as residual service times for jobs $k+2$ and $k+1$, respectively. This assures that a departure from the first station occurs sooner under π, as both jobs under π have smaller residual service times. Therefore $D^{\pi}(n+1) < D^{\pi'}(n+1)$.

Case 3: Assume $t = t_1$. This is similar to Case T3.1-3.2.

(Lemma 3.4) □

Lemma 3.5: *For the second scenario,* $\forall n.(D^{\pi}(n) < D^{\pi'}(n) \wedge ahead(n)) \Rightarrow (D^{\pi}(n+1) \leq D^{\pi'}(n+1))$.

Proof: The two primary steps of the proof (basis and inductive steps) are shown in the following.

Basis step:

If time t_0 is not reached, both systems remain the same and there is nothing to prove. Otherwise, start the system from the empty state and let the first departure from the first station ($D^{\omega}(1) = D^{\omega'}(1)$) occur. Now the system follows the second scenario. σ_d^1 is serving job 2 and σ_d^2 is serving job 1. Up until time t_0, the two policies are the same. When time t_0 is reached, π sends the idle flexible server to the first station to admit job 3. If $f_{3,t_0}^{\pi,1} < \min\{d_{2,t_0}^1, d_{1,t_0}^2\}$ or $d_{2,t_0}^1 < \min\{d_{1,t_0}^2, t\}$, then

$D^{\omega}(2) < D^{\omega'}(2)$. Otherwise, $D^{\omega}(2) = D^{\omega'}(2)$ with less residual service time for jobs under π.

Inductive step:

If t_0 has occurred in the basis step, Lemma 3.6 shows that for each case considered in the inductive step, the system respects the *ahead* property. Hence π' will not have departures occurring sooner than π. If time t_0 is not reached, the two systems remain the same and there will be nothing to prove at this stage.

If time t_0 occurs after the $n-1^{st}$ departure but before the n^{th} departure, consider this configuration: a dedicated busy server at the first station, a dedicated busy server at the second station and a flexible idle server. At t_0, π sends the idle flexible server to the first station with $f^{\pi,1}_{k+1,t_0}$ as service time. Assume π' idles the flexible server for t time units. Let $t_1 = d^2_{p,t_0}$ and $t_2 = d^1_{k,t_0}$.

Case 3: Assume $t_2 < t_1$. Let $t_f = f^{\pi,1}_{k+1,t_0}$ be the time where π sends the flexible server to the second station. We further divide this case.

Case 3.1: Assume $t < t_2$. We further divide this case.

Case 3.1.1: Assume $t_f < t$. Figure 3.19 illustrates the situation at $t_0 + t_f$. Let $t'_f = f^{\pi,2}_{k+1,t_0+t_f}$. We further divide this case.

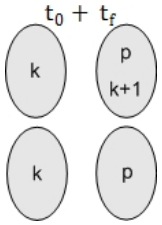

Figure 3.19: Theorem 3.2 - Case 3.1.1

Case 3.1.1.1: Assume $t'_f < t$. At $t_0 + t_f + t'_f$, π completes serving

job $k+1$ and sends the flexible server to the first station. Figure 3.20 illustrates the situation. We further divide this case.

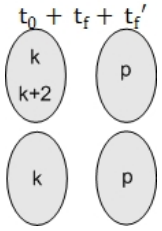

Figure 3.20: Theorem 3.2 - Case 3.1.1.1

Case 3.1.1.1.1: Assume $f^{\pi,1}_{k+2,t_0+t_f+t'_f} \leq t$. Therefore $D^{\pi}(n+1) < D^{\pi'}(n)$.

Case 3.1.1.1.2: Assume $f^{\pi,1}_{k+2,t_0+t_f+t'_f} > t$. At t_0+t, π' sends the flexible server to the first station to admit job $k+1$. Therefore, under π jobs k and $k+2$ are at the first station and under π' jobs k and $k+1$ are at the first station. This is similar to Case 4.2.1.2.

Case 3.1.1.2: Assume $t'_f > t$. At t_0+t, π' sends the flexible server to the first station. Figure 3.21 illustrates this case. Let $\hat{t}_f = f^{\pi',1}_{k+1,t_0+t}$. We further divide this case.

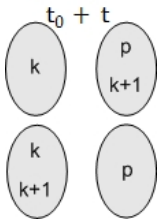

Figure 3.21: Theorem 3.2 - Case 3.1.1.2

Case 3.1.1.2.1: Assume $\hat{t}_f < t'_f - (t - t_f)$. At $t_0 + t_f + \hat{t}_f$, job k is at the first station and job p is at the second station under both policies. Under π, the residual service time of job $k+1$ is

$f^{\pi,2}_{k+1,t_0+t_f} - (t + \hat{t}_f - t_f)$. Under π', the residual service time of job $k+1$ is $f^{\pi',1}_{k+1,t_0+t_f+\hat{t}_f}$. This is similar to Case T3.1-1.2.

Case 3.1.1.2.2: Assume $\hat{t}_f > t'_f - (t - t_f)$. At $t_0 + t + \hat{t}_f$, π completes serving job $k+1$ and sends the flexible server to the first station to admit job $k+2$. This is similar to Case 4.2.1.2.

Case 3.1.1.2.3: Assume $\hat{t}_f = t'_f - (t - t_f)$. Then $D^{\pi}(n+1) < D^{\pi'}(n+1)$.

Case 3.1.1.3: Assume $t'_f = t$. Then $D^{\pi}(n+1) < D^{\pi'}(n+1)$.

Case 3.1.2: Assume $t \leq t_f < t_2$. At $t_0 + t$, job k is at the first station and job p is at the second station under both policies. Under π, the residual service time of job $k+1$ is $f^{\pi,1}_{k+1,t_0} - t$. Under π', the residual service time of job $k+1$ is $f^{\pi',1}_{k+1,t_0+t}$. At $t_0 + t_f$, π sends the flexible server to the second station, leading to $D^{\pi}(n) = t_0 + t_f < D^{\pi'}(n)$. We further divide this case.

Case 3.1.2.1: Assume $t'_f < f^{\pi',1}_{k+1,t_0+t}$. π completes serving job $k+1$ and sends the flexible server to the first station to admit job $k+2$. This case is similar to Case 4.2.1.2.

Case 3.1.2.2: Assume $t'_f > f^{\pi',1}_{k+1,t_0+t}$. π' sends job $k+1$ with the flexible server to the second station. This case is similar to Case T3.1-1.2.

Case 3.1.2.3: Assume $t'_f = f^{\pi',1}_{k+1,t_0+t}$. This case is similar to Case 3.1.1.2.3.

Case 3.1.3: Assume $t_2 < t_f < t_1$. Figure 3.22 illustrates the situation at $t_0 + t_2$. Therefore $D^{\pi}(n) = D^{\pi'}(n)$. Let $\breve{t}_f = f^2_{k,t_0+t_2}$. We further divide this case.

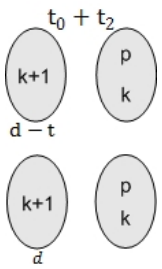

Figure 3.22: Theorem 3.2 - Case 3.1.3

Case 3.1.3.1: Assume $d^{\pi,1}_{k+1,t_0} - t \leq \min\{d^2_{p,t_0+t_2}, \breve{t}_f\} < d^{\pi',1}_{k+1,t_0+t}$. Therefore $D^{\pi}(n+1) < D^{\pi'}(n+1)$.

Case 3.1.3.2: Assume $d^{\pi',1}_{k+1,t_0+t} \leq \min\{d^2_{p,t_0+t_2}, \breve{t}_f\}$. Job $k+1$ becomes blocked under both policies and therefore $D^{\pi}(n+1) = D^{\pi'}(n+1)$.

Case 3.1.3.3: Assume $\breve{t}_f < d^{\pi,1}_{k+1,t_0} - t$. Job $k+2$ is at the first station and job p is at the second station under both policies. Under π, the residual service time of job $k+1$ is $d^{\pi,1}_{k+1,t_0} - t_2 - \breve{t}_f$. Under π', the residual service time of job $k+1$ is $d^{\pi',1}_{k+1,t_0+t} - (t_2 + \breve{t}_f - t)$. This is similar to Case T3.1-3.2.

Case 3.1.4: Assume $t_1 < t_f$. This is similar to Case 3.1.3.

Case 3.1.5: Assume $t_f = t_1$. This is similar to Case 3.1.1.3.

Case 3.1.6: Assume $t_f = t_2$. Then $D^{\pi}(n+1) < D^{\pi'}(n+1)$.

Case 3.2: Assume $t_2 \leq t < t_1$.

Case 3.2.1: Assume $t_f < t_2$. Figure 3.19 illustrates this case at $t_0 + t_f$. We further divide this case.

Case 3.2.1.1: Assume $t'_f < t_2$. At $t_0 + t_f + t'_f$, π completes serving job $k+1$ and sends the flexible server to the first station to admit job $k+2$. Departure of either job k or $k+2$ leads to $D^{\pi}(n+1) < D^{\pi'}(n+1)$.

Case 3.2.1.2: Assume $t'_f > t_2$. At $t_0 + t_2$ job k becomes blocked under both policies. We further divide this case.

Case 3.2.1.2.1: Assume $f^{\pi,2}_{k+1,t_0+t_2} < t - t_2$. Therefore $D^{\pi}(n+1) < D^{\pi'}(n)$.

Case 3.2.1.2.2: Assume $f^{\pi,2}_{k+1,t_0+t_2} > t - t_2$. At $t_0 + t$, job k is blocked at the first station under π and is being served at the first station under π'. Job p is at the second station. Under π, the residual service time of job $k+1$ is $f^{\pi,2}_{k+1,t_0+t_f} - (t - t_f)$. Under π', the residual service time of job $k+1$ is $f^{\pi',2}_{k+1,t_0+t}$. We further divide this case.

Case 3.2.1.2.2.1: Assume $\min\{d^2_{p,t_0+t}, f^{\pi,2}_{k+1,t_0+t}\} > d^{\pi',1}_{k,t_0+t}$. If π' completes serving job k, both policies result in the same situation and therefore $D^{\pi}(n+1) = D^{\pi'}(n+1)$.

Case 3.2.1.2.2.2: Assume $\min\{d^2_{p,t_0+t}, f^{\pi,2}_{k+1,t_0+t}\} \leq d^{\pi',1}_{k,t_0+t}$. Blocking is cleared and π sends job k to the second station, leading to $D^{\pi}(n+1) < D^{\pi'}(n+1)$.

Case 3.2.1.2.3: Assume $f^{\pi,2}_{k+1,t_0+t_2} = t - t_2$. This is the same as Case 3.1.1.3.

Case 3.2.1.3: Assume $t'_f = t_2$. Then $D^{\pi}(n+1) = D^{\pi'}(n) < D^{\pi'}(n+1)$.

Case 3.2.2: Assume $t_2 < t_f < t$. Let $t''_f = f^{\pi,2}_{k,t_0+t_2}$. Figure 3.23 illustrates the situation at $t_0 + t_2$. We further divide this case.

Case 3.2.2.1: Assume $t''_f < t_f$. Figure 3.24 illustrates the situation. We further divide this case.

Case 3.2.2.1.1: Assume $\min\{d^{\pi,1}_{k+1,t_0+t_1+t''_f}, f^{\pi,1}_{k+2,t_0+t_1+t''_f}\} \leq t_1$.

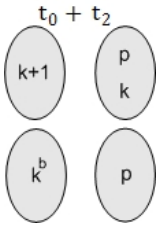

Figure 3.23: Theorem 3.2 - Case 3.2.2

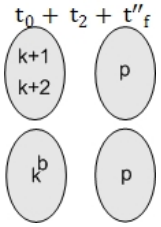

Figure 3.24: Theorem 3.2 - Case 3.2.2.1

Therefore $D^{\pi}(n+1) < D^{\pi'}(n)$.

Case 3.2.2.1.2: Assume $\min\{d^{\pi,1}_{k+1,t_0+t_1+t''_f}, f^{\pi,1}_{k+2,t_0+t_1+t''_f}\} > t_1$. At t_0+t_1, under π, the residual service time of job $k+1$ is $d^{\pi,1}_{k+1,t_0} - t_1$. Under π', the residual service time of job $k+1$ is $d^{\pi',1}_{k+1,t_0+t_1}$. This is similar to Case T3.1-2.1.

Case 3.2.2.2: Assume $t''_f > t_f$. Figure 3.25 illustrates the situation. We further divide this case.

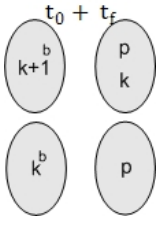

Figure 3.25: Theorem 3.2 - Case 3.2.2.2

Case 3.2.2.2.1: Assume $f^{\pi,2}_{k,t_0+t_f} \leq t-(t_0+t_f)$. Therefore $D^{\pi}(n+1) < D^{\pi'}(n)$.

Case 3.2.2.2.2: Assume $f^{\pi,2}_{k,t_0+t_f} > t-(t_0+t_f)$. Job p is at the second station and job $k+1$ is at the first station under both policies. Under π, the residual service time of job k is $f^{\pi,2}_{k,t_0+t_2} - (t-t_2)$. Under π, the residual service time of job k is $f^{\pi',2}_{k,t_0+t}$. This is similar to Case T3.1-1.2.

Case 3.2.2.3: Assume $t''_f = t_f$. Then $D^{\pi}(n+2) < D^{\pi'}(n)$.

Case 3.2.3: Assume $t < t_f < t_1$. We further divide this case.

Case 3.2.3.1: Assume $t''_f < t$. This is similar to Case 3.2.2.1.

Case 3.2.3.2: Assume $t''_f > t$. This is similar to Case 3.2.2.2.

Case 3.2.3.3: Assume $t''_f = t$. Then $D^{\pi}(n+1) < D^{\pi'}(n+1)$.

Case 3.2.4: Assume $t_1 < t_f$. This is similar to Case 3.2.3.

Case 3.3: Assume $t_1 \leq t$. We further divide this case.

Case 3.3.1: Assume $t_f < t_2$. We further divide this case.

Case 3.3.1.1: Assume $f^{\pi,2}_{k+1,t_0+t_f} < t_2$. This is similar to Case 3.2.1.1.

Case 3.3.1.2: Assume $f^{\pi,2}_{k+1,t_0+t_f} > t_2$. We further divide this case.

Case 3.3.1.2.1: Assume $f^{\pi,2}_{k+1,t_0+t_2} < t_1 - t_2$. This is similar to Case 3.2.1.2.1.

Case 3.3.1.2.2: Assume $f^{\pi,2}_{k+1,t_0+t_2} \geq t_1 - t_2$. Therefore $D^{\pi}(n+1) < D^{\pi'}(n)$.

Case 3.3.1.3: Assume $f^{\pi,2}_{k+1,t_0+t_f} = t_2$. Then $D^{\pi}(n+2) < D^{\pi'}(n)$.

Case 4: Assume $t_1 < t_2$. We further divide this case.

Case 4.1: Assume $t < t_1$. We further divide this case.

Case 4.1.1: Assume $t_f < t$. This is similar to Case 3.1.1. Let $t'_f =$

$f^{\pi,2}_{k+1,t_0+t_f}$.

Case 4.1.2: Assume $t \leq t_f < t_1$. This is similar to Case 3.1.2.

Case 4.1.3: Assume $t_1 \leq t_f < t_2$. Figure 3.28 illustrates the situation at $t_0 + t_f$. We have $D^{\pi}(n) = t_0 + t_f < D^{\pi'}(n)$. Let $t''_f = f^{\pi',1}_{k+1,t_0+t_f}$. We further divide this case.

Case 4.1.3.1: Assume $t'_f < t''_f$. Regardless of which task (k or $k+2$) departs next from the first station under π, $D^{\pi}(n+1) < D^{\pi'}(n+1)$.

Case 4.1.3.2: Assume $t''_f \leq t'_f$. Figure 3.26 illustrates the situation. This is similar to Case T3.1-2.1.

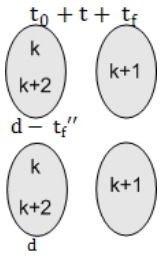

Figure 3.26: Theorem 3.2 - Case 4.1.3.2

Case 4.1.4: Assume $t_2 < t_f$. This is similar to Case 3.1.3.

Case 4.1.5: Assume $t_2 = t_f$. Then $D^{\pi}(n+1) = D^{\pi'}(n)$.

Case 4.2: Assume $t_1 < t < t_2$. Let $t_f = f^{\pi,1}_{k+1,t_0}$. We further divide this case.

Case 4.2.1: Assume $t_f < t_1$. At $t_0 + t_f$, π sends the flexible server with job $k+1$ to the second station, leading to $D^{\pi}(n) = t_0 + t_f$. We further divide this case.

Case 4.2.1.1: Assume $f^{\pi,2}_{k+1,t_0+t_f} < t_1$. Let $\bar{t}_f = f^{\pi,2}_{k+1,t_0+t_f}$. We further divide this case.

Case 4.2.1.1.1: Assume $f^{\pi,1}_{k+2,t_0+\bar{t}_f} \leq t$. Therefore $D^{\pi}(n+1) <$

$D^{\pi'}(n)$.

Case 4.2.1.1.2: Assume $t < f^{\pi,1}_{k+2,t_0+\bar{t}_f}$. Therefore $D^{\pi}(n+1) < D^{\pi'}(n+1)$.

Case 4.2.1.2: Assume $t_1 < f^{\pi,2}_{k+1,t_0+t_f} < t$. At $t_0 + t_1$, job p departs. Figure 3.27 illustrates this case. We further divide this case.

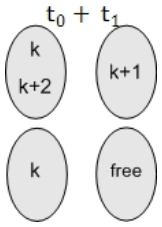

Figure 3.27: Theorem 3.2 - Case 4.2.1.2

Case 4.2.1.2.1: Assume $f^{\pi,1}_{k+2,t_0+t_1} \leq t$. Therefore $D^{\pi}(n+1) < D^{\pi'}(n)$.

Case 4.2.1.2.2: Assume $t < f^{\pi,1}_{k+2,t_0+t_1} < d^{1}_{k,t_0+t_1}$. At $t_0 + t$, π' sends the flexible server to the first station, as illustrated in Figure 3.28. Therefore $D^{\pi}(n+1) \leq D^{\pi'}(n)$.

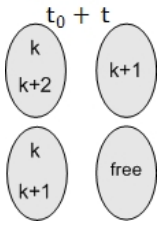

Figure 3.28: Theorem 3.2 - Case 4.2.1.2.2

Case 4.2.1.2.3: Assume $f^{\pi,1}_{k+2,t_0+t_1} \geq d^{1}_{k,t_0+t_1}$. We have $D^{\pi}(n+1) = t_0 + t_2 = D^{\pi'}(n) < D^{\pi'}(n+1)$.

Case 4.2.1.3: Assume $t \leq f^{\pi,2}_{k+1,t_0+t_f}$. This is similar to Case 4.2.1.1.2.

Case 4.2.1.4: Assume $t_1 = f^{\pi,2}_{k+1,t_0+t_f}$. This is similar to Case 4.2.1.2.

Case 4.2.2: Assume $t_1 < t_f < t$. Similar to Case 4.2.1.2, Figure 3.27 illustrates the situation at $t_0 + t$. We further divide this case.

Case 4.2.2.1: Assume under π, $d^1_{k,t_0+t} \leq f^{\pi,1}_{k+2,t_0+t}$ and under π', $d^1_{k,t_0+t} \leq f^{\pi',1}_{k+1,t_0+t}$. We have $D^{\pi}(n+1) = t_0 + t = D^{\pi'}(n) \leq D^{\pi'}(n+1)$.

Case 4.2.2.2: Assume $f^{\pi,1}_{k+2,t_0+t} \leq f^{\pi',1}_{k+1,t_0+t} < t_2$. We have $D^{\pi}(n+1) < D^{\pi'}(n)$.

Case 4.2.2.3: Assume $f^{\pi',1}_{k+1,t_0+t} < f^{\pi,1}_{k+2,t_0+t} < t_1$. Let $t_3 = f^{\pi',1}_{k+1,t_0+t}$. At $t_0 + t + t_3$, under π, the residual service time of job $k+2$ is $f^{\pi,1}_{k+2,t_0+t_f} - (t - t_f) - t_3$. Under π' the residual service time of job $k+2$ is $f^{\pi',1}_{k+2,t_0+t+t_3}$. Job k has the same residual service time under both policies. This is similar to Case T3.1-2.1.

Case 4.2.3: Assume $t < t_f \leq t_2$. At $t_0 + t$ under π, the residual service time of job $k+1$ is $f^{\pi,1}_{k+1,t_0} - t$. Under π', the residual service time of job $k+1$ is $f^{\pi',1}_{k+1,t_0+t}$. This is similar to Case T3.1-2.1.

Case 4.2.4: Assume $t_2 < t_f$. This case is similar to Case 4.2.3.

Case 4.2.5: Assume $t_f = t_1$. This is the same as Case 4.2.1.1.

Case 4.2.6: Assume $t_f = t$. This is the same as Case 4.2.1.2.2.

Case 4.3: Assume $t_2 < t$. We further divide this case.

Case 4.3.1: Assume $t_f < t_1$. Figure 3.19 illustrates the situation. We further divide this case.

Case 4.3.1.1: Assume $f^{\pi,2}_{k+1,t_0+t_f} \leq t_1$. Let $\bar{t}_f = f^{\pi,2}_{k+1,t_0+t_f}$. We further divide this case.

Case 4.3.1.1.1: Assume $f^{\pi,1}_{k+2,t_0+\bar{t}_f} \leq t_1$. Therefore $D^{\pi}(n+1) < D^{\pi'}(n)$.

Case 4.3.1.1.2: Assume $t_2 \leq f^{\pi,1}_{k+2,t_0+\bar{t}_f}$. Therefore $D^{\pi}(n+1) = D^{\pi'}(n) < D^{\pi'}(n+1)$.

Case 4.3.1.2: Assume $t_1 < f^{\pi,2}_{k+1,t_0+t_f} < t_2$. At t_0+t_1, job p departs. In a similar manner to Case 4.2.1.2, Figure 3.27 illustrates this case. We further divide this case.

Case 4.3.1.2.1: Assume $f^{\pi,1}_{k+2,t_0+t_1} \leq t_2$. We can immediately infer that $D^{\pi}(n+1) < D^{\pi'}(n+1)$.

Case 4.3.1.2.2: Assume $t_2 < f^{\pi,1}_{k+2,t_0+t_1} < t$. At t_0+t_2, job k is sent to the second station. Figure 3.29 illustrates the situation.

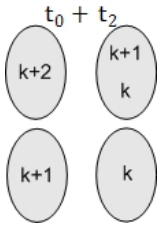

Figure 3.29: Theorem 3.2 - Case 4.3.1.2

Case 4.3.1.2.3: Assume $t \leq f^{\pi,1}_{k+2,t_0+t_1}$. This is similar to Case 4.3.1.2.2.

Case 4.3.1.3: Assume $t_2 \leq f^{\pi,2}_{k+1,t_0+t_f}$. Then $D^{\pi}(n+1) = D^{\pi'}(n) < D^{\pi'}(n+1)$.

Case 4.3.2: Assume $t_1 < t_f < t_2$. Figure 3.27 illustrates the situation at t_0+t_f.

Case 4.3.2.1: Assume $f^{\pi,1}_{k+2,t_0+t_1} \leq t_2$. Then $D^{\pi}(n+1) \leq D^{\pi'}(n)$.

Case 4.3.2.2: Assume $f^{\pi,1}_{k+2,t_0+t_1} > t_2$. Then $D^{\pi}(n+1) = D^{\pi'}(n) <$

$D^{\pi'}(n+1)$.

Case 4.3.3: Assume $t_2 < t_f < t$. This is similar to Case 4.3.2.

Case 4.3.4: Assume $t \leq t_f$. This is similar to Case 4.3.2.

Case 4.3.5: Assume $t_f = t_1$. This is the same as Case 4.2.1.2.

Case 4.3.6: Assume $t_f = t_2$. Then $D^{\pi}(n+1) = D^{\pi'}(n) < D^{\pi'}(n+1)$.

(Lemma 3.5) □

The following lemma shows that after each departure, the system state corresponds to one of the cases described in Lemmas 3.4 or 3.5.

Lemma 3.6: $\forall m > 2. ahead(m) \Rightarrow ahead(m+1)$.

Proof: To show this we should connect the output of each case in Lemmas 3.4 and 3.5 (the configuration after $n+1$ departures) to the assumed initial state for some other case. However instead of performing an exhaustive evaluation, we find that we can group cases together in a generic manner. We identify these generic cases and describe how they connect.

Consider the following generic cases: *g1*: different jobs in the first station and no blocking in the future (until the next departure under π); *g2*: different jobs in the first station with blocking in the future (until the next departure under π); *g3*: identical jobs with identical residual service times in the first station and identical jobs in the second station; *g4*: identical jobs with identical residual service times in the first station and different jobs in the second station; *g5*: identical jobs with different residual service times in the first station and identical jobs in the second station; *g6*: identical jobs with different residual service times in the first station and different

jobs in the second station.

Assume π' is serving a job in the first station that has already been served by π (e.g. Cases 4.3.2 and 4.3.1.3 in Lemma 3.5). The worst case scenario (for keeping the system ahead under π) is when π' serves its job in the first station and admits the job π is currently serving in the first station (label this job k). Now the residual service time of this job is t time units less under π compared to π'. *g1*: if there is no blocking, job k departs sooner under π than π' and the system stays in *g1*. *g2*: otherwise if job k is blocked under π, the same would happen under π' up to t time units later, as the same jobs (or jobs previously served at the second station by π) will be served at the second station under π' (Cases 3.2.1.2.1 and 3.2.2.1.1 in Lemma 3.5 are examples). From *g2*, the system can go to any of the following generic cases (i.e. *g3-g6*).

For *g3-g6*, assume that a specific job is being served in the first station under both polices. Let the residual service times of the job at the first station be identical. *g3*: if the jobs are identical (i.e. the same jobs with identical residual service times) in the second station under both policies, then the policies behave the same (e.g. Case 3.1.3.2 in Lemma 3.5). *g4*: if π has admitted new jobs in the second station, π is already ahead in the number of departures and π' needs to go through the same steps (e.g. Cases 3.1.1.1.1, 4.2.1.1, and 4.3.1.1.1 in Lemma 3.5). The system stays in *g4*. Now assume the residual service time under π is less in the first station. *g5*: if jobs are in the same state (i.e. the same jobs with identical residual service times) in the second station under both policies, π can lead to earlier departures from the first station (e.g. Cases 3.1.3.3 and 4.2.3 in Lemma 3.5). The system can go to *g1* or *g2*. *g6*: if π' is serving jobs previously served by π in the second station, with an

argument similar to *g2*, π leads to better results (e.g. Cases 4.1.3.2, 4.2.2.1.3, and 3.2.2.2.2 in Lemma 3.5).

(Lemma 3.6) $\square$

3.2.5 Extensions

In this subsection, we show that the results of Theorems 3.1 and 3.2 can be extended to systems with arrivals, clearing systems, and systems with additional dedicated or flexible servers. In the context of our work, clearing systems are lines in which a fixed number of jobs are waiting at the first station at time 0 and there are no arrivals.

Corollary 3.1: *Theorems 3.1 and 3.2 hold for i) systems where jobs are waiting at the first station, ii) systems with arrivals, and iii) clearing systems.*

Proof:

In the following we show the proof for Theorem 3.1. To construct a proof for Theorem 3.2, replace ω and ω' with π and π', respectively.

(i) The proofs above assume that jobs are always waiting at the first station.

(ii) Whenever decisions are made, if there are jobs waiting at the first station, the proof follows (i). If there are no jobs waiting at the first station, we show that ω' does not gain any benefit when there are no arrivals and $ahead(m)$ is not violated. The following cases could occur:

(a) If there are no free servers at the first station under both policies, the proof follows (i) as long as servers at the first station remain busy.

(b) If there are free servers at the first station under ω and no free servers at the first station under ω', ω' is serving a job at the first station that ω has already served. Let r be the time when the next arrival occurs and u be the remaining time until the systems become identical under both policies (i.e. the systems are serving the same jobs with the same residual service times).

If $r < u$, the proof continues as (i). If $r > u$, the systems are identical under both policies.

(c) If there are free servers at the first station under ω' and no free servers at the first station under ω, ω' is waiting for a job that ω is serving or has already served. Let r be the time when the next arrival occurs and u be the remaining time until a server becomes idle at the first station under ω. If $r < u$, when the next arrival happens, under ω there have been more departures and/or jobs have less residual service time ($ahead(m)$). If $r > u$, both policies should wait for the next departure. However, the system under ω has serviced more jobs and/or jobs have less residual service time ($ahead(m)$).

(d) If there are free servers at the first station under both policies and if both policies are waiting for the same job, the systems are identical under both policies. Otherwise ω' is waiting for a job that ω has already served. Upon the next arrival, the systems continue working with ω having a higher number of departures from the first station ($ahead(m)$).

In Theorem 3.2, let r be the time of the next arrival after time t_0. If $r \geq t$, the systems are the same under π and π'. If $r < t$, t is replaced with $t - r$ throughout the proof.

(iii) For a clearing system, the proof is the same as (i) up until the point where there are no more jobs to admit under ω (assuming time t_0 has passed). At this point either:

(a) the systems are identical; from here the policies behave the same and clear

the system at the same time.

(b) the same jobs are being served under both policies with smaller residual service times under ω; there are only three jobs left in the system and from Theorems 1 and 2 we know that under ω, the next departure does not happen later than ω'. Therefore the system reaches a time where there are two jobs in the system under ω and either two or three jobs under ω'. In the best case for ω', assume two jobs are in the system under both policies with identical residual service times. From here the policies clear the system at the same time.

(c) different jobs are being served under the two policies (i.e. ω has completed more jobs than ω'); ω can be left with two jobs in the system sooner than ω'. Therefore it can never be the case that ω' clears the system sooner than ω.

If time t_0 is reached after there are no more jobs to admit, the systems are identical at this point. Employing Lemmas 3.1 and 3.2, we know that the *ahead* property holds and therefore ω' will not complete jobs sooner than ω.

(Corollary 3.1) □

Corollary 3.2: *Theorems 3.1 and 3.2 and Corollary 3.1 hold for arbitrary numbers of dedicated and flexible servers.*

Proof: The required extensions are as follows:

- Number of dedicated servers ($N = 2, F = 1$): the case with many dedicated servers is a generalization of the case with one dedicated server at each station.

The generalization is as follows: a station including blocked servers is considered a blocked station; a station including only busy servers is considered a busy station; and a station including free servers is considered a free station. For example if the first station is blocked and the flexible server is at the first station, hand-off is possible.

When a station has more than one dedicated server, extending the proofs above becomes more complicated. With only one dedicated server at each station, we had to compare at most three different residual service times. With more dedicated servers, more comparisons are required. Moreover, the order in which the dedicated servers in a station should be considered (for comparison), becomes an issue. In other words, when the system is compared under two different policies, we need to determine the jobs that should be compared.

However, these additional complexities become conceptually easier to manage with the following considerations. Note that jobs are indistinguishable and service times depend only on the station. For example, if more than one job is blocked in the first station, it does not matter which one is chosen for departure (they are indistinguishable). However in order to be able to adapt the proofs, without loss of generality, we assume that both policies choose blocked servers in the same order, as the proofs try to compare the departure times or residual service times of the same jobs (comparing different jobs in the proofs will lead to the same results with the drawback that it would make it far harder for a reader to follow the proof).

In all of the proofs above if the first station is not blocked, consider the two dedicated servers in each of the stations with the earliest completion times. If

the first station is blocked, one of the blocked servers and the dedicated server with the earliest completion time in the second stations are considered. Note that the dedicated server with the earliest completion time in the second station is a free dedicated server, if the second station is free.

- Number of flexible servers ($N = 2, F > 1$): introducing more than one flexible server complicates the proofs. For example, if a station contains more than one flexible server, in which order should they be picked for comparison? Or as another example, is it required to compare the residual service times of flexible servers? Also, does the assignment of the flexible servers affect the proofs?

  It turns out that in all of the above proofs, we need to consider only one flexible server at a time. Note that flexible servers do not affect each other in the proof scenarios, i.e. each case of the proof involves only one flexible server and one or two dedicated servers. Assume a first flexible server is assigned to a job. When assigning a second flexible server, the only difference with $F = 1$ is that one of the busy servers is a flexible server. However when analysing the behavior of the second flexible server, we ignore the first flexible server and only consider dedicated servers to perform allocations.

  The fact that flexible servers become idle once they complete a job at the second station or hand off to a free dedicated server, makes it possible to treat the flexible servers independently. Note that here we only consider the correctness of the stated properties and do not specify the optimal policy. An optimal policy's decision on allocation of a flexible server will depend on the positions of other flexible servers.

(Corollary 3.2) □

In this section we proved that an optimal policy performs hand-off and does not idle. In the next section, we model tandem lines with two stations and determine the optimal allocation policy. The properties proven in Section 3.2 will be used to reduce the size of the action and state spaces.

3.3 Markov Decision Process Model

We use a Markov Decision Process (MDP) to model the problem above. MDP models are a mathematical framework for modelling situations where decisions need to be made in the presence of randomness. More specifically, an MDP is a stochastic process on the random variables of state, action, and reward. An MDP can be employed, if a process has the Markov property, i.e. the conditional probability distribution of future states of the process depends only upon the present state and the action chosen. For our controlled continuous-time Markov chain (CTMC), let S, A, and A_s represent the state space, the action space, and the action space conditioned on the Markov chain being in state $s \in S$, respectively.

Feldman and Valdez-Flores (1996) define an MDP as follows: let Z be a system description process with state space S and let D be a decision process with action space A. The process (Z, D) is a *Markov Decision Process*, if for $j \in S$, $d_n \in D$, and $n = 0, 1, \ldots$, the following holds:

$$Pr\{Z_{n+1} = j | Z_0, d_0, \ldots, Z_n, d_n\} = Pr\{Z_{n+1} = j | Z_n, d_n\}.$$

Furthermore, for each $a \in A$, let P_a be a Markov matrix (the transition matrix). Then

$$Pr\{Z_{n+1} = j | Z_n = s, d_n(s) = a\} = P_a(s, j)$$

and a reward $r(s, a)$ is gained whenever $Z_n = s$ and $d_n(s) = a$.

Puterman (2005) defines Policy Iteration (PI), or approximation in policy space, as a method for solving infinite-horizon Markov decision problems. The Policy Iteration

algorithm for Markov decision problems is as follows (for examples, look at Sections 3.3.1 and 3.3.2 below):

1. Set $n = 0$, and select an arbitrary decision rule $d_0 \in D$.
2. (Policy valuation) Solve

$$r_{d_n} - g_n e + (P_{d_n} - I_m)h_n = 0,$$

   to obtain h_n subject to $h_n(1, 0, 0) = 0$ with respect to g_n and h_n. Note that P_{d_n} is the transition matrix with actions chosen according to d_n, e is a column vector of ones, and I_m is the m-dimensional identity matrix, where $m = |S|$.
3. (Policy improvement) Choose d_{n+1} to satisfy

$$d_{n+1}(s) = \text{argmax}_{a \in A_s}\{r(s, a) + \sum_{j \in S} p(j|s, a)h_n(j)\}, \forall s \in S,$$

   setting $d_{n+1} = d_n$ if possible.
4. If $d_{n+1} = d_n$, stop and announce d_n as the optimal policy. Otherwise increment n by 1 and return to step 2.

For our specific problem, a state $s \in S$ denotes a tuple

$$s = (x_1, y_1, \ \ldots, \ x_i, y_i, \ \ldots, \ x_N) \tag{3.1}$$

where x_i is the number of busy servers in station i and y_i is the number of blocked servers in station i. Note that we do not need to include y_N in the state, as no servers can be blocked at the last station. We uniformize the CTMC (see Lippman (1975)) to convert the continuous time problem to discrete time. The normalization constant that we will employ is denoted by q, and is defined below.

Constructing the transition probability matrices is a two stage process. One needs to start from the initial state (s) and follow possible actions (a) to determine the new states (s'). s' is an intermediate state which does not appear in the transition probability matrix. The transition between s and s' is immediate. From there, it is possible to follow further transitions and reach new states (s'') with the probabilities defined in the transition matrix (P_a). We have:

$$s \xrightarrow{a} s' \xrightarrow{P_a} s''$$

that is reflected in the transition matrix as $P_a(s, s'') = \frac{\mu}{q}$ where $q = \max_{s \in S} \sum_{s'' \in \{S-s\}} P_a(s, s'')$ and μ is the transition rate from s' to s''.

The size of the action space is $|A| = \binom{F+N-1}{N-1}$. An action $a \in A$ is denoted by a tuple

$$a = (a_1, \ldots, a_j, \ldots, a_N)$$

where $0 \leq a_j \leq F$ and $\sum_{j=1}^{N} a_j = F$. a_j is the number of flexible servers assigned to station j.

3.3.1 Generic MDP

We begin by specializing to the case of $N = 2, F = 1$, I dedicated servers at the first station, and J dedicated servers at the second station. We intend to apply the Policy Iteration algorithm to find the optimal policy.

States:

Using the state description in (3.1), the state space is as follows:

$S =$
$\{(I+1,0,j), 0 \leq j < J,$
$(I+1,0,J),$
$(I,0,J+1),$
$(i,b,J+1), i,b>0, i+b=I,$
$(i,b,J), i \geq 0, b>0, i+b=I,$
$(0,I,J+1)\}$

Our state space is constrained so that hand-off is used whenever possible (as we showed in Theorem 3.1 that an optimal policy performs hand-off). If we made it optional to employ hand-off, additional states (such as $(I,1,J)$ and $(i,b,J), i,b > 0, i+b=I+1$) would need to be introduced.

Actions:

The action space is as follows:

$$A = \{a_1, a_2\}$$

where a_i means moving the flexible server to the i^{th} station (which is different from the previous page notations). The permissible actions are given by

$$A_s = \begin{cases} \{a_1, a_2\} & \text{for } s = (i,b,J) \\ \{a_1\} & \text{for } s = (I+1,0,j), (I+1,0,J) \\ \{a_2\} & \text{for } s = (I,0,J+1), (i,b,J+1), (0,I,J+1) \end{cases}$$

where $0 < j < J$ and $i+b=I$.

In the Policy Iteration algorithm, we choose the initial policy (d_0) to be:

$$d_0(s) = \begin{cases} a_1 & \text{for } s = (I+1,0,j), (I+1,0,J) \\ a_2 & \text{for } s = (i,b,J), (I,0,J+1), (i,b,J+1), (0,I,J+1), \end{cases}$$

where i, j, and b are constrained so that s is in the state space.

The reward gained from d_0 is:

$$r_{d_0}(s) = \frac{1}{q}\begin{cases} j\mu_2 & \text{for } s = (I+1, 0, j) \\ J\mu_2 & \text{for } s = (I+1, 0, J) \\ (J+1)\mu_2 & \text{for } s = (i, b, J), (I, 0, J+1), (i, b, J+1), (0, I, J+1). \end{cases}$$

The reward function as stated here is a valid choice for maximizing the throughput, as shown in Section 3 of (Andradóttir et al., 2001).

Transition Probabilities:

The transition probabilities filtered by each of the actions and by the initial policy are as follows:

$$P_{a_2}(s, s'') = \frac{1}{q}\begin{cases} I\mu_1 & \text{for } s = (I, 0, J+1), s'' = (I-1, 1, J+1) \\ (J+1)\mu_2 & \text{for } s = (I, 0, J+1), s'' = (I+1, 0, J) \\ q - I\mu_1 - (J+1)\mu_2 & \text{for } s = s'' = (I, 0, J+1) \\ i\mu_1 & \text{for } s = (i, b, J+1), s'' = (i-1, b+1, J+1), i, b > 0, \\ & i + b = I \\ (J+1)\mu_2 & \text{for } s = (i, b, J+1), s'' = (i, b, J), i, b > 0, i + b = I \\ q - i\mu_1 - (J+1)\mu_2 & \text{for } s = s'' = (i, b, J+1), i, b > 0, i + b = I \\ (i+1)\mu_1 & \text{for } s = (i, b, J), s'' = (i, b, J+1), i \geq 0, b > 0, i + b = I \\ (J+1)\mu_2 & \text{for } s = (i, b, J), s'' = (i+1, b-1, J), i \geq 0, b > 0, \\ & i + b = I \\ q - (i+1)\mu_1 - (J+1)\mu_2 & \text{for } s = s'' = (i, b, J), i \geq 0, b > 0, i + b = I \\ (J+1)\mu_2 & \text{for } s = (0, I, J+1), s'' = (0, I, J) \\ q - (J+1)\mu_2 & \text{for } s = s'' = (0, I, J+1) \end{cases}$$

$$P_{a_1}(s,s'') = \frac{1}{q}\begin{cases}(I+1)\mu_1 & \text{for } s=(I+1,0,j), s''=(I+1,0,j+1), 0\le j<J\\ j\mu_2 & \text{for } s=(I+1,0,j), s''=(I+1,0,j-1), 0<j<J\\ q-(I+1)\mu_1-j\mu_2 & \text{for } s=s''=(I+1,0,j), 0\le j<J\\ (I+1)\mu_1 & \text{for } s=(I+1,0,J), s''=(I,0,J+1)\\ J\mu_2 & \text{for } s=(I+1,0,J), s''=(I+1,0,J-1)\\ q-(I+1)\mu_1-J\mu_2 & \text{for } s=s''=(I+1,0,J)\\ (i+1)\mu_1 & \text{for } s=(i,b,J), s''=(i,b,J+1), i\ge 0, b>0, i+b=I\\ (J+1)\mu_2 & \text{for } s=(i,b,J), s''=(i+1,b-1,J), i\ge 0, b>0,\\ & i+b=I\\ q-(i+1)\mu_1-(J+1)\mu_2 & \text{for } s=s''=(i,b,J), i\ge 0, b>0, i+b=I\end{cases}$$

$$P_{d_0}(s,s'') = \frac{1}{q}\begin{cases}(I+1)\mu_1 & \text{for } s=(I+1,0,j), s''=(I+1,0,j+1), 0\le j<J\\ j\mu_2 & \text{for } s=(I+1,0,j), s''=(I+1,0,j-1), 0<j<J\\ q-(I+1)\mu_1-j\mu_2 & \text{for } s=s''=(I+1,0,j), 0\le j<J\\ (I+1)\mu_1 & \text{for } s=(I+1,0,J), s''=(I,0,J+1)\\ J\mu_2 & \text{for } s=(I+1,0,J), s''=(I+1,0,J-1)\\ q-(I+1)\mu_1-J\mu_2 & \text{for } s=s''=(I+1,0,J)\\ I\mu_1 & \text{for } s=(I,0,J+1), s''=(I-1,1,J+1)\\ (J+1)\mu_2 & \text{for } s=(I,0,J+1), s''=(I+1,0,J)\\ q-I\mu_1-(J+1)\mu_2 & \text{for } s=s''=(I,0,J+1)\\ i\mu_1 & \text{for } s=(i,b,J+1), s''=(i-1,b+1,J+1), i,b>0,\\ & i+b=I\\ (J+1)\mu_2 & \text{for } s=(i,b,J+1), s''=(i,b,J), i,b>0, i+b=I\\ q-i\mu_1-(J+1)\mu_2 & \text{for } s=s''=(i,b,J+1), i,b>0, i+b=I\\ (i+1)\mu_1 & \text{for } s=(i,b,J), s''=(i,b,J+1), i\ge 0, b>0,\\ & i+b=I\\ (J+1)\mu_2 & \text{for } s=(i,b,J), s''=(i+1,b-1,J), i\ge 0, b>0,\\ & i+b=I\\ q-(i+1)\mu_1-(J+1)\mu_2 & \text{for } s=s''=(i,b,J), i\ge 0, b>0, i+b=I\\ (J+1)\mu_2 & \text{for } s=(0,I,J+1), s''=(0,I,J)\\ q-(J+1)\mu_2 & \text{for } s=s''=(0,I,J+1)\end{cases}$$

where q is the normalization factor defined as $q = I\mu_1 + J\mu_2 + max(\mu_1, \mu_2)$.

To prove the optimality of d_0, we need to show $d_1(s) = d_0(s)$, where:

$$d_1(s) = \text{argmax}_{a \in A_s}\{r(s,a) + \sum_{j \in S} p(j|s,a)h_0(j)\}, \forall s \in S,$$

is the result of one iteration of the Policy Iteration algorithm. This is equivalent to showing that for all s

$$r(s,a) + \sum_{j \in S} p(j|s,a)h_0(j) - \left(r(s,d_0(s)) + \sum_{j \in S} p(j|s,d_0(s))h_0(j) \right) \leq 0. \qquad (3.2)$$

In inequality (3.2) given that $s \xrightarrow{a} s' \xrightarrow{P_a} s''$, $r(s,a) = x'_N \mu_N$ where x'_N appears in the representation of state s' according to (3.1) and $\frac{\mu_N}{q} = P_a(s,s'')$. Note that s' is also a function of s and a.

We implemented the Policy Iteration algorithm in MATLAB and intended to apply it to the stated generic MDP. However it is not an easy task to model the generic MDP in MATLAB as each of the state space members (in S) represents a set of states (as b, i, and j take a range of values). This fact becomes problematic when matrices are to be constructed. We have not been able to perform Policy Iteration for the generic MDP. Therefore we observed several configurations with $N = 2, F = 1$ and specific values for the numbers of dedicated servers. For example we set (I, J) to $\{(1,1), (1,2), (2,3), (4,2), (2,4), (3,3), (3,4), (7,3)\}$.

The result of the Policy Iteration algorithm (inequality (3.2)) is an expression in terms of service rates. We tried to verify the inequality for generic cases using induction on I, J. We have not been able to find such a relation to this point. However we could prove that the expression holds for arbitrary service rates, for every specific I, J value that we considered.

Conjecture 3.1. The optimal policy resulting from these observations is the policy that *clears blocking whenever possible and otherwise assigns the flexible server to the first station.*

Comparing this policy with allocation policies for collaborative servers, Andradóttir et al. (2007) (Lemma 4.1) show that for a tandem queue with two stations, a finite buffer, a dedicated server at each station, and a flexible server, the optimal policy is threshold type. The optimal policy assigns the flexible server based on the number of jobs in the system (both in service and in the buffer). The optimal policy sends the flexible server to the second station, when there is a busy dedicated server at each station (even if there are no blocked servers). This difference with our optimal policy occurs due to the server collaboration assumption in Andradóttir et al. (2007).

3.3.2 Sample MDP

In order to show how we applied the Policy Iteration algorithm, in the following we give details for a specific example with $I = 2, J = 3, F = 1$. To be able to construct the discrete-time MDP, we choose the following normalization factor: $q = 2\mu_1 + 3\mu_2 + max(\mu_1, \mu_2)$. Here,

$$A = \{a_1, a_2\}$$

$S = \{(3,0,3),(3,0,2),(3,0,1),(3,0,0),(2,1,3),(2,0,3),(1,2,3),(1,1,4),$
$(1,1,3),(0,2,4),(0,2,3)\}$

$$A_s = \begin{cases} \{a_1, a_2\} & \text{for } s = (2,1,3),(1,2,3),(1,1,3),(0,2,3) \\ \{a_1\} & \text{for } s = (3,0,0),(3,0,1),(3,0,2),(3,0,3),(2,0,3) \\ \{a_2\} & \text{for } s = (1,1,4),(0,2,4) \end{cases}$$

Our candidate optimal policy d_0 is:

$$d_0(s) = \begin{cases} a_1 & \text{for } s = (3,0,0),(3,0,1),(3,0,2),(3,0,3),(2,0,3) \\ a_2 & \text{for } s = (2,1,3),(1,2,3),(1,1,4),(1,1,3),(0,2,4),(0,2,3) \end{cases}$$

$$r_{d_0}(s) = \begin{cases} 4\mu_2 & \text{for } s = (2,1,3),(1,2,3),(1,1,4),(1,1,3),(0,2,4),(0,2,3) \\ 3\mu_2 & \text{for } s = (3,0,3),(2,0,3) \\ 2\mu_2 & \text{for } s = (3,0,2) \\ \mu_2 & \text{for } s = (3,0,1) \\ 0 & \text{for } s = (3,0,0) \end{cases}$$

For the following matrices, assume the following state ordering: (3, 0, 3), (3, 0, 2), (3, 0, 1), (3, 0, 0), (2, 0, 3), (1, 2, 3), (1, 1, 4), (1, 1, 3), (0, 2, 4), (0, 2, 3).

$$P_{a_1}(s,s'') = \frac{1}{q} \begin{pmatrix} q_{1,a_1} & 3\mu_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3\mu_1 & q_{2,a_1} & 2\mu_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3\mu_1 & q_{3,a_1} & \mu_2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3\mu_1 & q_{4,a_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 3\mu_2 & 0 & 0 & 0 & q_{5,a_1} & 0 & 0 & 0 & 0 & 0 & 2\mu_1 \\ 0 & 3\mu_2 & 0 & 0 & 3\mu_1 & q_{6,a_1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3\mu_2 & 0 & q_{7,a_1} & 0 & 0 & \mu_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & q_{8,a_1} & 0 & 0 & 0 \\ 3\mu_2 & 0 & 0 & 0 & 0 & 0 & 2\mu_1 & 0 & q_{9,a_1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & q_{10,a_1} & 0 \\ 0 & 0 & 0 & 0 & 3\mu_2 & 0 & 0 & 0 & 0 & \mu_1 & q_{11,a_1} \end{pmatrix}$$

$$P_{a_2}(s,s'') = \frac{1}{q}\begin{pmatrix} q_{1,a_2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & q_{2,a_2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & q_{3,a_2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & q_{4,a_2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & q_{5,a_2} & 4\mu_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & q_{6,a_2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & q_{7,a_2} & 0 & 4\mu_2 & \mu_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & q_{8,a_2} & 4\mu_2 & \mu_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4\mu_2 & 0 & 2\mu_1 & q_{9,a_2} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & q_{10,a_2} & 4\mu_2 \\ 0 & 0 & 0 & 0 & 3\mu_2 & 0 & 0 & 0 & 4\mu_2 & \mu_1 & q_{11,a_2} \end{pmatrix}$$

where $q_{i,a} = q - \sum_{j=1 \wedge j \neq i}^{|S|} P_a(i,j)$ and $a \in A$.

Now we want to show inequality (3.2) to verify that $d_0(s)$ is optimal. Assume $\mu_1 > \mu_2$. Then $q = 3(\mu_1 + \mu_2)$. Let

$$C = 27\mu_1^6 + 108\mu_1^5\mu_2 + 216\mu_1^4\mu_2^2 + 288\mu_1^3\mu_2^3 + 288\mu_1^2\mu_2^4 + 192\mu_1\mu_2^5 + 64\mu_2^6.$$

Then

$$g_0 = \frac{12(9\mu_1^6\mu_2 + 36\mu_1^5\mu_2^2 + 72\mu_1^4\mu_2^3 + 72\mu_1^3\mu_2^4 + 48\mu_1^2\mu_2^5 + 16\mu_1\mu_2^6)}{C}.$$

For example for $s = (2,0,3)$,

$$h_0(s) = \frac{q(9\mu_1^5\mu_2 + 54\mu_1^4\mu_2^2 + 162\mu_1^3\mu_2^3 + 224\mu_1^2\mu_2^4 + 176\mu_1\mu_2^5 + 64\mu_2^6)}{C}$$

and inequality (3.2) simplifies to checking

$$16\mu_2^6\mu_1 + 80\mu_2^5\mu_1^2 + 190\mu_2^4\mu_1^3 + 288\mu_2^3\mu_1^4 + 261\mu_2^2\mu_1^5 + 126\mu_2\mu_1^6 + 27\mu_1^7 > 0$$

which is trivially true. If $\mu_1 < \mu_2$, with similar calculations inequality (3.2) simplifies to verifying

$$32\mu_2^6\mu_1 + 144\mu_2^5\mu_1^2 + 316\mu_2^4\mu_1^3 + 450\mu_2^3\mu_1^4 + 360\mu_2^2\mu_1^5 + 153\mu_2\mu_1^6 + 27\mu_1^7 > 0,$$

which is also trivially true. Similarly it can be shown that inequality (3.2) holds for all other states. In this example d_0 makes the Policy Iteration algorithm converge in one step. Hence d_0 is the optimal policy. It should be noted that the policy is independent of μ_1 and μ_2.

3.4 Larger systems

With two stations, only one station can be blocked. As we saw in Section 3.3, the primary task of the optimal policy is to clear blocking. When the number of stations is increased, blocking can occur in more than one station, and policies should prioritize the order in which blocked servers are cleared. Also, starvation might occur in the system. Starvation occurs when a dedicated server becomes free and has no jobs to process. To study these effects, in this section we use an MDP model and the Policy Iteration algorithm for longer tandem lines, and use these results to provide heuristics. Similar to Section 3.3, the policies that we consider are non-idling and perform hand-off whenever possible (which are properties of optimal policies as shown by Theorems 3.1 and 3.2).

To begin, we study tandem lines where stations have equal mean service times, have a dedicated server per station, and have one flexible server. Solving the MDP for 3, 4, and 5 stations suggests the following near-optimal policy: *i) clear blocking from the end to the beginning unless it causes starvation; ii) if clearing a blocked server would cause starvation, admit a new job.*

With 3 stations, allocation of the flexible server does not cause starvation in any of the states. With 4 stations the following states cause starvation: (1,0, 1,0, 0,1, 1), (1,0, 0,0, 0,1, 1), (0,1, 1,0, 0,1, 1). In these states an optimal policy admits a new job. However while the following states also cause starvation, clearing a blocked server is preferred: (1,0, 0,1, 1,0, 1) and (1,0, 0,1, 0,1, 1).

For 5 stations the suggested policy offers the optimal allocations in most states. The number of states is 236 from which the suggested policy results in the optimal allocation in 228 of them. With 5 stations, the following states cause starvation: (1,0,

1,0, 0,0, 0,1, 1), (1,0, 0,0, 1,0, 0,1, 1), (1,0, 0,0, 0,1, 1,1, 1), (1,0, 0,0, 0,1, 1,0, 1), (1,0, 0,0, 0,0, 0,1, 1), (0,1, 1,0, 1,0, 0,1, 1), (0,1, 1,0, 0,1, 1,0, 1), (0,1, 1,0, 0,0, 1,0, 2), (0,1, 0,1, 1,0, 0,1, 1). In all these states avoiding starvation is favoured over clearing blocking. However, while the following states cause starvation, clearing blocking is preferred: (1,0, 1,0, 1,0, 0,1, 1), (1,0, 1,0, 0,1, 1,0, 1), (1,0, 0,1, 1,0, 1,0, 1), (1,0, 0,1, 1,0, 1,0, 0), (1,0, 0,1, 1,0, 0,0, 1).

Also, in the following three states blocking is not cleared and starvation does not occur: (1,0, 1,0, 1,0, 1,1, 1), (1,0, 0,0, 0,1, 1,1, 1), (0,1, 0,1, 0,1, 0,1, 1). The exceptions above suggest that while an optimal policy does not have a simple structure, a near-optimal policy exists as follows: *i) clear blocking from the end to the beginning unless it causes starvation; ii) if clearing a blocked server would cause starvation, admit a new job.*

Note that as N increases, the number of states increases exponentially, which makes the overhead of constructing the MDP prohibitive. For example when $N = 5$ (with the configuration introduced above), there are 5 actions, 236 states, and P_{a_1}, P_{a_2}, P_{a_3}, P_{a_4}, and P_{a_5} have 498, 496, 462, 447, and 413 non-zero entries, respectively. Calculating each of these entries is difficult to automate, and hence is error-prone. In Section 3.5 we will use simulation to show that if the policy stated in the beginning of this section is employed, near-optimal results are gained for different configurations.

3.5 Numerical results

A number of policies and configurations are considered here. Let s_i represent the number of dedicated servers at station i. The first set of configurations (Tables 3.1, 3.2, and 3.3) has $s_i = 1, \mu_i = 1, i = 1, \ldots, N$, and $F = 1$. N ranges from 4 to 30. The second set of configurations (Tables 3.4, 3.5, and 3.6) deals with heterogeneous service rates and a single server per station. We let $w = (w_1, \ldots, w_i, \ldots, w_N)$ be the vector of mean service times, where w_i is the mean service time at the i^{th} station (as defined in Section 2.1).

The policies we consider can be distinguished by the way they treat blocking and starvation and how they employ hand-off. A policy should determine the order in which it clears blocking. It should specify how it deals with the trade-off between blocking and starvation. The policy also describes if it uses hand-off or not. While Theorem 3.1 suggests that hand-off should be used, here we also consider policies that do not employ hand-off. We intend to show how the suggested heuristic works and also study the impact of allowing hand-offs.

In all policies if there is no blocking in the system and free flexible servers exist, flexible servers are sent to the first station to admit a new job. This is consistent with the necessity of non-idling for optimality.

The policies we studied are as follows:

- Policy 1: clears blocking from end to beginning only if it does not cause starvation in the $\lfloor \frac{2}{3}N \rfloor$ previous stations (derived experimentally); uses hand-off
- Policy 2: ignores blocking and admits new jobs when possible; uses hand-off

- Policy 3: clears blocking from end to beginning only if it does not cause starvation; uses hand-off
- Policy 4: clears blocking in order of descending service rates only if it does not cause starvation; uses hand-off
- Policy 5: clears blocking in order of descending service rates; uses hand-off
- Policy 6: clears blocking from beginning to end; uses hand-off
- Policy 7: clears blocking from end to beginning; uses hand-off
- Policy 8: clears blocking from beginning to end; does not use hand-off

Policies 4 and 5 prioritize clearing blocking caused by the station with the highest mean service time (among the stations that cause blocking). Policies 4 and 5 are applied only on the second set of configurations. In practice, Policies 2, 3, and 4 show similar behavior as Policies 3 and 4 clear a blocked station only if all stations preceding a blocked station are blocked. Hand-off in Policy 2 pushes a flexible server through all consecutive blocked stations. Simulation results reveal that Policies 1 and 4 perform better than the other policies, depending on the configuration considered. In Policy 1, the $\lfloor \frac{2}{3}N \rfloor$ parameter gives higher priority to avoiding starvation, compared to Policies 2, 3, and 4. This parameter is derived experimentally and is used to ignore a blocked server, if clearing that blocking causes simultaneous starvations in a large number of stations.

We applied the above policies to several configurations (defined below) and measured throughput in a simulation environment. Each simulation is a long run, in terms of the number of departures from the system. The run is long enough (100 million

departures) so that the effect of starting the system from an empty state becomes negligible. In the following tables policies are ordered based on their throughputs.

3.5.1 Homogeneous mean service times (the first set)

Table 3.1 compares the throughputs of configurations with one flexible server, $N \in \{4, 5, 8, 15, 30\}$ stations where the service time of each station is exponentially distributed with rate one, and one dedicated server at each station. Policy 1 is close to what the Policy Iteration algorithm suggested and results in the best throughput among these policies. It can be observed that Policies 1 and 2 lead to identical throughputs. This is because when a station and all its preceding stations up to the first station are blocked, Policy 1 clears blocking from the last station in the sequence of blocked stations and Policy 2 uses hand-off to pass the flexible server to the last station in the sequence of blocked stations. Although the throughputs are identical, Policy 2 needs to perform extra hand-offs compared to Policy 1.

N	**Policy1**	**Policy2**	**Policy3**	**Policy6**	**Policy7**	**Policy8**	$F = 0$
4	0.93248	0.93248	0.93065	0.92021	0.91869	0.78592	0.59596
5	0.83049	0.83049	0.82448	0.81400	0.81032	0.66214	0.53038
8	0.66720	0.66720	0.64631	0.64298	0.62598	0.47267	0.41827
15	0.51520	0.51520	0.47234	0.50238	0.44993	0.33768	0.32384
30	0.40490	0.40490	0.34292	0.40149	0.32791	0.26023	0.25696

Table 3.1: Throughput for configurations with $i = 1, \ldots, N, w_i = 1$, and $F = 1$

Comparing Policies 1 and 3, Policy 1 allows starvation in the initial stations, while Policy 3 avoids it in all stations. Policy 1 has better performance than Policy 3. The reason is that the downside of having starvation in the initial stations is less than the benefit of clearing blocked servers at the end stations. There are two explanations: i) starvation closer to the first station is likely to be resolved more

quickly compared to starvation in the end stations; ii) starvation at the initial stations, when the downstream stations are busy, does not affect the downstream stations. Hence in Policy 1, the trade-off between clearing blocking and avoiding starvation (in the initial stations) is resolved in favour of clearing blocking. Finally, notice that Policy 8 is the only policy which does not perform hand-off. Table 3.1 shows that Policy 8 provides the least throughput values among all the policies stated here. Also, while all of the other policies have relatively close throughput values, Policy 8 results in far smaller throughput values.

To illustrate the effect of including flexible servers, in the last column of Table 3.1 we show the case where there are no flexible servers and the extra server is assigned to one of the middle stations. It can be observed that making a server flexible can lead to throughput improvements up to 36.08%.

We also study configurations that include service time distributions with coefficients of variation other than 1. In simulations, an Erlang distribution is used for coefficients of variation less than one and a Hyper-exponential distribution for coefficients of variation greater than one. Table 3.2 is similar to Table 3.1 except that the coefficients of variation of all stations are 0.5. Table 3.3 is similar to Table 3.1 except that the coefficients of variation are 1.34. The results in both cases are almost the same as the case with coefficients of variation equal to 1 for all stations.

Let cv_i be the coefficient of variation of the i^{th} station and cv be the vector of coefficients of variation. We simulated a number of configurations with different coefficients of variation and concluded the same results as the case with coefficients of variation equal to 1 for all stations. More specifically we tested the following configurations: $N = 4, cv_3 = 0.5, 1.34$; $N = 5, cv_2 = 0.5$; $N = 5, cv_4 = 1.34$; $N =$

$8, cv_3 = 0.5, cv_7 = 1.34;\ N = 15, cv_3 = cv_{10} = 0.5, cv_7 = cv_{13} = 1.34;\ N = 30, cv_3 = cv_{10} = cv_{18} = cv_{25} = 0.5, cv_7 = cv_{13} = cv_{22} = cv_{28} = 1.34$. The unspecified coefficients of variation are equal to one.

N	Policy1	Policy2	Policy3	Policy6	Policy7	Policy8
4	0.99498	0.99498	0.99257	0.98428	0.98265	0.81297
5	0.90918	0.90918	0.90360	0.89475	0.84575	0.73190
8	0.77088	0.77088	0.75612	0.75361	0.73646	0.60107
15	0.64377	0.64377	0.61052	0.63681	0.58832	0.50407
30	0.54746	0.54746	0.50081	0.54584	0.48228	0.43911

Table 3.2: Throughput for configurations with $i = 1, \ldots, N, w_i = 1, cv_i = 0.5$, and $F = 1$

N	Policy1	Policy2	Policy3	Policy6	Policy7	Policy8
4	0.94136	0.94136	0.93919	0.92778	0.93919	0.79680
5	0.83372	0.83372	0.82736	0.81564	0.81244	0.66997
8	0.65759	0.65759	0.63753	0.63090	0.61551	0.46233
15	0.49317	0.49317	0.44717	0.47511	0.42999	0.29728
30	0.37423	0.37423	0.30740	0.36776	0.29448	0.19957

Table 3.3: Throughput for configurations with $i = 1, \ldots, N, w_i = 1, cv_i = 1.34$, and $F = 1$

3.5.2 Heterogeneous mean service times (the second set)

Table 3.4 compares the throughputs for configurations with various numbers of stations, different service rates, and one flexible server. Policies 4 and 5 are not applicable to the first set of configurations, but they are compared against other policies in the second set of configurations. Table 3.4 illustrates that the ordering of policies remains the same as Table 3.1, except that Policy 4 outperforms Policy 1 in certain cases. The intuition is that while for small systems, an optimal policy clears blocking

from the end to the beginning, for large systems an optimal policy prefers to clear blocking for bottleneck stations first (i.e. stations with higher mean service times). For small systems, several adjacent blocked stations could deny admissions to the system. Therefore it is valuable to clear blocking from the end as Policy 1 does.

w	**Policy4**	**Policy1**	**Policy2**	**Policy5**	**Policy7**
(1,3,6,9,5,4,2,3,7,10,5,2,1)	0.10409	0.10409	0.10409	0.09940	0.09676
(1,3,2,4,5,7,10,8,6,4,5,2,1)	0.10286	0.10286	0.10286	0.09769	0.09635
(7,5,6,4,2,3,1,3,2,4,5,9,8)	0.10876	0.10872	0.10872	0.10448	0.10472
(5,2,14,3,7,12,6,1,10)	0.08403	0.08402	0.08402	0.08088	0.07996
(9,8,7,6,5,4,3,2,1)	0.11143	0.11143	0.11143	0.10508	0.10319

Table 3.4: Throughput for configurations with $F = 1$ and $i = 1, \ldots, N$, $cv_i = 1$

Table 3.5 is similar to Table 3.4 except that the coefficients of variation of all service times are 0.5. The results are almost the same as when the coefficients of variation are equal to 1. Table 3.6 is similar to Table 3.4 except that the coefficients of variation are greater than 1 for all stations (with different values). For $w = (1, 3, 6, 9, 5, 4, 2, 3, 7, 10, 5, 2,\ 1)$, $w = (1, 3, 2, 4, 5, 7, 10, 8, 6, 4, 5, 2, 1)$, and $w = (7, 5, 6, 4, 2, 3, 1, 3, 2, 4, 5, 9, 8)$ mean service time vectors, coefficients of variation vectors are $cv = (2.38, 2.31, 1.66, 1.27, 1.83,\ 2.06, 2.63, 2.31, 1.51, 1.17, 1.83, 2.63, 2.38)$, $cv = (2.38, 2.31, 2.63, 2.06, 1.83, 1.51, 1.17, 1.38,\ 1.66, 2.06, 1.83, 2.63, 2.38)$, and $cv = (1.51, 1.83, 1.66, 2.06, 2.63, 2.31, 2.38, 2.31, 2.63, 2.06, 1.83, 1.27, 1.38)$, respectively. Compared to the case with coefficients of variation equal to 1, Policies 1 and 2 perform better than Policy 4. The intuition is that high variability increases the chance of blocking or starvation in adjacent stations.

Tables 3.2, 3.3, and 3.5 suggest that if coefficients of variation of all stations are changed uniformly, the ordering between policies remains the same as when coefficients of variation are 1. However, Table 3.6 indicates that when coefficients of

w	**Policy4**	**Policy1**	**Policy2**	**Policy5**	**Policy7**
(1,3,6,9,5,4,2,3,7,10,5,2,1)	0.11852	0.11848	0.11848	0.11628	0.11571
(1,3,2,4,5,7,10,8,6,4,5,2,1)	0.11734	0.11734	0.11734	0.11277	0.11080
(7,5,6,4,2,3,1,3,2,4,5,9,8)	0.12611	0.12492	0.12492	0.12488	0.12494

Table 3.5: Throughput for configurations with $F = 1$ and $i = 1, \ldots, N$, $cv_i = 0.5$

w	**Policy4**	**Policy1**	**Policy2**	**Policy5**	**Policy7**
(1,3,6,9,5,4,2,3,7,10,5,2,1)	0.09882	0.09901	0.09901	0.09478	0.09609
(1,3,2,4,5,7,10,8,6,4,5,2,1)	0.09759	0.09760	0.09760	0.09307	0.09547
(7,5,6,4,2,3,1,3,2,4,5,9,8)	0.10031	0.10049	0.10049	0.09897	0.09798

Table 3.6: Throughput for configurations with $F = 1$ and $i = 1, \ldots, N$, $cv_i > 1$

variation of stations are arbitrarily modified (in this case all greater than one), the ordering between policies might change. Finally we should comment that our results on coefficients of variation are not conclusive, in the sense that we have not considered vectors of coefficients of variation (cv) with more heterogeneous values.

3.6 Increasing the number of flexible servers

In this section we consider the effect of increasing F, the number of flexible servers. Assume we are given a system with a particular allocation of dedicated servers. We start to change dedicated servers to flexible servers until all servers are flexible. In other words, F takes all values between 0 and N. An interesting question is how much throughput improves as F increases.

3.6.1 Homogeneous w

For equal workloads, we consider a tandem line with 15 stations where originally each station possesses one dedicated server. Service times at each station follow an exponential distribution with rate 1. The deployed server allocation policy is one that clears blocking from the end to the beginning of the line, unless there is blocking caused by a station with no dedicated servers. In that case, the policy prioritizes allocating servers to the station with no dedicated servers over clearing blocked servers from the end to the beginning. The policy also avoids starvation.

While increasing F, one should specify the order in which dedicated servers are changed to flexible servers. We contemplate four ordering policies: Policy i) choosing servers from the end to the beginning of the line; Policy ii) picking servers from the beginning to the end of the line; Policy iii) selecting servers around the middle station in a zig-zag way; Policy iv) choosing servers randomly. Simulations showed that Policies i and ii lead to similar results.

Figure 3.30 contrasts Policies i and iii (for the 15 stations configuration described above) and shows that throughputs are fairly close for the two policies, but slightly higher for Policy i. Policy iv performs slightly worse than the other two policies,

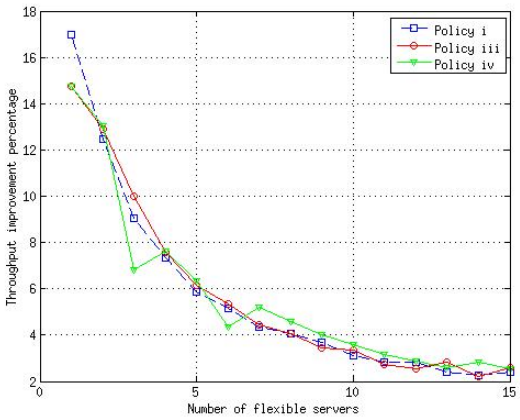

Figure 3.30: Throughput improvement (as a percentage) versus F for $w = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$

suggesting that choosing the first few servers is more sensitive to the employed policy, compared to choosing the last servers. It is worthwhile to note that, as can be seen in Figure 3.30, the highest improvements in throughput are gained from making the first few servers flexible. Also, as F increases, the throughput improvement is diminished.

3.6.2 Heterogeneous w

In addition to the equal workload case, we want to consider more generic configurations with heterogeneous service rates and multiple servers per station. We propose four ordering policies that specify the order in which dedicated servers are changed to flexible servers.

- Policy I. Changing a dedicated server of a station to a flexible server reduces the total service rate of that station and can even make the station a bottleneck. Therefore to gain better results, the order in which flexible servers are chosen should depend on the mean total service rates of stations. As we intend to

avoid introducing bottlenecks, consider the case where each station has one less dedicated server and then compare their total service rates. More specifically, start making servers flexible from the stations with the highest $\mu_i(s_i - 1)$ value. In case the values of $\mu_i(s_i - 1)$ are equal for more than one station, choose the one with the lowest μ_i first. See the work by Yarmand and Down (2012) and Chapter 2 for further details.

- Policy II. Start choosing servers from the station with the highest s_i value. In case more than one station has the same value for s_i, choose the one with the lowest μ_i.

- Policy III. Start picking servers from the station with the highest μ_i value regardless of the value of s_i.

- Policy IV. Select servers randomly.

Policy I follows a selection order that intends to avoid introducing new bottlenecks (if all s_i and μ_i values are the same, introducing bottlenecks is inevitable). On the other hand, Policy II makes use of the server multiplicity effect (see Chapter 2 and (Yarmand and Down, 2012)). This effect suggests that comparing two stations with equal total mean service times, the station with a higher number of servers leads to higher throughput. Policy II chooses the station that benefits most from this effect. Policy III is based on the fact that jobs could depart sooner from stations with higher service rates (when there is no blocking).

Figure 3.31 compares these four policies over a configuration with $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$ and an initial server allocation of (4, 5, 5, 2, 4, 2, 1, 4, 5). Figure 3.31 shows that the throughput improvement is dependent on the order that servers

are chosen to become flexible. As the server allocation policy in the simulations, we employed Policy 4 (from Section 3.5) enhanced as follows: the policy prioritizes helping a bottleneck caused by a station with no dedicated servers.

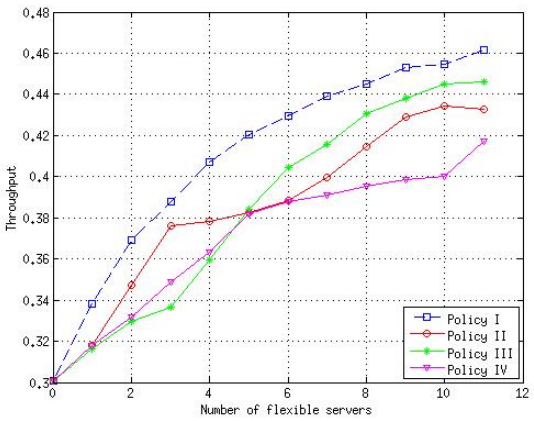

Figure 3.31: Throughput versus F for $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$

We tried additional configurations to compare the performance of the proposed policies. Figure 3.32 illustrates the comparison for $w = (9, 7, 10, 3, 5, 4, 1, 13, 12)$ with an initial server allocation of (11, 7, 12, 3, 5, 4, 1, 15, 14). Figure 3.33 presents the comparison for $w = (1, 3, 4, 5, 7, 9, 10, 12, 13)$ with an initial server allocation of (3, 5, 6, 7, 9, 11, 12, 14, 15). Figures 3.31, 3.32, and 3.33 show that Policy I performs better than Policies II and III. The figures also illustrate that the highest throughput improvements are gained when the first few servers are made flexible, and that as F increases, the throughput improvement is diminished. Additionally, the performance of Policy IV in these figures shows that the ordering policy employed affects the degree of throughput improvement (and if not chosen properly, results in lower improvements compared to when it is chosen properly). The fact that in some cases, Policy IV performs better than Policies II and III, implies that Policies II and III,

while intuitively reasonable, are not the best choices to employ.

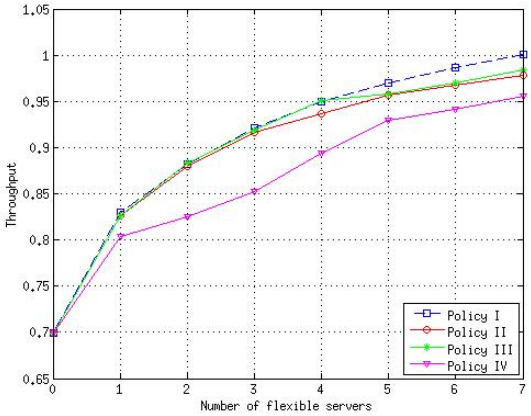

Figure 3.32: Throughput versus F for $w = (9, 7, 10, 3, 5, 4, 1, 13, 12)$

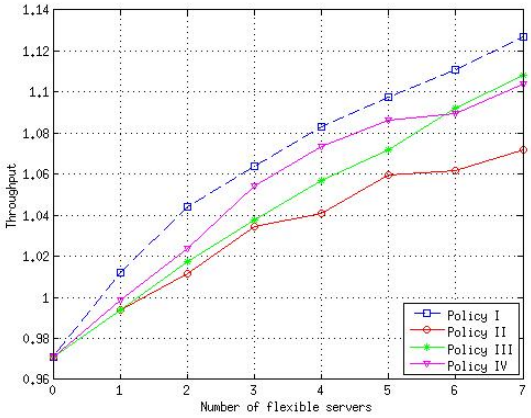

Figure 3.33: Throughput versus F for $w = (1, 3, 4, 5, 7, 9, 10, 12, 13)$

In this chapter we studied lines with flexible servers that can work at all of the stations. In the next chapter, we consider flexible servers which are constrained to be shared only between neighbouring stations. We report allocation policies for the constrained case and contrast them with the non-constrained case. We also compare the throughput results under the two constrained and non-constrained cases.

Chapter 4

Constrained Flexibility

In this chapter, we study situations where flexible servers are unable to work at all of the stations of a tandem line. We let flexible servers be constrained between two specific stations. The motivation for this study is that in practice, there might be situations where moving flexible servers among all stations is not possible (or if possible, it might be costly). Instead, servers can be shared between two adjacent stations.

With the insights gained from the previous chapter, here we only explore policies which are non-idling (assuming that hand-off is allowed). In the following, we consider tandem lines with three and four stations, in Sections 4.1 and 4.2, respectively.

4.1 Tandem lines with three stations

In this section, we study the following four cases: there is a flexible server which only moves between the first and second stations; there is a flexible server which only moves between the second and third stations; there are two flexible servers, one moving only

between the first and second stations and the other one moving only between the second and third stations; and finally there are two flexible servers that can move between all of the stations (which we call the non-constrained or fully flexible case).

4.1.1 Flexible server between first and second stations

Consider a tandem line with three stations and a dedicated server at each station. Assume a flexible server exists which can only move between the first and second stations. Theorem 4.1 describes the optimal policy.

Theorem 4.1. *The optimal policy prioritizes clearing blocking. It performs hand-off and allocates the flexible server to the first station, whenever possible. Otherwise the flexible server is assigned to the second station. The optimal policy is rate independent.*

Proof. To be able to construct the discrete-time MDP, we choose the following normalization factor: $q = 2\mu_1 + 2\mu_2 + \mu_3$. The MDP details are as follows.

$$A = \{a_1, a_2\}$$

$S = \{(2,0,1,0,1), (2,0,1,0,0), (2,0,0,1,1), (2,0,0,0,1), (2,0,0,0,0), (1,0,2,0,1),$
$(1,0,2,0,0), (1,0,1,1,1), (1,0,0,2,1), (0,1,2,0,1), (0,1,2,0,0), (0,1,1,1,1),$
$(0,1,1,0,1), (0,1,0,2,1), (0,1,0,1,1)\}$

$$A_s = \begin{cases} \{a_1, a_2\} & \text{for } s = (0,1,1,0,1), (0,1,0,1,1) \\ \{a_1\} & \text{for } s = (2,0,1,0,1), (2,0,1,0,0), (2,0,0,1,1), (2,0,0,0,1), (2,0,0,0,0) \\ \{a_2\} & \text{for } s = (1,0,2,0,1), (1,0,2,0,0), (1,0,1,1,1), (1,0,0,2,1), (0,1,2,0,1), \\ & (0,1,2,0,0), (0,1,1,1,1), (0,1,0,2,1) \end{cases}$$

Our candidate optimal policy d_0 is:

$$d_0(s) = \begin{cases} \{a_1\} & \text{for } s = (2,0,1,0,1),(2,0,1,0,0),(2,0,0,1,1),(2,0,0,0,1),(2,0,0,0,0), \\ & (0,1,1,0,1),(0,1,0,1,1) \\ \{a_2\} & \text{for } s = (1,0,2,0,1),(1,0,2,0,0),(1,0,1,1,1),(1,0,0,2,1),(0,1,2,0,1), \\ & (0,1,2,0,0),(0,1,1,1,1),(0,1,0,2,1) \end{cases}$$

$$r_{d_0}(s) = \frac{1}{q} \begin{cases} \mu_3 & \text{for } s = (2,0,1,0,1),(2,0,0,1,1),(2,0,0,0,1),(1,0,2,0,1),(1,0,1,1,1), \\ & (0,1,2,0,1),(0,1,1,1,1),(0,1,1,0,1),(0,1,0,2,1),(0,1,0,1,1),(1,0,0,2,1) \\ 0 & \text{for } s = (2,0,1,0,0),(2,0,0,0,0),(1,0,2,0,0),(0,1,2,0,0) \end{cases}$$

$$P_{a_1}(s,s') = \frac{1}{q} \begin{cases} 2\mu_1 & \text{for } s = (2,0,1,0,1), s' = (1,0,2,0,1) \\ \mu_2 & \text{for } s = (2,0,1,0,1), s' = (2,0,0,1,1) \\ \mu_3 & \text{for } s = (2,0,1,0,1), s' = (2,0,1,0,0) \\ 2\mu_1 & \text{for } s = (2,0,1,0,0), s' = (1,0,2,0,0) \\ \mu_2 & \text{for } s = (2,0,1,0,0), s' = (2,0,1,0,1) \\ 2\mu_1 & \text{for } s = (2,0,0,1,1), s' = (1,0,1,1,1) \\ \mu_3 & \text{for } s = (2,0,0,1,1), s' = (2,0,0,0,1) \\ 2\mu_1 & \text{for } s = (2,0,0,0,1), s' = (2,0,1,0,1) \\ \mu_3 & \text{for } s = (2,0,0,0,1), s' = (2,0,0,0,0) \\ 2\mu_1 & \text{for } s = (2,0,0,0,0), s' = (2,0,1,0,0) \\ \mu_1 & \text{for } s = (0,1,1,0,1), s' = (0,1,2,0,1) \\ 2\mu_2 & \text{for } s = (0,1,1,0,1), s' = (1,0,1,1,1) \\ \mu_3 & \text{for } s = (0,1,1,0,1), s' = (1,0,2,0,0) \\ \mu_1 & \text{for } s = (0,1,0,1,1), s' = (0,1,1,1,1) \\ \mu_2 & \text{for } s = (0,1,0,1,1), s' = (1,0,0,2,1) \\ \mu_3 & \text{for } s = (0,1,0,1,1), s' = (2,0,1,0,1) \end{cases}$$

$$P_{a_2}(s,s') = \frac{1}{q}\begin{cases} \mu_1 & \text{for } s=(1,0,2,0,1), s'=(0,1,2,0,1) \\ 2\mu_2 & \text{for } s=(1,0,2,0,1), s'=(1,0,1,1,1) \\ \mu_3 & \text{for } s=(1,0,2,0,1), s'=(1,0,2,0,0) \\ 2\mu_1 & \text{for } s=(1,0,2,0,0), s'=(0,1,2,0,0) \\ \mu_2 & \text{for } s=(1,0,2,0,0), s'=(2,0,1,0,1) \\ \mu_1 & \text{for } s=(1,0,1,1,1), s'=(0,1,1,1,1) \\ \mu_2 & \text{for } s=(1,0,1,1,1), s'=(1,0,0,2,1) \\ \mu_3 & \text{for } s=(1,0,1,0,1), s'=(2,0,1,0,1) \\ \mu_1 & \text{for } s=(1,0,0,2,1), s'=(0,1,0,2,1) \\ \mu_3 & \text{for } s=(1,0,0,2,1), s'=(2,0,0,1,1) \\ 2\mu_2 & \text{for } s=(0,1,2,0,1), s'=(0,1,1,1,1) \\ \mu_3 & \text{for } s=(0,1,2,0,1), s'=(0,1,2,0,0) \\ 2\mu_2 & \text{for } s=(0,1,2,0,0), s'=(0,1,1,0,1) \\ \mu_1 & \text{for } s=(0,1,1,0,1), s'=(0,1,2,0,1) \\ 2\mu_2 & \text{for } s=(0,1,1,0,1), s'=(1,0,1,1,1) \\ \mu_3 & \text{for } s=(0,1,1,0,1), s'=(1,0,2,0,0) \\ \mu_3 & \text{for } s=(0,1,0,2,1), s'=(0,1,0,1,1) \\ \mu_1 & \text{for } s=(0,1,0,1,1), s'=(0,1,1,1,1) \\ \mu_2 & \text{for } s=(0,1,0,1,1), s'=(1,0,0,2,1) \\ \mu_3 & \text{for } s=(0,1,0,1,1), s'=(2,0,1,0,1) \end{cases}$$

Now we want to show inequality (3.2) to verify that $d_0(s)$ is optimal. We only need to show this for states $(0,1,1,0,1)$ and $(0,1,0,1,1)$. However, for both it is trivial that inequality (3.2) simplifies to 0. For example for state $(0,1,0,1,1)$, inequality (3.2) simplifies to

$$\mu_1 h_0((0,1,1,1,1)) + \mu_2 h_0((1,0,0,2,1)) + \mu_3 h_0((2,0,1,0,1))$$
$$-\mu_1 h_0((0,1,1,1,1)) - \mu_2 h_0((1,0,0,2,1)) - \mu_3 h_0((2,0,1,0,1)) = 0.$$

(Theorem 4.1) □

4.1.2 Flexible server between second and third stations

Consider a tandem line with three stations and a dedicated server at each station. Assume a flexible server exists which can only move between the second and third stations. Theorem 4.2 describes the optimal policy.

Theorem 4.2. *The optimal policy prioritizes clearing blocking. It performs hand-off and allocates the flexible server to the second station, whenever possible. Otherwise the flexible server is assigned to the third station. The optimal policy is rate independent.*

Proof. To be able to construct the discrete-time MDP, we choose the following normalization factor: $q = \mu_1 + 2\mu_2 + 2\mu_3$. The MDP details are as follows.

$$A = \{a_2, a_3\}$$

$$S = \{(1,0,2,0,1),(1,0,2,0,0),(1,0,1,0,2),(1,0,1,0,1),(1,0,1,0,0),(1,0,0,1,2),(1,0,0,1,1),(1,0,0,0,2),(1,0,0,0,1),(1,0,0,0,0),(0,1,2,0,1),(0,1,2,0,0),(0,1,1,0,2),(0,1,0,1,1),(0,1,1,0,0),(0,1,0,1,2),(0,1,0,1,1)\}$$

$$A_s = \begin{cases} \{a_2, a_3\} & \text{for } s = (1,0,1,0,1),(1,0,1,0,0),(1,0,0,1,1),(0,1,1,0,1),(0,1,1,0,0),\\ & (0,1,0,1,1) \\ \{a_2\} & \text{for } s = (1,0,2,0,1),(1,0,2,0,0),(0,1,2,0,1),(0,1,2,0,0) \\ \{a_3\} & \text{for } s = (1,0,1,0,2),(1,0,0,1,2),(1,0,0,0,2),(1,0,0,0,1),(1,0,0,0,0),\\ & (0,1,1,0,2),(0,1,0,1,2) \end{cases}$$

Our candidate optimal policy d_0 is:

$$d_0(s) = \begin{cases} \{a_2\} & \text{for } s = (1,0,2,0,1),(1,0,2,0,0),(0,1,2,0,1),(0,1,2,0,0),(1,0,1,0,1),\\ & (1,0,1,0,0),(1,0,0,1,1),(0,1,1,0,1),(0,1,1,0,0),(0,1,0,1,1) \\ \{a_3\} & \text{for } s = (1,0,1,0,2),(1,0,0,1,2),(1,0,0,0,2),(1,0,0,0,1),(1,0,0,0,0),\\ & (0,1,1,0,2),(0,1,0,1,2) \end{cases}$$

$$r_{d_0}(s) = \frac{1}{q}\begin{cases} 2\mu_3 & \text{for } s = (0,1,1,0,2),(1,0,0,0,2),(1,0,0,1,2),(1,0,1,0,2),(1,0,0,1,1), \\ & (0,1,0,1,1),(0,1,0,1,2) \\ \mu_3 & \text{for } s = (1,0,2,0,1),(0,1,2,0,1),(1,0,1,0,1),(0,1,1,0,1),(1,0,0,0,1) \\ 0 & \text{for } s = (1,0,2,0,0),(0,1,2,0,0),(1,0,1,0,0),(0,1,1,0,0),(1,0,0,0,0) \end{cases}$$

$$P_{a_3}(s,s') = \frac{1}{q}\begin{cases} \mu_1 & \text{for } s = (1,0,1,0,2), s' = (0,1,1,0,2) \\ \mu_2 & \text{for } s = (1,0,1,0,2), s' = (1,0,0,1,2) \\ 2\mu_3 & \text{for } s = (1,0,1,0,2), s' = (1,0,1,0,1) \\ \mu_1 & \text{for } s = (1,0,0,1,2), s' = (0,1,0,1,2) \\ 2\mu_3 & \text{for } s = (1,0,0,1,2), s' = (1,0,0,1,1) \\ \mu_1 & \text{for } s = (1,0,0,1,1), s' = (1,0,1,0,2) \\ 2\mu_3 & \text{for } s = (1,0,0,1,1), s' = (1,0,0,0,1) \\ \mu_1 & \text{for } s = (1,0,0,0,2), s' = (1,0,1,0,2) \\ 2\mu_3 & \text{for } s = (1,0,0,0,2), s' = (1,0,0,0,1) \\ \mu_2 & \text{for } s = (0,1,1,0,2), s' = (0,1,0,1,2) \\ 2\mu_3 & \text{for } s = (0,1,1,0,2), s' = (0,1,1,0,1) \\ \mu_2 & \text{for } s = (0,1,1,0,1), s' = (1,0,1,0,2) \\ 2\mu_3 & \text{for } s = (0,1,1,0,1), s' = (0,1,1,0,0) \\ 2\mu_3 & \text{for } s = (0,1,0,1,2), s' = (0,1,0,1,1) \\ \mu_1 & \text{for } s = (0,1,0,1,1), s' = (0,1,1,0,2) \\ \mu_2 & \text{for } s = (0,1,0,1,1), s' = (1,0,0,1,2) \\ 2\mu_3 & \text{for } s = (0,1,0,1,1), s' = (1,0,1,0,1) \end{cases}$$

$$P_{a_2}(s,s') = \frac{1}{q}\begin{cases}
\mu_1 & \text{for } s=(1,0,2,0,1), s'=(0,1,2,0,1)\\
2\mu_2 & \text{for } s=(1,0,2,0,1), s'=(1,0,1,0,2)\\
\mu_3 & \text{for } s=(1,0,2,0,1), s'=(1,0,2,0,0)\\
\mu_1 & \text{for } s=(1,0,2,0,0), s'=(0,1,2,0,0)\\
2\mu_2 & \text{for } s=(1,0,2,0,0), s'=(1,0,1,0,1)\\
\mu_1 & \text{for } s=(1,0,1,0,1), s'=(1,0,2,0,1)\\
\mu_2 & \text{for } s=(1,0,1,0,1), s'=(1,0,0,0,2)\\
\mu_3 & \text{for } s=(1,0,1,0,1), s'=(1,0,1,0,0)\\
\mu_1 & \text{for } s=(1,0,1,0,0), s'=(1,0,2,0,0)\\
\mu_2 & \text{for } s=(1,0,1,0,0), s'=(1,0,0,0,1)\\
\mu_1 & \text{for } s=(1,0,0,1,1), s'=(1,0,1,0,2)\\
2\mu_3 & \text{for } s=(1,0,0,1,1), s'=(1,0,0,0,1)\\
\mu_1 & \text{for } s=(1,0,0,0,1), s'=(1,0,1,0,1)\\
\mu_3 & \text{for } s=(1,0,0,0,1), s'=(1,0,0,0,0)\\
\mu_1 & \text{for } s=(1,0,0,0,1), s'=(1,0,1,0,1)\\
\mu_3 & \text{for } s=(1,0,0,0,1), s'=(1,0,0,0,0)\\
\mu_1 & \text{for } s=(1,0,0,0,0), s'=(1,0,1,0,0)\\
2\mu_2 & \text{for } s=(0,1,2,0,1), s'=(0,1,1,0,2)\\
\mu_3 & \text{for } s=(0,1,2,0,1), s'=(0,1,2,0,0)\\
2\mu_2 & \text{for } s=(0,1,2,0,0), s'=(0,1,1,0,1)\\
\mu_1 & \text{for } s=(0,1,1,0,1), s'=(0,1,2,0,1)\\
2\mu_2 & \text{for } s=(0,1,1,0,1), s'=(1,0,1,0,2)\\
\mu_3 & \text{for } s=(0,1,1,0,1), s'=(1,0,2,0,0)\\
\mu_1 & \text{for } s=(0,1,1,0,0), s'=(0,1,2,0,0)\\
2\mu_2 & \text{for } s=(0,1,1,0,0), s'=(1,0,1,0,1)\\
\mu_1 & \text{for } s=(0,1,0,1,1), s'=(0,1,1,0,2)\\
\mu_2 & \text{for } s=(0,1,0,1,1), s'=(1,0,0,1,2)\\
2\mu_3 & \text{for } s=(0,1,0,1,1), s'=(1,0,1,0,1)
\end{cases}$$

Now we want to show inequality (3.2) to verify that $d_0(s)$ is optimal. For example, for state $(0,1,1,0,1)$, inequality (3.2) simplifies to:

$\mu_1 h_0((0,1,2,0,1)) + 2\mu_2 h_0((1,0,1,0,2)) + \mu_3 h_0((1,0,2,0,0)) - \{\mu_2 h_0((1,0,1,0,2)) + \mu_3 h_0((0,1,1,0,0))\}$

$= \mu_1 h_0((0,1,2,0,1)) + \mu_2 h_0((1,0,1,0,2)) + \mu_3 h_0((1,0,2,0,0)) - \mu_3 h_0((0,1,1,0,0))$

$= \mu_1 \frac{qD}{B} + \mu_2 \frac{qC}{2B}$

where

$D = \mu_1^9\mu_2^2\mu_3 + 3\mu_1^9\mu_2\mu_3^2 + 2\mu_1^9\mu_3^3 + 7\mu_1^8\mu_2^3\mu_3 + 30\mu_1^8\mu_2^2\mu_3^2 + 41\mu_1^8\mu_2\mu_3^3 + 18\mu_1^8\mu_3^4 + 19\mu_1^7\mu_2^4\mu_3 + 107\mu_1^7\mu_2^3\mu_3^2 + 221\mu_1^7\mu_2^2\mu_3^3 + 199\mu_1^7\mu_2\mu_3^4 + 66\mu_1^7\mu_3^5 + 25\mu_1^6\mu_2^5\mu_3 + 191\mu_1^6\mu_2^4\mu_3^2 + 560\mu_1^6\mu_2^3\mu_3^3 + 763\mu_1^6\mu_2^2\mu_3^4 + 497\mu_1^6\mu_2\mu_3^5 + 126\mu_1^6\mu_3^6 + 16\mu_1^5\mu_2^6\mu_3 + 181\mu_1^5\mu_2^5\mu_3^2 + 810\mu_1^5\mu_2^4\mu_3^3 + 1584\mu_1^5\mu_2^3\mu_3^4 + 1505\mu_1^5\mu_2^2\mu_3^5 + 708\mu_1^5\mu_2\mu_3^6 + 132\mu_1^5\mu_3^7 + 4\mu_1^4\mu_2^7\mu_3 + 78\mu_1^4\mu_2^6\mu_3^2 + 712\mu_1^4\mu_2^5\mu_3^3 + 2148\mu_1^4\mu_2^4\mu_3^4 + 2806\mu_1^4\mu_2^3\mu_3^5 + 1814\mu_1^4\mu_2^2\mu_3^6 + 572\mu_1^4\mu_2\mu_3^7 + 72\mu_1^4\mu_3^8 + 8\mu_1^3\mu_2^7\mu_3^2 + 408\mu_1^3\mu_2^6\mu_3^3 + 2086\mu_1^3\mu_2^5\mu_3^4 + 3780\mu_1^3\mu_2^4\mu_3^5 + 3136\mu_1^3\mu_2^3\mu_3^6 + 1268\mu_1^3\mu_2^2\mu_3^7 + 232\mu_1^3\mu_2\mu_3^8 + 16\mu_1^3\mu_3^9 + 168\mu_1^2\mu_2^7\mu_3^3 + 1420\mu_1^2\mu_2^6\mu_3^4 + 3604\mu_1^2\mu_2^5\mu_3^5 + 3956\mu_1^2\mu_2^4\mu_3^6 + 2036\mu_1^2\mu_2^3\mu_3^7 + 456\mu_1^2\mu_2^2\mu_3^8 + 32\mu_1^2\mu_2\mu_3^9 + 32\mu_1\mu_2^8\mu_3^3 + 512\mu_1\mu_2^7\mu_3^4 + 1880\mu_1\mu_2^6\mu_3^5 + 2856\mu_1\mu_2^5\mu_3^6 + 2032\mu_1\mu_2^4\mu_3^7 + 640\mu_1\mu_2^3\mu_3^8 + 64\mu_1\mu_2^2\mu_3^9 + 64\mu_2^8\mu_3^4 + 384\mu_2^7\mu_3^5 + 848\mu_2^6\mu_3^6 + 848\mu_2^5\mu_3^7 + 384\mu_2^4\mu_3^8 + 64\mu_2^3\mu_3^9,$

$B = \mu_1^9\mu_2^3 + 2\mu_1^9\mu_2^2\mu_3 + 2\mu_1^9\mu_2\mu_3^2 + \mu_1^9\mu_3^3 + 7\mu_1^8\mu_2^4 + 23\mu_1^8\mu_2^3\mu_3 + 32\mu_1^8\mu_2^2\mu_3^2 + 25\mu_1^8\mu_2\mu_3^3 + 9\mu_1^8\mu_3^4 + 19\mu_1^7\mu_2^5 + 88\mu_1^7\mu_2^4\mu_3 + 173\mu_1^7\mu_2^3\mu_3^2 + 186\mu_1^7\mu_2^2\mu_3^3 + 116\mu_1^7\mu_2\mu_3^4 + 33\mu_1^7\mu_3^5 + 25\mu_1^6\mu_2^6 + 160\mu_1^6\mu_2^5\mu_3 + 439\mu_1^6\mu_2^4\mu_3^2 + 649\mu_1^6\mu_2^3\mu_3^3 + 560\mu_1^6\mu_2^2\mu_3^4 + 280\mu_1^6\mu_2\mu_3^5 + 63\mu_1^6\mu_3^6 + 16\mu_1^5\mu_2^7 + 149\mu_1^5\mu_2^6\mu_3 + 587\mu_1^5\mu_2^5\mu_3^2 + 1238\mu_1^5\mu_2^4\mu_3^3 + 1460\mu_1^5\mu_2^3\mu_3^4 + 998\mu_1^5\mu_2^2\mu_3^5 + 387\mu_1^5\mu_2\mu_3^6 + 66\mu_1^5\mu_3^7 + 4\mu_1^4\mu_2^8 + 64\mu_1^4\mu_2^7\mu_3 + 407\mu_1^4\mu_2^6\mu_3^2 + 1337\mu_1^4\mu_2^5\mu_3^3 + 2318\mu_1^4\mu_2^4\mu_3^4 + 2140\mu_1^4\mu_2^3\mu_3^5 + 1102\mu_1^4\mu_2^2\mu_3^6 + 304\mu_1^4\mu_2\mu_3^7 + 36\mu_1^4\mu_3^8 + 8\mu_1^3\mu_2^8\mu_3 + 128\mu_1^3\mu_2^7\mu_3^2 + 798\mu_1^3\mu_2^6\mu_3^3 + 2234\mu_1^3\mu_2^5\mu_3^4 + 2996\mu_1^3\mu_2^4\mu_3^5 + 2034\mu_1^3\mu_2^3\mu_3^6 + 714\mu_1^3\mu_2^2\mu_3^7 + 120\mu_1^3\mu_2\mu_3^8 + 8\mu_1^3\mu_3^9 + 16\mu_1^2\mu_2^8\mu_3^2 + 256\mu_1^2\mu_2^7\mu_3^3 + 1268\mu_1^2\mu_2^6\mu_3^4 +$

$2588\mu_1^2\mu_2^5\mu_3^5+2496\mu_1^2\mu_2^4\mu_3^6+1164\mu_1^2\mu_2^3\mu_3^7+240\mu_1^2\mu_2^2\mu_3^8+16\mu_1^2\mu_2\mu_3^9+32\mu_1\mu_2^8\mu_3^3+352\mu_1\mu_2^7\mu_3^4+$
$1152\mu_1\mu_2^6\mu_3^5+1640\mu_1\mu_2^5\mu_3^6+1112\mu_1\mu_2^4\mu_3^7+336\mu_1\mu_2^3\mu_3^8+32\mu_1\mu_2^2\mu_3^9+32\mu_2^8\mu_3^4+192\mu_2^7\mu_3^5+$
$424\mu_2^6\mu_3^6+424\mu_2^5\mu_3^7+192\mu_2^4\mu_3^8+32\mu_2^3\mu_3^9,$

and

$C=\mu_1^9\mu_2^2\mu_3+3\mu_1^9\mu_2\mu_3^2+2\mu_1^9\mu_3^3+7\mu_1^8\mu_2^3\mu_3+30\mu_1^8\mu_2^2\mu_3^2+41\mu_1^8\mu_2\mu_3^3+18\mu_1^8\mu_3^4+19\mu_1^7\mu_2^4\mu_3+$
$105\mu_1^7\mu_2^3\mu_3^2+215\mu_1^7\mu_2^2\mu_3^3+195\mu_1^7\mu_2\mu_3^4+66\mu_1^7\mu_3^5+25\mu_1^6\mu_2^5\mu_3+181\mu_1^6\mu_2^4\mu_3^2+506\mu_1^6\mu_2^3\mu_3^3+$
$691\mu_1^6\mu_2^2\mu_3^4+469\mu_1^6\mu_2\mu_3^5+126\mu_1^6\mu_3^6+16\mu_1^5\mu_2^6\mu_3+165\mu_1^5\mu_2^5\mu_3^2+624\mu_1^5\mu_2^4\mu_3^3+1164\mu_1^5\mu_2^3\mu_3^4+$
$1179\mu_1^5\mu_2^2\mu_3^5+632\mu_1^5\mu_2\mu_3^6+132\mu_1^5\mu_3^7+4\mu_1^4\mu_2^7\mu_3+70\mu_1^4\mu_2^6\mu_3^2+388\mu_1^4\mu_2^5\mu_3^3+988\mu_1^4\mu_2^4\mu_3^4+$
$1374\mu_1^4\mu_2^3\mu_3^5+1126\mu_1^4\mu_2^2\mu_3^6+472\mu_1^4\mu_2\mu_3^7+72\mu_1^4\mu_3^8+8\mu_1^3\mu_2^7\mu_3^2+96\mu_1^3\mu_2^6\mu_3^3+366\mu_1^3\mu_2^5\mu_3^4+$
$696\mu_1^3\mu_2^4\mu_3^5+800\mu_1^3\mu_2^3\mu_3^6+544\mu_1^3\mu_2^2\mu_3^7+168\mu_1^3\mu_2\mu_3^8+16\mu_1^3\mu_3^9+8\mu_1^2\mu_2^7\mu_3^3+52\mu_1^2\mu_2^6\mu_3^4+$
$140\mu_1^2\mu_2^5\mu_3^5+212\mu_1^2\mu_2^4\mu_3^6+196\mu_1^2\mu_2^3\mu_3^7+96\mu_1^2\mu_2^2\mu_3^8+16\mu_1^2\mu_2\mu_3^9.$

Therefore the above expression is trivially positive. Similarly, it can be shown that inequality (3.2) holds for the other states. In this example d_0 makes the Policy Iteration algorithm converge in one step. Hence d_0 is the optimal policy. It should be noted that the policy is independent of μ_1 and μ_2.

(Theorem 4.2) □

Comparing policies described in this and the previous section with the non-constrained case, described in Section 3.4, all policies have similar structures. They clear blocking and send the flexible server to upstream stations whenever possible.

4.1.3 Two constrained flexible servers

Consider a tandem line with three stations and a dedicated server at each station. Assume that there are two flexible servers. One of them can only move between the first and second stations (call this server $\sigma_{1,2}$). The other one can only move between the second and third stations (call this server $\sigma_{2,3}$). The optimal policy is rate dependent. When the service rates are equal, Theorem 4.3 describes the optimal policy.

Theorem 4.3. *The optimal policy prioritizes clearing blocking from the end to the beginning. $\sigma_{1,2}$ hands off its job to $\sigma_{2,3}$, whenever possible. As a result, $\sigma_{1,2}$ becomes free and admits a new job to the system. Flexible servers hand off their jobs to free dedicated servers, whenever possible. Also, blocked dedicated servers hand off their jobs to flexible servers, whenever possible. Otherwise, $\sigma_{1,2}$ is sent to the first station and/or $\sigma_{2,3}$ is sent to the second station.*

Proof. To be able to construct the discrete-time MDP, we choose the following normalization factor: $q = \mu_1 + 2\mu_2 + \mu_3 + \max\{\mu_1 + \mu_2, \mu_1 + \mu_3, \mu_2 + \mu_3, 2\mu_2\}$. The MDP details are as follows.

$$A = \{a_{12}, a_{13}, a_{22}, a_{23}\}$$

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

Let $\bar{s}$ represent the index of state $s \in S$, according to the order represented above.

$$A_{\bar{s}} = \begin{cases} a_{12} & \text{for } \bar{s} = 1, 2 \\ a_{13} & \text{for } \bar{s} = 4, 7, 9 \\ a_{22} & \text{for } \bar{s} = 18, 19, 29, 30 \\ a_{23} & \text{for } \bar{s} = 21, 25, 27, 32, 36, 40 \\ \{a_{12}, a_{13}\} & \text{for } \bar{s} = 3, 5, 6, 8, 10, 11 \\ \{a_{12}, a_{22}\} & \text{for } \bar{s} = 12 \\ \{a_{13}, a_{23}\} & \text{for } \bar{s} = 13, 16, 38, 42, 45, 47 \\ \{a_{22}, a_{23}\} & \text{for } \bar{s} = 20, 24, 31, 35 \\ \{a_{12}, a_{22}, a_{23}\} & \text{for } \bar{s} = 22, 23, 33, 34, 41 \\ \{a_{13}, a_{22}, a_{23}\} & \text{for } \bar{s} = 28 \\ \{a_{12}, a_{13}, a_{22}, a_{23}\} & \text{for } \bar{s} = 14, 15, 17, 26, 37, 39, 43, 44, 46, 48, 49 \end{cases}$$

Our candidate optimal policy d_0 is:

$$d_0(\bar{s}) = \begin{cases} a_{12} & \text{for } \bar{s} = 1, 2, 5, 6, 10, 11, 14, 15, 22, 23, 39, 43, 44, 48, 49 \\ a_{13} & \text{for } \bar{s} = 3, 4, 8, 9, 17, 26, 28, 37, 42, 45, 46, 47 \\ a_{22} & \text{for } \bar{s} = 12, 18, 19, 29, 30, 33, 34 \\ a_{23} & \text{for } \bar{s} = 13, 16, 20, 21, 24, 25, 27, 31, 32, 35, 36, 38, 40, 41 \end{cases}$$

$$r_{d_0}(\bar{s}) = \frac{1}{q} \begin{cases} 0 & \text{for } \bar{s} = 2, 6, 11, 12, 15, 19, 23, 30, 34, 44, 49 \\ \mu_3 & \text{for } \bar{s} = 1, 5, 10, 14, 18, 22, 29, 33, 39, 43, 48 \\ 2\mu_3 & \text{for } \bar{s} = 3, 4, 7, 8, 9, 13, 16, 17, 20, 21, 24, 25, 26, 27, 28, 31, 32, 35, 36, 37, \\ & 38, 40, 41, 42, 45, 46, 47 \end{cases}$$

$$P_{a_{12}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_1 & \text{for } \bar{s}=1,\bar{s}'=18 \\ 2\mu_2 & \text{for } \bar{s}=1,\bar{s}'=4 \\ \mu_3 & \text{for } \bar{s}=1,\bar{s}'=2 \\ 2\mu_1 & \text{for } \bar{s}=2,\bar{s}'=19 \\ 2\mu_2 & \text{for } \bar{s}=2,\bar{s}'=5 \\ 2\mu_1 & \text{for } \bar{s}=3,\bar{s}'=20 \\ \mu_2 & \text{for } \bar{s}=3,\bar{s}'=7 \\ \mu_3 & \text{for } \bar{s}=3,\bar{s}'=5 \\ 2\mu_1 & \text{for } \bar{s}=5,\bar{s}'=1 \\ \mu_2 & \text{for } \bar{s}=5,\bar{s}'=9 \\ \mu_3 & \text{for } \bar{s}=5,\bar{s}'=6 \\ 2\mu_1 & \text{for } \bar{s}=6,\bar{s}'=2 \\ \mu_2 & \text{for } \bar{s}=6,\bar{s}'=10 \\ 2\mu_1 & \text{for } \bar{s}=8,\bar{s}'=3 \\ \mu_3 & \text{for } \bar{s}=8,\bar{s}'=10 \\ 2\mu_1 & \text{for } \bar{s}=10,\bar{s}'=5 \\ \mu_3 & \text{for } \bar{s}=10,\bar{s}'=11 \\ 2\mu_1 & \text{for } \bar{s}=11,\bar{s}'=6 \\ \mu_1 & \text{for } \bar{s}=12,\bar{s}'=30 \\ 2\mu_2 & \text{for } \bar{s}=12,\bar{s}'=14 \\ 2\mu_1 & \text{for } \bar{s}=14,\bar{s}'=18 \\ 2\mu_2 & \text{for } \bar{s}=14,\bar{s}'=4 \\ \mu_3 & \text{for } \bar{s}=14,\bar{s}'=2 \\ 2\mu_1 & \text{for } \bar{s}=15,\bar{s}'=19 \\ 2\mu_2 & \text{for } \bar{s}=15,\bar{s}'=5 \\ 2\mu_1 & \text{for } \bar{s}=17,\bar{s}'=20 \\ \mu_2 & \text{for } \bar{s}=17,\bar{s}'=7 \\ \mu_3 & \text{for } \bar{s}=17,\bar{s}'=5 \\ 2\mu_1 & \text{for } \bar{s}=22,\bar{s}'=18 \\ 2\mu_2 & \text{for } \bar{s}=22,\bar{s}'=4 \\ \mu_3 & \text{for } \bar{s}=22,\bar{s}'=2 \\ 2\mu_1 & \text{for } \bar{s}=23,\bar{s}'=19 \\ 2\mu_2 & \text{for } \bar{s}=23,\bar{s}'=5 \end{cases}$$

$$P_{a_{12}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_1 & \text{for } \bar{s}=26, \bar{s}'=20 \\ \mu_2 & \text{for } \bar{s}=26, \bar{s}'=7 \\ \mu_3 & \text{for } \bar{s}=26, \bar{s}'=5 \\ \mu_1 & \text{for } \bar{s}=33, \bar{s}'=19 \\ 2\mu_2 & \text{for } \bar{s}=33, \bar{s}'=13 \\ \mu_3 & \text{for } \bar{s}=33, \bar{s}'=12 \\ \mu_1 & \text{for } \bar{s}=34, \bar{s}'=30 \\ 2\mu_2 & \text{for } \bar{s}=34, \bar{s}'=14 \\ \mu_1 & \text{for } \bar{s}=37, \bar{s}'=31 \\ \mu_2 & \text{for } \bar{s}=37, \bar{s}'=16 \\ \mu_3 & \text{for } \bar{s}=37, \bar{s}'=14 \\ 2\mu_1 & \text{for } \bar{s}=39, \bar{s}'=18 \\ 2\mu_2 & \text{for } \bar{s}=39, \bar{s}'=4 \\ \mu_3 & \text{for } \bar{s}=39, \bar{s}'=2 \\ \mu_1 & \text{for } \bar{s}=41, \bar{s}'=31 \\ \mu_2 & \text{for } \bar{s}=41, \bar{s}'=16 \\ \mu_3 & \text{for } \bar{s}=41, \bar{s}'=14 \\ 2\mu_1 & \text{for } \bar{s}=43, \bar{s}'=1 \\ \mu_2 & \text{for } \bar{s}=43, \bar{s}'=9 \\ \mu_3 & \text{for } \bar{s}=43, \bar{s}'=6 \\ 2\mu_1 & \text{for } \bar{s}=44, \bar{s}'=2 \\ \mu_2 & \text{for } \bar{s}=44, \bar{s}'=10 \\ 2\mu_1 & \text{for } \bar{s}=46, \bar{s}'=3 \\ \mu_3 & \text{for } \bar{s}=46, \bar{s}'=10 \\ 2\mu_1 & \text{for } \bar{s}=48, \bar{s}'=5 \\ \mu_3 & \text{for } \bar{s}=48, \bar{s}'=11 \\ 2\mu_1 & \text{for } \bar{s}=49, \bar{s}'=6 \end{cases}$$

$$P_{a_{13}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_1 & \text{for } \bar{s}=3,\bar{s}'=21 \\ \mu_2 & \text{for } \bar{s}=3,\bar{s}'=7 \\ 2\mu_3 & \text{for } \bar{s}=3,\bar{s}'=5 \\ 2\mu_1 & \text{for } \bar{s}=4,\bar{s}'=21 \\ \mu_2 & \text{for } \bar{s}=4,\bar{s}'=7 \\ 2\mu_3 & \text{for } \bar{s}=4,\bar{s}'=5 \\ 2\mu_1 & \text{for } \bar{s}=5,\bar{s}'=22 \\ \mu_2 & \text{for } \bar{s}=5,\bar{s}'=9 \\ \mu_3 & \text{for } \bar{s}=5,\bar{s}'=6 \\ 2\mu_1 & \text{for } \bar{s}=6,\bar{s}'=23 \\ \mu_2 & \text{for } \bar{s}=6,\bar{s}'=10 \\ 2\mu_1 & \text{for } \bar{s}=7,\bar{s}'=25 \\ 2\mu_3 & \text{for } \bar{s}=7,\bar{s}'=8 \\ 2\mu_1 & \text{for } \bar{s}=8,\bar{s}'=4 \\ 2\mu_3 & \text{for } \bar{s}=8,\bar{s}'=10 \\ 2\mu_1 & \text{for } \bar{s}=9,\bar{s}'=4 \\ 2\mu_3 & \text{for } \bar{s}=9,\bar{s}'=10 \\ 2\mu_1 & \text{for } \bar{s}=10,\bar{s}'=5 \\ \mu_3 & \text{for } \bar{s}=10,\bar{s}'=11 \\ 2\mu_1 & \text{for } \bar{s}=11,\bar{s}'=6 \\ \mu_1 & \text{for } \bar{s}=13,\bar{s}'=32 \\ \mu_2 & \text{for } \bar{s}=13,\bar{s}'=16 \\ 2\mu_3 & \text{for } \bar{s}=13,\bar{s}'=14 \\ \mu_1 & \text{for } \bar{s}=14,\bar{s}'=33 \\ \mu_2 & \text{for } \bar{s}=14,\bar{s}'=42 \\ \mu_3 & \text{for } \bar{s}=14,\bar{s}'=15 \\ \mu_1 & \text{for } \bar{s}=15,\bar{s}'=34 \\ \mu_2 & \text{for } \bar{s}=15,\bar{s}'=43 \\ \mu_1 & \text{for } \bar{s}=16,\bar{s}'=36 \\ 2\mu_3 & \text{for } \bar{s}=16,\bar{s}'=17 \\ 2\mu_1 & \text{for } \bar{s}=17,\bar{s}'=21 \\ \mu_2 & \text{for } \bar{s}=17,\bar{s}'=7 \\ 2\mu_3 & \text{for } \bar{s}=17,\bar{s}'=5 \end{cases}$$

$$P_{a_{13}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_1 & \text{for } \bar{s}=26,\bar{s}'=21 \\ \mu_2 & \text{for } \bar{s}=26,\bar{s}'=7 \\ 2\mu_3 & \text{for } \bar{s}=26,\bar{s}'=5 \\ 2\mu_1 & \text{for } \bar{s}=28,\bar{s}'=25 \\ 2\mu_3 & \text{for } \bar{s}=28,\bar{s}'=8 \\ \mu_1 & \text{for } \bar{s}=37,\bar{s}'=36 \\ 2\mu_3 & \text{for } \bar{s}=37,\bar{s}'=17 \\ \mu_1 & \text{for } \bar{s}=38,\bar{s}'=32 \\ \mu_2 & \text{for } \bar{s}=38,\bar{s}'=16 \\ 2\mu_3 & \text{for } \bar{s}=38,\bar{s}'=14 \\ \mu_1 & \text{for } \bar{s}=39,\bar{s}'=33 \\ \mu_2 & \text{for } \bar{s}=39,\bar{s}'=42 \\ \mu_3 & \text{for } \bar{s}=39,\bar{s}'=15 \\ 2\mu_1 & \text{for } \bar{s}=42,\bar{s}'=21 \\ \mu_2 & \text{for } \bar{s}=42,\bar{s}'=7 \\ 2\mu_3 & \text{for } \bar{s}=42,\bar{s}'=5 \\ 2\mu_1 & \text{for } \bar{s}=43,\bar{s}'=22 \\ \mu_2 & \text{for } \bar{s}=43,\bar{s}'=9 \\ 2\mu_3 & \text{for } \bar{s}=43,\bar{s}'=6 \\ 2\mu_1 & \text{for } \bar{s}=44,\bar{s}'=23 \\ \mu_2 & \text{for } \bar{s}=44,\bar{s}'=10 \\ 2\mu_1 & \text{for } \bar{s}=45,\bar{s}'=25 \\ 2\mu_3 & \text{for } \bar{s}=45,\bar{s}'=8 \\ 2\mu_1 & \text{for } \bar{s}=46,\bar{s}'=4 \\ 2\mu_3 & \text{for } \bar{s}=46,\bar{s}'=10 \\ 2\mu_1 & \text{for } \bar{s}=47,\bar{s}'=4 \\ 2\mu_3 & \text{for } \bar{s}=47,\bar{s}'=10 \\ 2\mu_1 & \text{for } \bar{s}=48,\bar{s}'=5 \\ \mu_3 & \text{for } \bar{s}=48,\bar{s}'=11 \\ 2\mu_1 & \text{for } \bar{s}=49,\bar{s}'=6 \end{cases}$$

$$P_{a_{22}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases}
\mu_1 & \text{for } \bar{s}=12, \bar{s}'=30\\
3\mu_2 & \text{for } \bar{s}=12, \bar{s}'=22\\
\mu_1 & \text{for } \bar{s}=14, \bar{s}'=18\\
2\mu_2 & \text{for } \bar{s}=14, \bar{s}'=42\\
\mu_3 & \text{for } \bar{s}=14, \bar{s}'=23\\
\mu_1 & \text{for } \bar{s}=15, \bar{s}'=19\\
2\mu_2 & \text{for } \bar{s}=15, \bar{s}'=43\\
\mu_1 & \text{for } \bar{s}=17, \bar{s}'=20\\
\mu_2 & \text{for } \bar{s}=17, \bar{s}'=45\\
\mu_3 & \text{for } \bar{s}=17, \bar{s}'=43\\
\mu_1 & \text{for } \bar{s}=18, \bar{s}'=29\\
3\mu_2 & \text{for } \bar{s}=18, \bar{s}'=21\\
\mu_3 & \text{for } \bar{s}=18, \bar{s}'=19\\
\mu_1 & \text{for } \bar{s}=19, \bar{s}'=30\\
3\mu_2 & \text{for } \bar{s}=19, \bar{s}'=22\\
\mu_1 & \text{for } \bar{s}=20, \bar{s}'=31\\
2\mu_2 & \text{for } \bar{s}=20, \bar{s}'=25\\
\mu_3 & \text{for } \bar{s}=20, \bar{s}'=22\\
\mu_1 & \text{for } \bar{s}=22, \bar{s}'=18\\
2\mu_2 & \text{for } \bar{s}=22, \bar{s}'=42\\
\mu_3 & \text{for } \bar{s}=22, \bar{s}'=23\\
\mu_1 & \text{for } \bar{s}=23, \bar{s}'=19\\
2\mu_2 & \text{for } \bar{s}=23, \bar{s}'=43\\
\mu_1 & \text{for } \bar{s}=24, \bar{s}'=35\\
\mu_2 & \text{for } \bar{s}=24, \bar{s}'=27\\
\mu_3 & \text{for } \bar{s}=24, \bar{s}'=42\\
\mu_1 & \text{for } \bar{s}=26, \bar{s}'=20\\
\mu_2 & \text{for } \bar{s}=26, \bar{s}'=45\\
\mu_3 & \text{for } \bar{s}=26, \bar{s}'=43\\
\mu_1 & \text{for } \bar{s}=28, \bar{s}'=24\\
\mu_3 & \text{for } \bar{s}=28, \bar{s}'=47\\
2\mu_2 & \text{for } \bar{s}=29, \bar{s}'=32\\
\mu_3 & \text{for } \bar{s}=29, \bar{s}'=30
\end{cases}$$

$$P_{a_{22}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 3\mu_2 & \text{for } \bar{s}=30, \bar{s}'=33 \\ 2\mu_2 & \text{for } \bar{s}=31, \bar{s}'=36 \\ \mu_3 & \text{for } \bar{s}=31, \bar{s}'=33 \\ \mu_1 & \text{for } \bar{s}=33, \bar{s}'=29 \\ 3\mu_2 & \text{for } \bar{s}=33, \bar{s}'=21 \\ \mu_3 & \text{for } \bar{s}=33, \bar{s}'=19 \\ \mu_1 & \text{for } \bar{s}=34, \bar{s}'=30 \\ 3\mu_2 & \text{for } \bar{s}=34, \bar{s}'=22 \\ \mu_2 & \text{for } \bar{s}=35, \bar{s}'=40 \\ \mu_3 & \text{for } \bar{s}=35, \bar{s}'=37 \\ \mu_1 & \text{for } \bar{s}=37, \bar{s}'=31 \\ 2\mu_2 & \text{for } \bar{s}=37, \bar{s}'=25 \\ \mu_3 & \text{for } \bar{s}=37, \bar{s}'=22 \\ \mu_1 & \text{for } \bar{s}=39, \bar{s}'=18 \\ 2\mu_2 & \text{for } \bar{s}=39, \bar{s}'=42 \\ \mu_3 & \text{for } \bar{s}=39, \bar{s}'=23 \\ \mu_1 & \text{for } \bar{s}=41, \bar{s}'=35 \\ \mu_2 & \text{for } \bar{s}=41, \bar{s}'=27 \\ \mu_3 & \text{for } \bar{s}=41, \bar{s}'=42 \\ \mu_1 & \text{for } \bar{s}=43, \bar{s}'=22 \\ \mu_2 & \text{for } \bar{s}=43, \bar{s}'=47 \\ \mu_3 & \text{for } \bar{s}=43, \bar{s}'=44 \\ \mu_1 & \text{for } \bar{s}=44, \bar{s}'=23 \\ \mu_2 & \text{for } \bar{s}=44, \bar{s}'=48 \\ \mu_1 & \text{for } \bar{s}=46, \bar{s}'=26 \\ \mu_3 & \text{for } \bar{s}=46, \bar{s}'=48 \\ \mu_1 & \text{for } \bar{s}=48, \bar{s}'=43 \\ \mu_3 & \text{for } \bar{s}=48, \bar{s}'=49 \\ \mu_1 & \text{for } \bar{s}=49, \bar{s}'=44 \end{cases}$$

$$P_{a_{23}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases}
\mu_1 & \text{for } \bar{s}=13, \bar{s}'=32\\
2\mu_2 & \text{for } \bar{s}=13, \bar{s}'=25\\
2\mu_3 & \text{for } \bar{s}=13, \bar{s}'=22\\
\mu_1 & \text{for } \bar{s}=14, \bar{s}'=33\\
2\mu_2 & \text{for } \bar{s}=14, \bar{s}'=42\\
\mu_3 & \text{for } \bar{s}=14, \bar{s}'=23\\
\mu_1 & \text{for } \bar{s}=15, \bar{s}'=34\\
2\mu_2 & \text{for } \bar{s}=15, \bar{s}'=43\\
\mu_1 & \text{for } \bar{s}=16, \bar{s}'=36\\
\mu_2 & \text{for } \bar{s}=16, \bar{s}'=27\\
2\mu_3 & \text{for } \bar{s}=16, \bar{s}'=26\\
\mu_1 & \text{for } \bar{s}=17, \bar{s}'=21\\
\mu_2 & \text{for } \bar{s}=17, \bar{s}'=45\\
2\mu_3 & \text{for } \bar{s}=17, \bar{s}'=43\\
\mu_1 & \text{for } \bar{s}=20, \bar{s}'=32\\
2\mu_2 & \text{for } \bar{s}=20, \bar{s}'=25\\
2\mu_3 & \text{for } \bar{s}=20, \bar{s}'=22\\
\mu_1 & \text{for } \bar{s}=21, \bar{s}'=32\\
2\mu_2 & \text{for } \bar{s}=21, \bar{s}'=25\\
2\mu_3 & \text{for } \bar{s}=21, \bar{s}'=22\\
\mu_1 & \text{for } \bar{s}=22, \bar{s}'=33\\
2\mu_2 & \text{for } \bar{s}=22, \bar{s}'=42\\
\mu_3 & \text{for } \bar{s}=22, \bar{s}'=23\\
\mu_1 & \text{for } \bar{s}=23, \bar{s}'=34\\
2\mu_2 & \text{for } \bar{s}=23, \bar{s}'=43\\
\mu_1 & \text{for } \bar{s}=24, \bar{s}'=36\\
\mu_2 & \text{for } \bar{s}=24, \bar{s}'=27\\
2\mu_3 & \text{for } \bar{s}=24, \bar{s}'=26\\
\mu_1 & \text{for } \bar{s}=25, \bar{s}'=36\\
\mu_2 & \text{for } \bar{s}=25, \bar{s}'=27\\
2\mu_3 & \text{for } \bar{s}=25, \bar{s}'=26\\
\mu_1 & \text{for } \bar{s}=26, \bar{s}'=21\\
\mu_2 & \text{for } \bar{s}=26, \bar{s}'=45\\
2\mu_3 & \text{for } \bar{s}=26, \bar{s}'=43
\end{cases}$$

$$P_{a_{23}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases}
\mu_1 & \text{for } \bar{s}=27, \bar{s}'=40\\
2\mu_3 & \text{for } \bar{s}=27, \bar{s}'=28\\
\mu_1 & \text{for } \bar{s}=28, \bar{s}'=25\\
2\mu_2 & \text{for } \bar{s}=28, \bar{s}'=46\\
2\mu_2 & \text{for } \bar{s}=31, \bar{s}'=36\\
2\mu_3 & \text{for } \bar{s}=31, \bar{s}'=33\\
2\mu_2 & \text{for } \bar{s}=32, \bar{s}'=36\\
2\mu_3 & \text{for } \bar{s}=32, \bar{s}'=33\\
2\mu_2 & \text{for } \bar{s}=33, \bar{s}'=38\\
\mu_3 & \text{for } \bar{s}=33, \bar{s}'=34\\
2\mu_2 & \text{for } \bar{s}=34, \bar{s}'=39\\
\mu_2 & \text{for } \bar{s}=35, \bar{s}'=40\\
2\mu_3 & \text{for } \bar{s}=35, \bar{s}'=37\\
\mu_2 & \text{for } \bar{s}=36, \bar{s}'=40\\
2\mu_3 & \text{for } \bar{s}=36, \bar{s}'=37\\
\mu_1 & \text{for } \bar{s}=37, \bar{s}'=32\\
2\mu_2 & \text{for } \bar{s}=37, \bar{s}'=25\\
2\mu_3 & \text{for } \bar{s}=37, \bar{s}'=22\\
\mu_1 & \text{for } \bar{s}=38, \bar{s}'=32\\
2\mu_2 & \text{for } \bar{s}=38, \bar{s}'=25\\
2\mu_3 & \text{for } \bar{s}=38, \bar{s}'=22\\
\mu_1 & \text{for } \bar{s}=39, \bar{s}'=33\\
2\mu_2 & \text{for } \bar{s}=39, \bar{s}'=42\\
\mu_3 & \text{for } \bar{s}=39, \bar{s}'=23\\
2\mu_3 & \text{for } \bar{s}=40, \bar{s}'=41\\
\mu_1 & \text{for } \bar{s}=41, \bar{s}'=36\\
\mu_2 & \text{for } \bar{s}=41, \bar{s}'=27\\
2\mu_3 & \text{for } \bar{s}=41, \bar{s}'=26\\
\mu_1 & \text{for } \bar{s}=42, \bar{s}'=21\\
\mu_2 & \text{for } \bar{s}=42, \bar{s}'=45\\
2\mu_3 & \text{for } \bar{s}=42, \bar{s}'=43
\end{cases}$$

$$P_{a_{23}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} \mu_1 & \text{for } \bar{s}=43, \bar{s}'=22 \\ \mu_2 & \text{for } \bar{s}=43, \bar{s}'=47 \\ \mu_3 & \text{for } \bar{s}=43, \bar{s}'=44 \\ \mu_1 & \text{for } \bar{s}=44, \bar{s}'=23 \\ \mu_2 & \text{for } \bar{s}=44, \bar{s}'=48 \\ \mu_1 & \text{for } \bar{s}=45, \bar{s}'=25 \\ 2\mu_3 & \text{for } \bar{s}=45, \bar{s}'=46 \\ \mu_1 & \text{for } \bar{s}=46, \bar{s}'=42 \\ 2\mu_2 & \text{for } \bar{s}=46, \bar{s}'=48 \\ \mu_1 & \text{for } \bar{s}=47, \bar{s}'=42 \\ 2\mu_2 & \text{for } \bar{s}=47, \bar{s}'=48 \\ \mu_1 & \text{for } \bar{s}=48, \bar{s}'=43 \\ \mu_3 & \text{for } \bar{s}=48, \bar{s}'=49 \\ \mu_1 & \text{for } \bar{s}=49, \bar{s}'=44 \end{cases}$$

With the above set up, we ran the Policy Iteration algorithm to show that inequality (3.2) holds and hence the stated policy is optimal.

(Theorem 4.3) □

4.1.4 Two non-constrained flexible servers

Consider a tandem line with three stations and a dedicated server at each station. Assume two flexible servers exist that can move between all stations. The optimal policy is rate dependent. When the service rates are equal, Theorem 4.4 describes the optimal policy (assuming that hand-off is always performed).

Theorem 4.4. *The optimal policy clears blocking from the end to the beginning. The policy sends the flexible servers to the first station, whenever possible.*

Proof. To be able to construct the discrete-time MDP, we choose the following

normalization factor: $q = 3(\mu_1 + \mu_2 + \mu_3)$. The MDP details are as follows.

$$A = \{a_{11}, a_{12}, a_{13}, a_{22}, a_{23}, a_{33}\}$$

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

Let $\bar{s}$ represent the index of state $s \in S$, according to the order represented above.

$$A_{\bar{s}} = \begin{cases} a_{11} & \text{for } \bar{s} = 1,2,3,4,5 \\ a_{12} & \text{for } \bar{s} = 6,7 \\ a_{13} & \text{for } \bar{s} = 8,9,11 \\ a_{22} & \text{for } \bar{s} = 12,13,18,19 \\ a_{23} & \text{for } \bar{s} = 14,20 \\ a_{33} & \text{for } \bar{s} = 15,16,21,22 \\ \{a_{11}, a_{13}\} & \text{for } \bar{s} = 10 \\ \{a_{13}, a_{33}\} & \text{for } \bar{s} = 17 \end{cases}$$

Our candidate optimal policy d_0 is:

$$d_0(\bar{s}) = \begin{cases} a_{11} & \text{for } \bar{s} = 1,2,3,4,5,10 \\ a_{12} & \text{for } \bar{s} = 6,7 \\ a_{13} & \text{for } \bar{s} = 8,9,11,17 \\ a_{22} & \text{for } \bar{s} = 12,13,18,19 \\ a_{23} & \text{for } \bar{s} = 14,20 \\ a_{33} & \text{for } \bar{s} = 15,16,21,22 \end{cases}$$

$$r_{d_0}(\bar{s}) = \frac{1}{q} \begin{cases} 0 & \text{for } \bar{s} = 2,5,7,13,19 \\ \mu_3 & \text{for } \bar{s} = 1,3,4,6,10,12,18 \\ 2\mu_3 & \text{for } \bar{s} = 8,9,11,14,17,20 \\ 3\mu_3 & \text{for } \bar{s} = 15,16,21,22 \end{cases}$$

$$P_{a_{11}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 3\mu_1 & \text{for } \bar{s}=1,\bar{s}'=6 \\ \mu_2 & \text{for } \bar{s}=1,\bar{s}'=3 \\ \mu_3 & \text{for } \bar{s}=1,\bar{s}'=2 \\ 3\mu_1 & \text{for } \bar{s}=2,\bar{s}'=7 \\ \mu_2 & \text{for } \bar{s}=2,\bar{s}'=4 \\ 3\mu_1 & \text{for } \bar{s}=3,\bar{s}'=8 \\ \mu_3 & \text{for } \bar{s}=3,\bar{s}'=4 \\ 3\mu_1 & \text{for } \bar{s}=4,\bar{s}'=1 \\ \mu_3 & \text{for } \bar{s}=4,\bar{s}'=5 \\ 3\mu_1 & \text{for } \bar{s}=5,\bar{s}'=2 \\ 3\mu_1 & \text{for } \bar{s}=10,\bar{s}'=8 \\ \mu_3 & \text{for } \bar{s}=10,\bar{s}'=4 \end{cases}$$

$$P_{a_{12}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_1 & \text{for } \bar{s}=6,\bar{s}'=12 \\ 2\mu_2 & \text{for } \bar{s}=6,\bar{s}'=8 \\ \mu_3 & \text{for } \bar{s}=6,\bar{s}'=7 \\ 2\mu_1 & \text{for } \bar{s}=7,\bar{s}'=17 \\ 2\mu_2 & \text{for } \bar{s}=7,\bar{s}'=1 \end{cases}$$

$$P_{a_{13}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_1 & \text{for } \bar{s}=8,\bar{s}'=14 \\ \mu_2 & \text{for } \bar{s}=8,\bar{s}'=9 \\ 2\mu_3 & \text{for } \bar{s}=8,\bar{s}'=1 \\ 2\mu_1 & \text{for } \bar{s}=9,\bar{s}'=15 \\ 2\mu_3 & \text{for } \bar{s}=9,\bar{s}'=10 \\ 2\mu_1 & \text{for } \bar{s}=10,\bar{s}'=8 \\ 2\mu_3 & \text{for } \bar{s}=10,\bar{s}'=4 \\ 2\mu_1 & \text{for } \bar{s}=11,\bar{s}'=8 \\ 2\mu_3 & \text{for } \bar{s}=11,\bar{s}'=4 \\ 2\mu_1 & \text{for } \bar{s}=17,\bar{s}'=15 \\ 2\mu_3 & \text{for } \bar{s}=17,\bar{s}'=10 \end{cases}$$

$$P_{a_{22}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} \mu_1 & \text{for } \bar{s}=12, \bar{s}'=18 \\ 3\mu_2 & \text{for } \bar{s}=12, \bar{s}'=14 \\ \mu_3 & \text{for } \bar{s}=12, \bar{s}'=13 \\ \mu_1 & \text{for } \bar{s}=13, \bar{s}'=19 \\ 3\mu_2 & \text{for } \bar{s}=13, \bar{s}'=6 \\ 3\mu_2 & \text{for } \bar{s}=18, \bar{s}'=20 \\ \mu_3 & \text{for } \bar{s}=18, \bar{s}'=19 \\ 3\mu_2 & \text{for } \bar{s}=19, \bar{s}'=13 \end{cases}$$

$$P_{a_{23}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} \mu_1 & \text{for } \bar{s}=14, \bar{s}'=20 \\ 2\mu_2 & \text{for } \bar{s}=14, \bar{s}'=15 \\ 2\mu_3 & \text{for } \bar{s}=14, \bar{s}'=6 \\ 2\mu_2 & \text{for } \bar{s}=20, \bar{s}'=21 \\ 2\mu_3 & \text{for } \bar{s}=20, \bar{s}'=12 \end{cases}$$

$$P_{a_{33}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} \mu_1 & \text{for } \bar{s}=15, \bar{s}'=21 \\ \mu_2 & \text{for } \bar{s}=15, \bar{s}'=16 \\ 3\mu_3 & \text{for } \bar{s}=15, \bar{s}'=8 \\ \mu_1 & \text{for } \bar{s}=16, \bar{s}'=22 \\ 3\mu_2 & \text{for } \bar{s}=16, \bar{s}'=17 \\ \mu_1 & \text{for } \bar{s}=17, \bar{s}'=15 \\ 3\mu_3 & \text{for } \bar{s}=17, \bar{s}'=11 \\ \mu_2 & \text{for } \bar{s}=21, \bar{s}'=22 \\ 3\mu_3 & \text{for } \bar{s}=21, \bar{s}'=14 \\ 3\mu_3 & \text{for } \bar{s}=22, \bar{s}'=15 \end{cases}$$

With the above set up, we ran the Policy Iteration algorithm to show inequality (3.2) to verify that $d_0(s)$ is optimal.

(Theorem 4.4) □

Now we compare the structure of the optimal policies with two flexible servers, when the workload is balanced. In both constrained and non-constrained cases, the optimal policies prioritize clearing blocking. Both of the policies coordinate allocations such that flexible servers are freed to send them to the first station. Both of the policies are rate dependent.

In terms of throughput results, Table 4.1 compares the two policies for a number of workloads. In this table, the entries in the header row represent service rate vectors and the other table entries are throughput values.

Flexibility \ Rates	$(1,1,1)$	$(2,1,1)$	$(1,2,1)$	$(1,1,2)$
No flexibility	0.8873	1.2812	1.1186	1.2888
Constrained	1.3606	1.6355	1.5031	1.6038
Non-constrained	1.46	1.6407	1.7252	1.6435
All flexible	1.6667	2.0	2.0	2.0

Table 4.1: Comparison of throughputs for different flexibility situations

For the constrained and non-constrained situations, throughput values are calculated through solving the corresponding MDPs. For example, the throughput for the constrained case with a balanced workload is given by

$$T = \mu_3(p_1 + p_5 + p_{10} + p_{14} + p_{18} + p_{22} + p_{29} + p_{33} + p_{39} + p_{43} + p_{48}) + 2\mu_3(p_3 + p_4 + p_7 + p_8 + p_9 + p_{13} + p_{16} + p_{17} + p_{20} + p_{21} + p_{24} + p_{25} + p_{26} + p_{27} + p_{28} + p_{31} + p_{32} + p_{35} + p_{36} + p_{37} + p_{38} + p_{40} + p_{41} + p_{42} + p_{45} + p_{46} + p_{47})$$

where $p_{\bar{s}}$ is the steady state probability of being in state $\bar{s}$.

The first row of the table represents the situation where all servers are dedicated. We solved the corresponding Markov chains to determine throughput values. For

service rates $(1, 1, 1)$ and $(2, 1, 1)$, the optimal server allocation is $(1, 2, 2)$. For service rates $(1, 2, 1)$ and $(1, 1, 2)$, the optimal server allocations are $(2, 1, 2)$ and $(2, 2, 1)$, respectively.

The last row of the table represents the situation where all servers are fully flexible. The optimal policy allocates a new job to each server and the server travels through all stations of the tandem line, until the job departs from the system. The throughput in this case is calculated by

$$T = \frac{|S| + F}{W}$$

As Table 4.1 illustrates, constrained flexibility results in lower throughput values compared to non-constrained flexibility. Also, the fully flexible situation results in lower throughput values compared to the situation where all servers are flexible. As W decreases, the throughput is increased in all cases.

Notice that throughput improvements are significant, when we modify an allocation with all dedicated servers to include constrained flexible servers. However, constrained and non-constrained flexibilities result in close throughput values. Hence, if making servers fully flexible is costly in practice, we can constrain their flexibility (with a lower cost) and still benefit from similar throughput improvements.

4.2 Tandem lines with four stations

In this section we study the following three cases: there is a flexible server which only moves between the second and third stations; there is a flexible server which only moves between the first and second stations; and finally there is a flexible server

which only moves between the third and fourth stations. We only provide the details of the first case. The details of the other two cases are similar.

4.2.1 Flexible server between second and third stations

Consider a tandem line with four stations and a dedicated server at each station. Assume there is a flexible server which can only move between the second and third stations. The optimal policy is rate dependent. When the workload is distributed uniformly, Theorem 4.5 describes the optimal policy for equal workloads.

Theorem 4.5. *For equal workloads, the optimal policy prioritizes clearing blocking. It performs hand-off and allocates the flexible server to the second station, whenever possible. Otherwise the flexible server is assigned to the third station.*

Proof. To be able to construct the discrete-time MDP, we choose the following normalization factor: $q = \mu_1 + 2\mu_2 + \mu_3 + \max\{\mu_2, \mu_3\}$. The MDP details are as follows.

$$A = \{a_2, a_3\}$$

$$S = \{(1,0,2,0,1,0,1),(1,0,2,0,1,0,0),(1,0,2,0,0,1,1),(1,0,2,0,0,0,1),$$
$$(1,0,2,0,0,0,0),(1,0,1,0,2,0,1),(1,0,1,0,2,0,0),(1,0,1,0,1,1,1),(1,0,1,0,1,0,1),$$
$$(1,0,1,0,1,0,0),(1,0,1,0,0,2,1),(1,0,1,0,0,1,1),(1,0,1,0,0,0,1),(1,0,1,0,0,0,0),$$
$$(1,0,0,1,2,0,1),(1,0,0,1,2,0,0),(1,0,0,1,1,1,1),(1,0,0,1,1,0,1),(1,0,0,1,0,2,1),$$
$$(1,0,0,0,2,0,1),(1,0,0,0,2,0,0),(1,0,0,0,1,1,1),(1,0,0,0,1,0,1),(1,0,0,0,1,0,0),$$
$$(1,0,0,0,0,2,1),(1,0,0,0,0,1,1),(1,0,0,0,0,0,1),(1,0,0,0,0,0,0),(0,1,2,0,1,0,1),$$
$$(0,1,2,0,1,0,0),(0,1,2,0,0,1,1),(0,1,2,0,0,0,1),(0,1,2,0,0,0,0),(0,1,1,0,2,0,1),$$
$$(0,1,1,0,2,0,0),(0,1,1,0,1,1,1),(0,1,1,0,1,0,1),(0,1,1,0,1,0,0),(0,1,1,0,0,2,1),$$
$$(0,1,1,0,0,1,1),(0,1,1,0,0,0,1),(0,1,1,0,0,0,0),(0,1,0,1,2,0,1),(0,1,0,1,2,0,0),$$
$$(0,1,0,1,1,1,1),(0,1,0,1,1,0,1),(0,1,0,1,0,2,1),(0,1,0,1,0,1,1)\}$$

Let $\bar{s}$ represent the index of state $s \in S$, according to the order represented above.

$$A_{\bar{s}} = \begin{cases} a_2 & \text{for } \bar{s} = 1,2,3,4,5,14,23,24,26,27,28,29,30,31,32,33,38,40,42 \\ a_3 & \text{for } \bar{s} = 6,7,8,11,15,16,17,19,20,21,22,25,34,35,36,39,43,44,45,47 \\ \{a_2, a_3\} & \text{for } \bar{s} = 9,10,12,13,18,37,41,46,48 \end{cases}$$

Our candidate optimal policy d_0 is:

$$d_0(\bar{s}) = \begin{cases} a_2 & \text{for } \bar{s} = 1,2,3,4,5,9,10,12,13,14,18,23,24,26,27,28,29,30,31,32,33, \\ & 37,38,40,41,42,46,48 \\ a_3 & \text{for } \bar{s} = 6,7,8,11,15,16,17,19,20,21,22,25,34,35,36,39,43,44,45,47 \end{cases}$$

$$r_{d_0}(\bar{s}) = \frac{1}{q} \begin{cases} 0 & \text{for } \bar{s} = 2,5,7,10,14,16,21,24,28,30,33,35,38,42,44 \\ \mu_4 & \text{for } \bar{s} = 1,3,4,6,8,9,11,12,13,15,17,18,19,20,22,23,25,26,27,29,31, \\ & 32,34,36,37,39,40,41,43,45,46,47,48 \end{cases}$$

$$P_{a_2}(\bar{s}, \bar{s}') = \frac{1}{q} \begin{cases} \mu_1 & \text{for } \bar{s} = 1, \bar{s}' = 29 \\ 2\mu_2 & \text{for } \bar{s} = 1, \bar{s}' = 6 \\ \mu_3 & \text{for } \bar{s} = 1, \bar{s}' = 3 \\ \mu_4 & \text{for } \bar{s} = 1, \bar{s}' = 2 \\ \mu_1 & \text{for } \bar{s} = 2, \bar{s}' = 30 \\ 2\mu_2 & \text{for } \bar{s} = 2, \bar{s}' = 7 \\ \mu_3 & \text{for } \bar{s} = 2, \bar{s}' = 4 \\ \mu_1 & \text{for } \bar{s} = 3, \bar{s}' = 31 \\ 2\mu_2 & \text{for } \bar{s} = 3, \bar{s}' = 8 \\ \mu_4 & \text{for } \bar{s} = 3, \bar{s}' = 4 \\ \mu_1 & \text{for } \bar{s} = 4, \bar{s}' = 32 \\ 2\mu_2 & \text{for } \bar{s} = 4, \bar{s}' = 9 \\ \mu_4 & \text{for } \bar{s} = 4, \bar{s}' = 5 \\ \mu_1 & \text{for } \bar{s} = 5, \bar{s}' = 33 \\ 2\mu_2 & \text{for } \bar{s} = 5, \bar{s}' = 10 \\ \mu_1 & \text{for } \bar{s} = 9, \bar{s}' = 1 \\ \mu_2 & \text{for } \bar{s} = 9, \bar{s}' = 20 \\ \mu_3 & \text{for } \bar{s} = 9, \bar{s}' = 12 \end{cases}$$

$$P_{a_2}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases}
\mu_4 & \text{for } \bar{s}=9,\bar{s}'=10\\
\mu_1 & \text{for } \bar{s}=10,\bar{s}'=2\\
\mu_2 & \text{for } \bar{s}=10,\bar{s}'=21\\
\mu_3 & \text{for } \bar{s}=10,\bar{s}'=13\\
\mu_1 & \text{for } \bar{s}=12,\bar{s}'=3\\
\mu_2 & \text{for } \bar{s}=12,\bar{s}'=22\\
\mu_4 & \text{for } \bar{s}=12,\bar{s}'=13\\
\mu_1 & \text{for } \bar{s}=13,\bar{s}'=4\\
\mu_2 & \text{for } \bar{s}=13,\bar{s}'=23\\
\mu_4 & \text{for } \bar{s}=13,\bar{s}'=14\\
\mu_1 & \text{for } \bar{s}=14,\bar{s}'=5\\
\mu_2 & \text{for } \bar{s}=14,\bar{s}'=24\\
\mu_1 & \text{for } \bar{s}=18,\bar{s}'=6\\
2\mu_3 & \text{for } \bar{s}=18,\bar{s}'=22\\
\mu_1 & \text{for } \bar{s}=23,\bar{s}'=9\\
\mu_3 & \text{for } \bar{s}=23,\bar{s}'=26\\
\mu_4 & \text{for } \bar{s}=23,\bar{s}'=24\\
\mu_1 & \text{for } \bar{s}=24,\bar{s}'=10\\
\mu_3 & \text{for } \bar{s}=24,\bar{s}'=27\\
\mu_1 & \text{for } \bar{s}=26,\bar{s}'=12\\
\mu_4 & \text{for } \bar{s}=26,\bar{s}'=27\\
\mu_1 & \text{for } \bar{s}=27,\bar{s}'=13\\
\mu_4 & \text{for } \bar{s}=27,\bar{s}'=28\\
\mu_1 & \text{for } \bar{s}=28,\bar{s}'=14\\
\mu_1 & \text{for } \bar{s}=29,\bar{s}'=34\\
\mu_3 & \text{for } \bar{s}=29,\bar{s}'=31\\
\mu_4 & \text{for } \bar{s}=29,\bar{s}'=30\\
2\mu_2 & \text{for } \bar{s}=30,\bar{s}'=35\\
\mu_3 & \text{for } \bar{s}=30,\bar{s}'=32\\
2\mu_2 & \text{for } \bar{s}=31,\bar{s}'=36\\
\mu_4 & \text{for } \bar{s}=31,\bar{s}'=32\\
2\mu_2 & \text{for } \bar{s}=32,\bar{s}'=1\\
\mu_4 & \text{for } \bar{s}=32,\bar{s}'=33
\end{cases}$$

$$P_{a_2}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_2 & \text{for } \bar{s}=33, \bar{s}'=2 \\ \mu_1 & \text{for } \bar{s}=37, \bar{s}'=29 \\ 2\mu_2 & \text{for } \bar{s}=37, \bar{s}'=6 \\ \mu_3 & \text{for } \bar{s}=37, \bar{s}'=3 \\ \mu_4 & \text{for } \bar{s}=37, \bar{s}'=2 \\ \mu_1 & \text{for } \bar{s}=38, \bar{s}'=30 \\ 2\mu_2 & \text{for } \bar{s}=38, \bar{s}'=7 \\ \mu_3 & \text{for } \bar{s}=38, \bar{s}'=4 \\ \mu_1 & \text{for } \bar{s}=40, \bar{s}'=31 \\ 2\mu_2 & \text{for } \bar{s}=40, \bar{s}'=8 \\ \mu_4 & \text{for } \bar{s}=40, \bar{s}'=4 \\ \mu_1 & \text{for } \bar{s}=41, \bar{s}'=32 \\ 2\mu_2 & \text{for } \bar{s}=41, \bar{s}'=9 \\ \mu_4 & \text{for } \bar{s}=41, \bar{s}'=5 \\ \mu_1 & \text{for } \bar{s}=42, \bar{s}'=33 \\ 2\mu_2 & \text{for } \bar{s}=42, \bar{s}'=10 \\ \mu_1 & \text{for } \bar{s}=46, \bar{s}'=34 \\ \mu_2 & \text{for } \bar{s}=46, \bar{s}'=15 \\ 2\mu_3 & \text{for } \bar{s}=46, \bar{s}'=8 \\ \mu_4 & \text{for } \bar{s}=46, \bar{s}'=7 \\ \mu_1 & \text{for } \bar{s}=48, \bar{s}'=36 \\ \mu_2 & \text{for } \bar{s}=48, \bar{s}'=17 \\ \mu_3 & \text{for } \bar{s}=48, \bar{s}'=11 \\ \mu_4 & \text{for } \bar{s}=48, \bar{s}'=9 \end{cases}$$

$$P_{a_3}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} \mu_1 & \text{for } \bar{s}=6, \bar{s}'=34 \\ \mu_2 & \text{for } \bar{s}=6, \bar{s}'=15 \\ 2\mu_3 & \text{for } \bar{s}=6, \bar{s}'=8 \\ \mu_4 & \text{for } \bar{s}=6, \bar{s}'=7 \\ \mu_1 & \text{for } \bar{s}=7, \bar{s}'=35 \\ \mu_2 & \text{for } \bar{s}=7, \bar{s}'=16 \\ 2\mu_3 & \text{for } \bar{s}=7, \bar{s}'=9 \\ \mu_1 & \text{for } \bar{s}=8, \bar{s}'=36 \\ \mu_2 & \text{for } \bar{s}=8, \bar{s}'=17 \end{cases}$$

$$P_{a_3}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases}
\mu_3 & \text{for } \bar{s}=8,\bar{s}'=11\\
\mu_4 & \text{for } \bar{s}=8,\bar{s}'=9\\
\mu_1 & \text{for } \bar{s}=9,\bar{s}'=37\\
\mu_2 & \text{for } \bar{s}=9,\bar{s}'=20\\
\mu_3 & \text{for } \bar{s}=9,\bar{s}'=12\\
\mu_4 & \text{for } \bar{s}=9,\bar{s}'=10\\
\mu_1 & \text{for } \bar{s}=10,\bar{s}'=38\\
\mu_2 & \text{for } \bar{s}=10,\bar{s}'=21\\
\mu_3 & \text{for } \bar{s}=10,\bar{s}'=13\\
\mu_1 & \text{for } \bar{s}=11,\bar{s}'=39\\
\mu_2 & \text{for } \bar{s}=11,\bar{s}'=19\\
\mu_4 & \text{for } \bar{s}=11,\bar{s}'=12\\
\mu_1 & \text{for } \bar{s}=12,\bar{s}'=40\\
\mu_2 & \text{for } \bar{s}=12,\bar{s}'=22\\
\mu_4 & \text{for } \bar{s}=12,\bar{s}'=13\\
\mu_1 & \text{for } \bar{s}=13,\bar{s}'=41\\
\mu_2 & \text{for } \bar{s}=13,\bar{s}'=23\\
\mu_4 & \text{for } \bar{s}=13,\bar{s}'=14\\
\mu_1 & \text{for } \bar{s}=15,\bar{s}'=43\\
2\mu_3 & \text{for } \bar{s}=15,\bar{s}'=17\\
\mu_4 & \text{for } \bar{s}=15,\bar{s}'=16\\
\mu_1 & \text{for } \bar{s}=16,\bar{s}'=44\\
2\mu_3 & \text{for } \bar{s}=16,\bar{s}'=20\\
\mu_1 & \text{for } \bar{s}=17,\bar{s}'=45\\
\mu_3 & \text{for } \bar{s}=17,\bar{s}'=19\\
\mu_4 & \text{for } \bar{s}=17,\bar{s}'=18\\
\mu_1 & \text{for } \bar{s}=18,\bar{s}'=6\\
2\mu_3 & \text{for } \bar{s}=18,\bar{s}'=22\\
\mu_1 & \text{for } \bar{s}=19,\bar{s}'=47\\
\mu_4 & \text{for } \bar{s}=19,\bar{s}'=22\\
\mu_1 & \text{for } \bar{s}=20,\bar{s}'=6\\
2\mu_3 & \text{for } \bar{s}=20,\bar{s}'=22\\
\mu_1 & \text{for } \bar{s}=21,\bar{s}'=7\\
2\mu_3 & \text{for } \bar{s}=21,\bar{s}'=23
\end{cases}$$

$$P_{a_3}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases}
\mu_1 & \text{for } \bar{s}=22, \bar{s}'=8 \\
\mu_3 & \text{for } \bar{s}=22, \bar{s}'=25 \\
\mu_4 & \text{for } \bar{s}=22, \bar{s}'=23 \\
\mu_1 & \text{for } \bar{s}=25, \bar{s}'=11 \\
\mu_4 & \text{for } \bar{s}=25, \bar{s}'=26 \\
\mu_2 & \text{for } \bar{s}=34, \bar{s}'=43 \\
2\mu_3 & \text{for } \bar{s}=34, \bar{s}'=36 \\
\mu_4 & \text{for } \bar{s}=34, \bar{s}'=35 \\
\mu_2 & \text{for } \bar{s}=35, \bar{s}'=44 \\
2\mu_3 & \text{for } \bar{s}=35, \bar{s}'=37 \\
\mu_2 & \text{for } \bar{s}=36, \bar{s}'=45 \\
\mu_3 & \text{for } \bar{s}=36, \bar{s}'=39 \\
\mu_4 & \text{for } \bar{s}=36, \bar{s}'=37 \\
\mu_2 & \text{for } \bar{s}=37, \bar{s}'=6 \\
\mu_3 & \text{for } \bar{s}=37, \bar{s}'=40 \\
\mu_4 & \text{for } \bar{s}=37, \bar{s}'=38 \\
\mu_2 & \text{for } \bar{s}=39, \bar{s}'=47 \\
\mu_4 & \text{for } \bar{s}=39, \bar{s}'=40 \\
\mu_2 & \text{for } \bar{s}=41, \bar{s}'=9 \\
\mu_4 & \text{for } \bar{s}=41, \bar{s}'=42 \\
2\mu_3 & \text{for } \bar{s}=43, \bar{s}'=45 \\
\mu_4 & \text{for } \bar{s}=43, \bar{s}'=44 \\
2\mu_3 & \text{for } \bar{s}=44, \bar{s}'=46 \\
\mu_3 & \text{for } \bar{s}=45, \bar{s}'=47 \\
\mu_4 & \text{for } \bar{s}=45, \bar{s}'=46 \\
\mu_1 & \text{for } \bar{s}=46, \bar{s}'=34 \\
\mu_2 & \text{for } \bar{s}=46, \bar{s}'=15 \\
2\mu_3 & \text{for } \bar{s}=46, \bar{s}'=8 \\
\mu_4 & \text{for } \bar{s}=46, \bar{s}'=7 \\
\mu_4 & \text{for } \bar{s}=47, \bar{s}'=48 \\
\mu_1 & \text{for } \bar{s}=48, \bar{s}'=36 \\
\mu_2 & \text{for } \bar{s}=48, \bar{s}'=17 \\
\mu_3 & \text{for } \bar{s}=48, \bar{s}'=11 \\
\mu_4 & \text{for } \bar{s}=48, \bar{s}'=9
\end{cases}$$

With the above set up, we ran the Policy Iteration algorithm to show that inequality (3.2) holds and hence the stated policy is optimal.

(Theorem 4.5) □

4.2.2 Flexible server between first and second stations

Consider a tandem line with four stations and a dedicated server at each station. Assume a flexible server exists which can only move between the first and second stations. Theorem 4.6 describes the optimal policy for equal workloads.

Theorem 4.6. *For equal workloads, the optimal policy prioritizes clearing blocking. It performs hand-off and allocates the flexible server to the first station, whenever possible. Otherwise the flexible server is assigned to the second station.*

The optimal policy appears to be rate independent, the policy is as given in Theorem 4.6 for the rates we have tried (i.e. the following service rate vectors = (1,1,1,1), (1,5,10,20), (1,10,5,20), (1,5,20,10), (1,10,20,5), (10,1,5,20), (10,1,20,5), (10,5,1,20), (10,5,20,1), (10,20,1,5), (20,1,5,10)). However we have not attempted to verify if the policy is generally rate independent.

4.2.3 Flexible server between third and fourth stations

Consider a tandem line with four stations and a dedicated server at each station. Assume a flexible server exists which can only move between the third and fourth stations. Theorem 4.7 describes the optimal policy for equal workloads.

Theorem 4.7. *For equal workloads, the optimal policy prioritizes clearing blocking. It performs hand-off and allocates the flexible server to the third station, whenever*

possible. Otherwise the flexible server is assigned to the fourth station.

The optimal policy appears to be rate independent, the policy is as given in Theorem 4.7 for the rates we have tried (i.e. the following service rate vectors = (1,1,1,1), (1,5,10,20), (1,10,5,20), (1,5,20,10), (1,10,20,5), (10,1,5,20), (10,1,20,5), (10,5,1,20), (10,5,20,1), (10,20,1,5), (20,1,5,10)). However we have not attempted to verify if the policy is generally rate independent.

The policies introduced in Section 4.2 are similar to the policy introduced in Section 3.4 for a fully flexible server. All of the policies perform hand-off such that the flexible server is freed to send it to upstream stations. They also clear blocked servers, if any exist.

Based on our observations in this chapter, the optimal policies under constrained flexibility have a similar structure to the optimal policies under full flexibility. Also as expected, the throughput improvement under constrained flexibility is less compared to full flexibility. The trade-off between the cost of making servers flexible (fully or constrained) and the throughput improvement can be used to decide if the flexible servers should be constrained or not.

So far we have explored allocation policies within a single line. In the next chapter, we extend the problem to involve two parallel identical lines. We let flexible servers move between corresponding stations and propose dynamic allocation policies. We will study how the throughput improves, when such flexibility is introduced.

Chapter 5

Dynamic server allocation for parallel zero-buffer tandem queues with dedicated and vertical flexible servers

5.1 Introduction

Consider two parallel tandem queues where each of them follows the model introduced in Section 1.1. Assume $M \geq N$ dedicated servers and a number of vertical flexible servers are present. Vertical flexible servers are servers that can move between corresponding stations of the two parallel tandem queues. Let $\bar{F} = (f_1, \ldots, f_i, \ldots, f_N)$ represent the allocation vector of the vertical flexible servers, where f_i is the number of vertical flexible servers available for the pair of stations i in the two tandem lines.

Assume that dedicated servers are already assigned to the stations. We are interested in *determining the dynamic assignment policy for vertical flexible servers that maximizes the long-run average throughput.* For simplicity, we assume the travel times for vertical flexible servers to move between stations are negligible.

Similar to Section 3.1, we introduce the concept of "hand-off" for vertical flexible servers. Hand-off happens when a vertical flexible server passes the job it is serving to a dedicated server in the same station. As jobs must always be allocated to a server, "hand-off" is performed only if the target dedicated server is free.

We have not been able to find any work that targets a problem similar to that stated above. In the following, we provide a number of previous works in the same general area as our problem of interest.

A rich literature exists on the routing of jobs among parallel queues. In a system of parallel queues, the shortest queue policy has often proved to be optimal. As an example, Winston (1977) showed that the shortest queue policy stochastically minimizes the discounted cost over any time horizon for identical exponential servers with infinite queues. Researchers have considered admission and routing controls in parallel tandem queues to minimize the number of customers in the system (e.g. Hordijk and Koole (1992)), blocking probability (e.g. Spicer and Ziedins (2006)), or holding costs due to loss (e.g. Sheu and Ziedins (2010)).

Simultaneous optimization of routing and allocation controls has also been studied. For example, Ahn and Lewis (2012) consider a parallel queueing system serving two types of jobs, and propose routing and allocation policies to meet the challenge of demand variability. Jobs can be rerouted after they are assigned. Servers can work collaboratively and routing costs are assumed. They seek optimal policies under

discounted or long-run average cost criteria. They state that in the super-additive collaboration case, jobs should never be routed away from the lower cost queue and that the optimal policy is a non-idling priority rule based on the holding costs. In the sub-additive collaboration case, the optimal policy need not prioritize one station.

Different researchers have worked on the classic scheduling problem of optimally allocating a server to multiple competing jobs (stored in separate queues). Buyukkoc et al. (1985) aim to minimize the weighted holding costs. They provide a policy which serves jobs preemptively in decreasing order of the product of service rate and holding cost rate (called a list policy). Tassiulas and Ephremides (1993) consider a similar setting, but assume that queues may or may not be connected to the server at different time slots (e.g. a radio network with mobile users). The server is allocated to only one queue in each time slot. With infinite buffers, they report that the policy serving the longest connected queue stabilizes the system. When arrivals and service statistics of all queues are the same, they propose a policy that minimizes the delay and maximizes the throughput. Duenyas and Van Oyen (1995) consider the problem of allocating a single server to a number of parallel queues (they also study other topologies). Each queue represents a class of jobs and has a holding cost rate. Their objective is to minimize the expected total cost due to waiting jobs, the switching of the server, and set-up efforts. They partially characterize the optimal policy and provide a scheduling policy based on a simple heuristic.

Hajek (1982) studies the control issue of a generic Markovian network, including a case with two parallel stations, with external arrivals/departures to/from the stations. He describes optimal controls by switching curves to govern routing and service priorities to minimize the finite horizon and long run average costs. Value iteration

and other dynamic programming techniques are employed.

Yoneyama and Ishii (1997) consider a parallel two-stage tandem queuing system with no buffer spaces. A job served at the first stage can enter both of the second stage channels. Service times are dependent according to a bivariate exponential distribution. They derive the throughput of the system, using a matrix-geometric approach.

Comparing our work with the above related works, we focus on allocation control whereas these works concentrate on routing control (e.g. Winston (1977), Hordijk and Koole (1992), Spicer and Ziedins (2006), and Sheu and Ziedins (2010)). In addition we aim to maximize the throughput, while most of these works target cost minimization (e.g. Buyukkoc et al. (1985), Tassiulas and Ephremides (1993), Duenyas and Van Oyen (1995), and Hajek (1982)). Moreover, these works study parallel queues rather than parallel tandem queues.

This chapter is organized as follows. In Section 5.2, the optimality of non-idling and hand-off for parallel tandem lines with two stations is shown. In Section 5.3 we use Markov Decision Process theory to derive the optimal policy for parallel tandem lines with two stations. We further show how to employ the Policy Iteration algorithm to construct the optimal policy. In Section 5.4, we provide heuristics for allocation policies for systems of arbitrary size and with heterogeneous mean service times. Section 5.5 studies the effects of increasing the number of vertical flexible servers on the throughput.

5.2 Optimal policy properties

In this section, we introduce and prove some properties of the optimal policy for systems with two tandem lines, each with two stations. The properties concern performing hand-off and idling vertical flexible servers. The properties provide insights for expressing the optimal policy. A policy resulting in the maximum possible throughput is called optimal. In Theorems 5.1 and 5.2, there is one dedicated server at each station and one vertical flexible server for the second stations. Alternatively, we can consider the vertical flexible server is at the first stations (and state and prove the properties with a similar mechanism). We concentrate on the case where the vertical flexible server is at the second stations, so that it can help clear blocking (which has the greatest impact on throughput).

Theorem 5.1. *The optimal policy performs hand-off whenever possible.*

Proof: There is only one scenario under which hand-off is possible: a dedicated server becomes free in the same station where a vertical flexible server is busy. We investigate this scenario in the following.

Let ω and ω' be policies that send the vertical flexible server to the other line, only if there is blocking in that line and no blocking in the current line, and always perform hand-off when possible. The only exception is that ω' does not perform a hand-off at one of the times when hand-off is possible. Let the system run until it reaches the situation illustrated in Figure 5.1 at t_0. We track the departures from the second stations.

In all figures in the proof: the top label is the time stamp of the configuration, ovals represent stations, the space to the left of the vertical line shows ω and the space

to the right shows ω', labels inside the stations are job numbers, in each station the top job is served by a dedicated server and the bottom job by the vertical flexible server, labels below the stations show the residual service time of a job appearing in the same station under both policies when residual service times are different. If a number inside a station is aligned lower than other numbers, it implies that the job is served by the vertical flexible server and the dedicated server is free. A server marked by the 'b' superscript means that the server is blocked.

The notation used in this proof is the same as that introduced in Subsection 3.2.2.

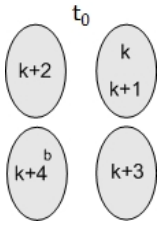

Figure 5.1: Theorem 5.1 - The initial case

Case 1: Assume $d^2_{k,t_0} < \min\{f^2_{k+1,t_0}, d^2_{k+3,t_0}\}$. Let $t_1 = d^2_{k,t_0}$. At $t_0 + t_1$, ω performs a hand-off and sends the vertical flexible server to the second line to admit job $k+4$ to the second station. But ω' leaves the vertical flexible server in the first tandem line, idling the dedicated server in the second station of the first tandem line, and job $k+4$ remains blocked in the first station of the second tandem line. Figure 5.2 illustrates this case.

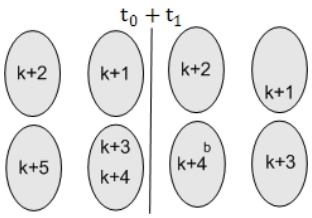

Figure 5.2: Theorem 5.1 - Case 1

Case 1.1: Assume $X^2_{k+1,t_0+t_1} < \min\{d^2_{k+3,t_0+t_1}, f^{2,\omega}_{k+4,t_0+t_1}\}$. Let $t_2 = X^2_{k+1,t_0+t_1}$. This assumption results in the situation illustrated in Figure 5.3 at time $t_0+t_1+t_2$. Let $t_3 = f^{2,\omega}_{k+4,t_0+t_1} = f^{2,\omega'}_{k+4,t_0+t_1} - t_2$.

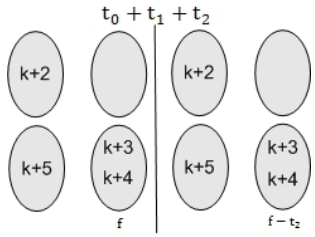

Figure 5.3: Theorem 5.1 - Case 1.1

Case 1.1.1: Assume $t_3 < d^2_{k+3,t_0+t_1+t_2}$. This assumption leads to the situation illustrated in Figure 5.4.

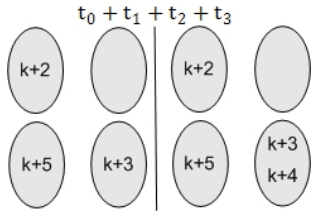

Figure 5.4: Theorem 5.1 - Case 1.1.1

Case 1.1.1.1: Assume $d^{2,\omega'}_{k+4,t_0+t_1+t_2+t_3} \leq \min\{d^2_{k+3,t_0+t_1+t_2}, d^1_{k+5,t_0+t_1+t_2}\}$. The system becomes the same under both policies.

Case 1.1.1.2: Assume $d^2_{k+3,t_0+t_1+t_2} < \min\{d^{2,\omega'}_{k+4,t_0+t_1+t_2+t_3}, d^1_{k+5,t_0+t_1+t_2}\}$. The system under ω' still needs to serve job $k+4$ whereas under ω, job $k+4$ has been served. This leaves no room for ω' to serve future jobs sooner than ω.

Case 1.1.1.3: Assume $d^1_{k+5,t_0+t_1+t_2} < \min\{d^{2,\omega'}_{k+4,t_0+t_1+t_2+t_3}, d^2_{k+3,t_0+t_1+t_2}\}$.

Job $k+5$ goes to the second station under ω but gets blocked under ω'. If job $k+4$ departs, the systems hold the same jobs with less residual service time for job $k+5$ under ω. If job $k+5$ departs from the second station under ω, this policy has completed two more jobs compared to ω'. If job $k+3$ departs, under ω' job $k+5$ goes to the second station with larger residual service time compared to ω. No matter which of the above departures occur sooner, it is apparent that there is no way that ω' completes jobs sooner than ω.

Case 1.1.1.4: Assume $d^2_{k+3,t_0+t_1+t_2} = d^{2,\omega'}_{k+4,t_0+t_1+t_2+t_3} < d^1_{k+5,t_0+t_1+t_2}$. The systems become the same under both policies.

Case 1.1.1.5: Assume $d^2_{k+3,t_0+t_1+t_2} = d^1_{k+5,t_0+t_1+t_2} < d^{2,\omega'}_{k+4,t_0+t_1+t_2+t_3}$. This is similar to Case 1.1.1.

Case 1.1.1.6: Assume $d^1_{k+5,t_0+t_1+t_2} = d^{2,\omega'}_{k+4,t_0+t_1+t_2+t_3} < d^2_{k+3,t_0+t_1+t_2}$. The systems become the same under both policies.

Case 1.1.1.7: Assume $d^2_{k+3,t_0+t_1+t_2} = d^{2,\omega'}_{k+4,t_0+t_1+t_2+t_3} = d^1_{k+5,t_0+t_1+t_2}$. The systems become the same under both policies.

Case 1.1.2: Assume $d^2_{k+3,t_0+t_1+t_2} \leq t_3$. The vertical flexible server hands off job $k+4$ to the dedicated server in the second station under both policies with less residual service time under ω. Hence, ω' will not be able to complete jobs sooner than ω.

Case 1.2: Assume $d^2_{k+3,t_0+t_1} < \min\{X^2_{k+1,t_0+t_1}, f^{2,\omega}_{k+4,t_0+t_1}\}$. Let $t_2 = d^2_{k+3,t_0+t_1}$. This assumption results in the situation illustrated in Figure 5.5 at time $t_0+t_1+t_2$. If jobs $k+1$ or $k+2$ depart, the systems serve the same jobs under both policies

with less residual service time for job $k+4$ under ω'. If job $k+4$ departs, the systems become the same under both policies. If job $k+5$ departs, ω assigns the vertical flexible server to the second tandem line to admit the job to the second station. But under ω', the vertical flexible server is busy in the first tandem line. In all these cases, it is apparent that there is no way that ω' completes jobs sooner than ω.

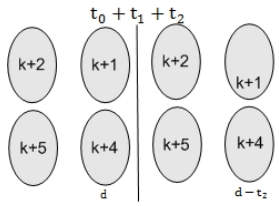

Figure 5.5: Theorem 5.1 - Case 1.2

Case 1.3: Assume $f^{2,\omega}_{k+4,t_0+t_1} < \min\{X^2_{k+1,t_0+t_1}, d^2_{k+3,t_0+t_1}\}$. Let $t_2 = f^{2,\omega}_{k+4,t_0+t_1}$. This assumption results in the situation illustrated in Figure 5.6 at time $t_0+t_1+t_2$. Let $t_3 = X^2_{k+1,t_0+t_1+t_2}$.

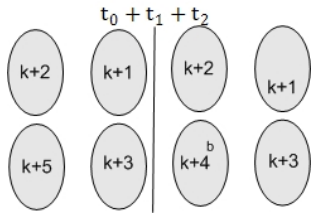

Figure 5.6: Theorem 5.1 - Case 1.3

Case 1.3.1: Assume $t_3 < d^2_{k+3,t_0+t_1+t_2}$. Under ω', the vertical flexible server is sent to the second station of the second line to admit job $k+4$. This is while ω has already served job $k+4$ and has a free vertical flexible server

at the second station. Therefore ω' will not complete jobs sooner than ω.

Case 1.3.2: Assume $d^2_{k+3,t_0+t_1+t_2} \leq t_3$. Under ω', job $k+4$ is sent to the second station and job $k+5$ is admitted to the first station. This is while ω has already served job $k+4$ and job $k+5$ has less residual service time. Therefore ω' will not depart jobs sooner than ω.

Case 1.4: Assume $X^2_{k+1,t_0+t_1} = d^2_{k+3,t_0+t_1} < f^{2,\omega}_{k+4,t_0+t_1}$. This is similar to Case 1.1.2.

Case 1.5: Assume $X^2_{k+1,t_0+t_1} = f^{2,\omega}_{k+4,t_0+t_1} < d^2_{k+3,t_0+t_1}$. This is similar to Case 1.1.1.

Case 1.6: Assume $d^2_{k+3,t_0+t_1} = X^2_{k+1,t_0+t_1} < f^{2,\omega}_{k+4,t_0+t_1}$. This is similar to Case 1.1.

Case 1.7: Assume $d^2_{k+3,t_0+t_1} = f^{2,\omega}_{k+4,t_0+t_1} < X^2_{k+1,t_0+t_1}$. This is similar to Case 1.3.2.

Case 1.8: Assume $X^2_{k+1,t_0+t_1} = d^2_{k+3,t_0+t_1} = f^{2,\omega}_{k+4,t_0+t_1}$. Under ω', job $k+4$ is sent to the second station and job $k+5$ is admitted to the first station. This is while ω has already served job $k+4$ and job $k+5$ has less residual service time. Therefore ω' will not complete jobs sooner than ω.

Case 2: Now assume $d^2_{k,t_0} \geq \min\{f^2_{k+1,t_0}, d^2_{k+3,t_0}\}$. The systems are the same under both policies. As a hand-off opportunity is not introduced, there is no need to analyse this case.

(Theorem 5.1) □

In Theorem 5.1 we proved that an optimal policy uses hand-off whenever possible.

Now in Theorem 5.2, we aim to show that an optimal policy avoids idling the vertical flexible server. In the proof of Theorem 5.2, we will use the results of Theorem 5.1, i.e. the two policies used in the proof perform a hand-off whenever possible. Having said that, we only claim non-idling to be a property of an optimal policy if hand-off can be employed. Otherwise there might be cases where idling leads to better results.

For example, assume at time t_0, under the idling policy, jobs $k+2$ and k are in the first and second stations of the first line, respectively, and jobs $k+3$ and $k+1$ are in the first and second stations of the second line, respectively. Job $k+3$ is blocked and the flexible server is idling. Under the non-idling policy, jobs $k+2$ and k are in the first and second stations of the first line, respectively, and jobs $k+4$ and $k+1$ are in the first and second stations of the second line, respectively. The flexible server is serving job $k+3$ at the second station of the second line. Let $d^1_{k+2,t_0} < t_1$ and $d^2_{k+1,t_0} < t_1$ and assume t_1 is less than the residual service times of other jobs (jobs $k+3$ and $k+4$). At time t_0+t_1, the idling policy sends the flexible server to the second station of the first line to admit job $k+2$ and subsequently admits job $k+5$ at the first station of the first line. However under the non-idling policy at t_0+t_1, job $k+2$ becomes blocked. Further we can assume jobs $k+2$ and $k+5$ depart at the same time and sooner than other jobs (jobs $k+1$, $k+3$, and $k+4$). Hence earlier departures from the first and second stations occur, under the idling policy. As a direct result of such a mechanism, our proof of Theorem 5.2 below would not go through.

Theorem 5.2. *The optimal policy is non-idling.*

Proof: Let π and π' be two non-idling policies with the only exception that π' idles the vertical flexible server once for only t time units starting from t_0. Both are policies

that send the vertical flexible server to the other line, only if there is blocking in that line and no blocking in the current line, and always perform hand-off when possible.

With the above description for the policies, we claim it is sufficient to show that the property holds only for one tandem line. The reason is that if π is able to assign a job to the vertical flexible server, it can be inferred that there is a dedicated busy server in the station (where the flexible server is) and an arrival occurs at the station between t_0 and $t_0 + t$ (as if no arrivals occur, policies π and π' behave exactly the same). However this means that π' introduces blocking by idling the vertical flexible server. Therefore, by definition, the vertical flexible server is not sent to the other tandem line even if blocking exists, as the current tandem line also has blocking. Hence we only need to consider the current tandem line.

Notice that the way that the above policy is defined does not affect the proof by introducing extra assumptions. Specifically, the decision to stay in the current tandem line in case of occurrence of blocking in both tandem lines is supported by the symmetry in the system. The total throughput of a pair of stations is of interest and therefore when counting the departures from a pair of stations, it does not matter which station has had more departures.

The only scenario to consider is when an arrival (between t_0 and t_0+t under π) occurs to a station, where a busy dedicated server exists. Three cases are possible within a tandem line.

Case 1: The dedicated server (in the same station as the vertical flexible server) becomes free before $t_0 + t$. In this case the vertical flexible server hands off its job to the dedicated server. This is similar to Case T3.1-2.1 (case T3.1 *num* refers to Case *num* of Theorem 3.1).

Case 2: The vertical flexible server becomes free before $t_0 + t$. This is similar to Case T3.1-1.3.

Case 3: No departures occur before $t_0 + t$ and π' admits the next job at $t_0 + t$. This is similar to Case T3.1-1.1.

(Theorem 5.2) $\square$

Comparing the proofs of Theorems 5.1 and 5.2 with Theorems 3.1 and 3.2, there are two major differences: 1) Unlike a single line, it is not possible to state the problem in a similar form of induction. The reason is that for a single line the optimal act (hand-off or non-idling) results in an immediate reward (i.e. an earlier departure). But in parallel lines no immediate reward is necessarily gained. 2) The more important point is that in a single line, using the flexible server in a station means that it can not be used in other stations (at the same time). Therefore, gaining higher throughput in that station is achieved at the cost of possible lower throughput in the other stations. However, in parallel lines, using the vertical flexible server in a station does not directly affect the other stations, as the server can not be used in the other stations. Therefore not using the vertical flexible server does not introduce any benefit to the system. This fact makes it trivial that employing the vertical flexible server is necessary for optimality.

5.3 Markov Decision Process model

We use a Markov Decision Process (MDP) to model the above problem. We use the same notation as introduced in Section 3.3.

A state $s \in S$ is denoted by a tuple

$$s = (x_{11}, y_{11}, \ \ldots, \ x_{1i}, y_{1i}, \ \ldots, \ x_{1N}, \ x_{21}, y_{21}, \ \ldots, \ x_{2i}, y_{2i}, \ \ldots, \ x_{2N}) \tag{5.1}$$

where x_{ki} is the number of busy servers at station i in line k and y_{ki} is the number of blocked servers at station j in line k, $k \in \{1, 2\}$. Note that y_{kN} need not be included, as they are always zero.

The total number of vertical flexible servers is given by $|\bar{F}| = \sum_{i=1}^{N} f_i$. The size of the action space would become $\prod_{i=1}^{N} f_i^3$ for an idling policy and $\prod_{i=1}^{N} f_i^2$ for a non-idling policy. An action $a \in A$ is denoted by a tuple

$$a = (a_{11}, a_{21}, \ \ldots, \ a_{1j}, a_{2j}, \ \ldots, \ a_{1N}, a_{2N})$$

where a_{kj} is the number of vertical flexible servers assigned to station i in line $k \in \{1, 2\}$ and $0 \leq a_{1j} + a_{2j} \leq f_j$.

In inequality (3.2), given that $s \xrightarrow{a} s' \xrightarrow{P_a} s''$, $r(s, a) = x'_{1N}\mu_{1N} + x'_{2N}\mu_{2N}$ where x'_{1N} and x'_{2N} appear in the representation of state s' according to (5.1) and $\frac{\mu_{1N}+\mu_{2N}}{q} = P_a(s, s'')$. Note that s' is also a function of s and a.

We implemented the Policy Iteration algorithm in MATLAB. In the algorithm, the initial policy entries are set to actions $a \in A_s$ that prioritize the first tandem line over the second tandem line.

5.3.1 Policy Iteration algorithm results for $N = 2$

We used the described MDP to model a number of configurations with $N = 2$ and a single dedicated server per station. Further we applied the Policy Iteration algorithm on the model and determined optimal policies. Unlike the horizontal flexibility situation, policies are rate dependent when vertical flexibility is considered (for $N = 2$). In the following, we provide MDP details only for $\bar{F} = (1, 1)$. The details are similar for $\bar{F} = (1, 0)$ and $\bar{F} = (0, 1)$. All the theorems and conjectures in this section are verified through MDP analysis.

In the following, we report the optimal policy for several service rates. By *fast line* we mean $\mu_{k1} = \mu_{k2} > \mu_{k'1} = \mu_{k'2}$, where k is the fast line and k' is the other line. We tried $\mu_{k1} = 4$ and $\mu_{k'1} = 1$. By *first fast station* we mean $\mu_{k1} > \mu_{k2} = \mu_{k'1} = \mu_{k'2}$, given that k is the line including the fast station and k' is the other line. We tried $\mu_{k1} = 4$ and $\mu_{k'1} = 1$. By *first pair fast* we mean $\mu_{k1} = \mu_{k'1} > \mu_{k2} = \mu_{k'2}$. We tried $\mu_{k1} = 4$ and $\mu_{k2} = 1$.

Tandem queues with $\bar{F} = (1, 0)$

Consider $\bar{F} = (1, 0)$, i.e. one vertical flexible server is available for the first pair of stations. The optimal policies, divided based on service rates, are presented in the following. A line has a lower likelihood of blocking in a station, if it has more free dedicated servers at the next station, compared to the other line.

Theorem 5.3. *When rates are equal, the optimal policy assigns the vertical flexible server to the line that does not have a blocked server or has a lower likelihood of blocking, as a result of allocating the vertical flexible server. If the lines are the same in this respect, it arbitrarily chooses the line.*

Conjecture 5.1. *When a line is faster (i.e. has stations with smaller mean service times), the optimal policy assigns the vertical flexible server to the fast line, even if it increases the likelihood of blocking.*

Conjecture 5.2. *When the first station is faster in one of the lines (i.e. the first station of one line has a smaller mean service time), the optimal policy assigns the vertical flexible server to the line with the slow first station to help this station. The only exceptions are the* $(1,0,0,1,0,1)$ *and* $(1,0,0,0,1,1)$ *states, where the policy chooses the line with the fast station.*

Conjecture 5.3. *When the second station is faster in one of the lines (i.e. the second station of one line has a smaller mean service time), the optimal policy assigns the vertical flexible server to the line with the fast second station, even if the first station is blocked.*

Tandem queues with $\bar{F} = (0,1)$

Consider $\bar{F} = (0,1)$, i.e. one vertical flexible server is available for the second pair of stations. The optimal policies are presented in the following.

Theorem 5.4. *When rates are equal, the optimal policy assigns the vertical flexible server to one line and sends it to the other line, only if the other line has blocking and the original line has no blocking.*

Conjecture 5.4. *When a line is faster, the optimal policy assigns the vertical flexible server to the fast line, except for the* $(1,0,0,0,1,1)$ *state.*

Conjecture 5.5. *When the first station is faster in one of the lines, the optimal policy assigns the vertical flexible server to the line with the fast station, except for the* $(1,0,0,0,1,1)$ *and* $(1,0,1,0,1,1)$ *states.*

Conjecture 5.6. *When the second station is faster in one of the lines, the optimal policy is the same as the policy described in Theorem 5.4.*

Tandem queues with $\bar{F} = (1, 1)$

Consider $\bar{F} = (1, 1)$, i.e. one vertical flexible server is available for each pair of stations. The optimal policies are as follows.

Theorem 5.5. *When rates are equal, the optimal policy distributes the vertical flexible servers. For the second pair, the policy chooses a line if it has a blocked server and the other line does not. Otherwise, it chooses any of the lines. For the first pair, the policy allocates the vertical flexible server to the line not including the vertical flexible server in its second station. The only exception occurs when the vertical flexible server at the first pair is available and there is no blocking in either of the lines. In this case, it synchronizes and assigns both vertical flexible servers to a single line (chosen randomly).*

Proof. To be able to construct the discrete-time MDP, we choose the following normalization factor: $q = 2(\mu_{11} + \mu_{12} + \mu_{21} + \mu_{22})$. Here,

$$A = \{a_{11}, a_{12}, a_{21}, a_{22}\}$$

where $a_{11} = (1, 1, 0, 0)$, $a_{12} = (1, 0, 0, 1)$, $a_{21} = (0, 1, 1, 0)$, and $a_{22} = (0, 0, 1, 1)$, and

$S = \{(2,0,2,1,0,1),(2,0,2,1,0,0),(2,0,2,0,1,1),(2,0,1,1,0,2),(2,0,1,1,0,1),$
$(2,0,1,1,0,0),(2,0,1,0,1,2),(2,0,1,0,1,1),(2,0,0,1,0,2),(2,0,0,1,0,1),(2,0,0,1,0,0),$
$(2,0,0,0,1,2),(2,0,0,0,1,1),(1,1,2,1,0,1),(1,1,2,1,0,0),(1,1,2,0,1,1),(1,1,1,1,0,2),$
$(1,1,1,1,0,1),(1,1,1,1,0,0),(1,1,1,0,1,2),(1,1,1,0,1,1),(1,0,2,2,0,1),(1,0,2,2,0,0),$
$(1,0,2,1,1,1),(1,0,2,1,0,1),(1,0,2,1,0,0),(1,0,2,0,2,1),(1,0,2,0,1,1),(1,0,1,2,0,2),$

$(1,0,1,2,0,1),(1,0,1,2,0,0),(1,0,1,1,1,2),(1,0,1,1,1,1),(1,0,1,1,0,2),(1,0,1,1,0,1),$
$(1,0,1,1,0,0),(1,0,1,0,2,2),(1,0,1,0,2,1),(1,0,1,0,1,2),(1,0,1,0,1,1),(1,0,0,2,0,2),$
$(1,0,0,2,0,1),(1,0,0,2,0,0),(1,0,0,1,1,2),(1,0,0,1,1,1),(1,0,0,1,0,1),(1,0,0,0,2,2),$
$(1,0,0,0,2,1),(1,0,0,0,1,1),(0,2,2,1,0,1),(0,2,2,1,0,0),(0,2,2,0,1,1),(0,2,1,1,0,2),$
$(0,2,1,1,0,1),(0,2,1,1,0,0),(0,2,1,0,1,2),(0,2,1,0,1,1),(0,1,2,2,0,1),(0,1,2,2,0,0),$
$(0,1,2,1,1,1),(0,1,2,1,0,1),(0,1,2,0,1,2),(0,1,2,0,1,1),(0,1,1,2,0,2),(0,1,1,2,0,1),$
$(0,1,1,2,0,0),(0,1,1,1,1,2),(0,1,1,1,1,1),(0,1,1,1,0,2),(0,1,1,1,0,1),(0,1,1,1,0,0),$
$(0,1,1,0,2,2),(0,1,1,0,2,1),(0,1,1,0,1,2),(1,0,0,1,0,2),(1,0,0,1,0,0)\}$

Let $\bar{s}$ represent the index of state $s \in S$, according to the order represented above.

$$A_{\bar{s}} = \begin{cases} \{a_{11}\} & \text{for } \bar{s} = 1,2,3,14,15,16,50,51,52 \\ \{a_{12}\} & \text{for } \bar{s} = 4,7,9,12,17,20,53,56 \\ \{a_{21}\} & \text{for } \bar{s} = 22,23,24,27,58,59,60,62 \\ \{a_{22}\} & \text{for } \bar{s} = 29,32,37,41,44,47,64,67,72 \\ \{a_{11},a_{12}\} & \text{for } \bar{s} = 5,6,8,10,11,13 \\ \{a_{11},a_{21}\} & \text{for } \bar{s} = 25,26,28,61,63,74,75 \\ \{a_{12},a_{22}\} & \text{for } \bar{s} = 34,39 \\ \{a_{21},a_{22}\} & \text{for } \bar{s} = 30,31,42,43,65,66 \\ \{a_{11},a_{12},a_{21}\} & \text{for } \bar{s} = 18,19,21,54,55,57 \\ \{a_{12},a_{21},a_{22}\} & \text{for } \bar{s} = 33,38,45,46,48,68,69,73 \\ \{a_{11},a_{12},a_{21},a_{22}\} & \text{for } \bar{s} = 35,36,40,49,70,71,76 \end{cases}$$

Our candidate optimal policy d_0 is:

$$d_0(\bar{s}) = \begin{cases} \{a_{11}\} & \text{for } \bar{s} = 1,2,3,5,6,10,11,14,15,16,18,46,50,51,52 \\ \{a_{12}\} & \text{for } \bar{s} = 4,7,8,9,12,13,17,20,21,33,34,38,39,40,45,48,49,53,56,68, \\ & 69,73,74,75 \\ \{a_{21}\} & \text{for } \bar{s} = 19,22,23,24,25,26,27,28,30,31,35,36,42,43,54,55,57,58,59, \\ & 60,61,62,63,65,66,70,71,76 \\ \{a_{22}\} & \text{for } \bar{s} = 29,32,37,41,44,47,64,67,72 \end{cases}$$

$$r_{d_0}(\bar{s}) = \frac{1}{q}\begin{cases} 0 & \text{for } \bar{s} = 11, 43, 76 \\ 2\mu_{12} + \mu_{22} & \text{for } \bar{s} = 1, 3, 14, 16, 18, 22, 24, 25, 27, 28, 50, 52, 54, 57, 58, 60, \\ & 61, 62, 63, 65, 66, 70 \\ \mu_{12} + \mu_{22} & \text{for } \bar{s} = 5, 6, 10, 30, 35, 36 \\ 2\mu_{12} & \text{for } \bar{s} = 2, 15, 19, 23, 26, 51, 55, 59, 71 \\ \mu_{12} + 2\mu_{22} & \text{for } \bar{s} = 4, 7, 8, 13, 17, 20, 21, 29, 32, 33, 37, 38, 39, 40, 53, 56, \\ & 64, 67, 68, 69, 72, 73, 74 \\ \mu_{12} & \text{for } \bar{s} = 31 \\ 2\mu_{22} & \text{for } \bar{s} = 9, 12, 41, 44, 45, 47, 48, 49, 75 \\ \mu_{22} & \text{for } \bar{s} = 42, 46 \end{cases}$$

$$P_{a_{11}}(\bar{s}, \bar{s}') = \frac{1}{q}\begin{cases} 2\mu_{11} & \text{for } \bar{s} = 1, \bar{s}' = 14 \\ 2\mu_{12} & \text{for } \bar{s} = 1, \bar{s}' = 5 \\ \mu_{21} & \text{for } \bar{s} = 1, \bar{s}' = 3 \\ \mu_{22} & \text{for } \bar{s} = 1, \bar{s}' = 2 \\ 2\mu_{11} & \text{for } \bar{s} = 2, \bar{s}' = 15 \\ 2\mu_{12} & \text{for } \bar{s} = 2, \bar{s}' = 6 \\ \mu_{21} & \text{for } \bar{s} = 2, \bar{s}' = 1 \\ 2\mu_{11} & \text{for } \bar{s} = 3, \bar{s}' = 16 \\ 2\mu_{12} & \text{for } \bar{s} = 3, \bar{s}' = 8 \\ \mu_{22} & \text{for } \bar{s} = 3, \bar{s}' = 1 \\ 2\mu_{11} & \text{for } \bar{s} = 5, \bar{s}' = 25 \\ \mu_{12} & \text{for } \bar{s} = 5, \bar{s}' = 10 \\ \mu_{21} & \text{for } \bar{s} = 5, \bar{s}' = 8 \\ \mu_{22} & \text{for } \bar{s} = 5, \bar{s}' = 6 \\ 2\mu_{11} & \text{for } \bar{s} = 6, \bar{s}' = 26 \\ \mu_{12} & \text{for } \bar{s} = 6, \bar{s}' = 11 \\ \mu_{21} & \text{for } \bar{s} = 6, \bar{s}' = 5 \\ 2\mu_{11} & \text{for } \bar{s} = 8, \bar{s}' = 28 \\ \mu_{12} & \text{for } \bar{s} = 8, \bar{s}' = 13 \\ \mu_{22} & \text{for } \bar{s} = 8, \bar{s}' = 5 \end{cases}$$

$$P_{a_{11}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_{11} & \text{for } \bar{s}=10,\bar{s}'=35 \\ \mu_{21} & \text{for } \bar{s}=10,\bar{s}'=13 \\ \mu_{22} & \text{for } \bar{s}=10,\bar{s}'=11 \\ 2\mu_{11} & \text{for } \bar{s}=11,\bar{s}'=36 \\ \mu_{21} & \text{for } \bar{s}=11,\bar{s}'=10 \\ 2\mu_{11} & \text{for } \bar{s}=13,\bar{s}'=40 \\ \mu_{22} & \text{for } \bar{s}=13,\bar{s}'=10 \\ \mu_{11} & \text{for } \bar{s}=14,\bar{s}'=50 \\ 2\mu_{12} & \text{for } \bar{s}=14,\bar{s}'=18 \\ \mu_{21} & \text{for } \bar{s}=14,\bar{s}'=16 \\ \mu_{22} & \text{for } \bar{s}=14,\bar{s}'=15 \\ \mu_{11} & \text{for } \bar{s}=15,\bar{s}'=51 \\ 2\mu_{12} & \text{for } \bar{s}=15,\bar{s}'=19 \\ \mu_{21} & \text{for } \bar{s}=15,\bar{s}'=14 \\ \mu_{11} & \text{for } \bar{s}=16,\bar{s}'=52 \\ 2\mu_{12} & \text{for } \bar{s}=16,\bar{s}'=21 \\ \mu_{22} & \text{for } \bar{s}=16,\bar{s}'=14 \\ 2\mu_{11} & \text{for } \bar{s}=18,\bar{s}'=14 \\ 2\mu_{12} & \text{for } \bar{s}=18,\bar{s}'=5 \\ \mu_{21} & \text{for } \bar{s}=18,\bar{s}'=3 \\ \mu_{22} & \text{for } \bar{s}=18,\bar{s}'=2 \\ 2\mu_{11} & \text{for } \bar{s}=19,\bar{s}'=15 \\ 2\mu_{12} & \text{for } \bar{s}=19,\bar{s}'=6 \\ \mu_{21} & \text{for } \bar{s}=19,\bar{s}'=1 \\ 2\mu_{11} & \text{for } \bar{s}=21,\bar{s}'=16 \\ 2\mu_{12} & \text{for } \bar{s}=21,\bar{s}'=8 \\ \mu_{22} & \text{for } \bar{s}=21,\bar{s}'=1 \\ 2\mu_{11} & \text{for } \bar{s}=25,\bar{s}'=14 \\ 2\mu_{12} & \text{for } \bar{s}=25,\bar{s}'=5 \\ \mu_{21} & \text{for } \bar{s}=25,\bar{s}'=3 \\ \mu_{22} & \text{for } \bar{s}=25,\bar{s}'=2 \\ 2\mu_{11} & \text{for } \bar{s}=26,\bar{s}'=15 \\ 2\mu_{12} & \text{for } \bar{s}=26,\bar{s}'=6 \\ \mu_{21} & \text{for } \bar{s}=26,\bar{s}'=1 \end{cases}$$

$$P_{a_{11}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_{11} & \text{for } \bar{s}=28, \bar{s}'=16 \\ 2\mu_{12} & \text{for } \bar{s}=28, \bar{s}'=8 \\ \mu_{22} & \text{for } \bar{s}=28, \bar{s}'=1 \\ 2\mu_{11} & \text{for } \bar{s}=35, \bar{s}'=25 \\ \mu_{12} & \text{for } \bar{s}=35, \bar{s}'=10 \\ \mu_{21} & \text{for } \bar{s}=35, \bar{s}'=8 \\ \mu_{22} & \text{for } \bar{s}=35, \bar{s}'=6 \\ 2\mu_{11} & \text{for } \bar{s}=36, \bar{s}'=26 \\ \mu_{12} & \text{for } \bar{s}=36, \bar{s}'=11 \\ \mu_{21} & \text{for } \bar{s}=36, \bar{s}'=5 \\ 2\mu_{11} & \text{for } \bar{s}=40, \bar{s}'=38 \\ \mu_{12} & \text{for } \bar{s}=40, \bar{s}'=13 \\ \mu_{22} & \text{for } \bar{s}=40, \bar{s}'=5 \\ 2\mu_{11} & \text{for } \bar{s}=46, \bar{s}'=35 \\ \mu_{21} & \text{for } \bar{s}=46, \bar{s}'=13 \\ \mu_{22} & \text{for } \bar{s}=46, \bar{s}'=11 \\ 2\mu_{11} & \text{for } \bar{s}=49, \bar{s}'=40 \\ \mu_{22} & \text{for } \bar{s}=49, \bar{s}'=10 \\ 2\mu_{11} & \text{for } \bar{s}=50, \bar{s}'=54 \\ \mu_{21} & \text{for } \bar{s}=50, \bar{s}'=52 \\ \mu_{22} & \text{for } \bar{s}=50, \bar{s}'=51 \\ 2\mu_{12} & \text{for } \bar{s}=51, \bar{s}'=55 \\ \mu_{21} & \text{for } \bar{s}=51, \bar{s}'=50 \\ 2\mu_{12} & \text{for } \bar{s}=52, \bar{s}'=57 \\ \mu_{22} & \text{for } \bar{s}=52, \bar{s}'=50 \\ \mu_{11} & \text{for } \bar{s}=54, \bar{s}'=50 \\ 2\mu_{12} & \text{for } \bar{s}=54, \bar{s}'=18 \\ \mu_{21} & \text{for } \bar{s}=54, \bar{s}'=16 \\ \mu_{22} & \text{for } \bar{s}=54, \bar{s}'=15 \\ \mu_{11} & \text{for } \bar{s}=55, \bar{s}'=51 \\ 2\mu_{12} & \text{for } \bar{s}=55, \bar{s}'=19 \\ \mu_{21} & \text{for } \bar{s}=55, \bar{s}'=14 \end{cases}$$

$$P_{a_{11}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} \mu_{11} & \text{for } \bar{s}=57, \bar{s}'=52 \\ 2\mu_{12} & \text{for } \bar{s}=57, \bar{s}'=21 \\ \mu_{22} & \text{for } \bar{s}=57, \bar{s}'=14 \\ \mu_{11} & \text{for } \bar{s}=61, \bar{s}'=50 \\ 2\mu_{12} & \text{for } \bar{s}=61, \bar{s}'=18 \\ \mu_{21} & \text{for } \bar{s}=61, \bar{s}'=16 \\ \mu_{22} & \text{for } \bar{s}=61, \bar{s}'=15 \\ \mu_{11} & \text{for } \bar{s}=63, \bar{s}'=52 \\ 2\mu_{12} & \text{for } \bar{s}=63, \bar{s}'=21 \\ \mu_{22} & \text{for } \bar{s}=63, \bar{s}'=14 \\ 2\mu_{11} & \text{for } \bar{s}=70, \bar{s}'=14 \\ 2\mu_{12} & \text{for } \bar{s}=70, \bar{s}'=5 \\ \mu_{21} & \text{for } \bar{s}=70, \bar{s}'=3 \\ \mu_{22} & \text{for } \bar{s}=70, \bar{s}'=2 \\ 2\mu_{11} & \text{for } \bar{s}=71, \bar{s}'=15 \\ 2\mu_{12} & \text{for } \bar{s}=71, \bar{s}'=6 \\ \mu_{21} & \text{for } \bar{s}=71, \bar{s}'=1 \\ 2\mu_{11} & \text{for } \bar{s}=76, \bar{s}'=36 \\ \mu_{21} & \text{for } \bar{s}=76, \bar{s}'=10 \end{cases}$$

$$P_{a_{12}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_{11} & \text{for } \bar{s}=4,\bar{s}'=17 \\ 2\mu_{12} & \text{for } \bar{s}=4,\bar{s}'=9 \\ \mu_{21} & \text{for } \bar{s}=4,\bar{s}'=7 \\ \mu_{22} & \text{for } \bar{s}=4,\bar{s}'=5 \\ 2\mu_{11} & \text{for } \bar{s}=5,\bar{s}'=18 \\ \mu_{12} & \text{for } \bar{s}=5,\bar{s}'=10 \\ \mu_{21} & \text{for } \bar{s}=5,\bar{s}'=4 \\ \mu_{22} & \text{for } \bar{s}=5,\bar{s}'=6 \\ 2\mu_{11} & \text{for } \bar{s}=6,\bar{s}'=19 \\ \mu_{12} & \text{for } \bar{s}=6,\bar{s}'=11 \\ \mu_{21} & \text{for } \bar{s}=6,\bar{s}'=5 \\ 2\mu_{11} & \text{for } \bar{s}=7,\bar{s}'=20 \\ \mu_{12} & \text{for } \bar{s}=7,\bar{s}'=12 \\ 2\mu_{22} & \text{for } \bar{s}=7,\bar{s}'=8 \\ 2\mu_{11} & \text{for } \bar{s}=8,\bar{s}'=17 \\ \mu_{12} & \text{for } \bar{s}=8,\bar{s}'=9 \\ \mu_{21} & \text{for } \bar{s}=8,\bar{s}'=7 \\ 2\mu_{22} & \text{for } \bar{s}=8,\bar{s}'=5 \\ 2\mu_{11} & \text{for } \bar{s}=9,\bar{s}'=34 \\ \mu_{21} & \text{for } \bar{s}=9,\bar{s}'=12 \\ 2\mu_{22} & \text{for } \bar{s}=9,\bar{s}'=10 \\ 2\mu_{11} & \text{for } \bar{s}=10,\bar{s}'=35 \\ \mu_{21} & \text{for } \bar{s}=10,\bar{s}'=9 \\ \mu_{22} & \text{for } \bar{s}=10,\bar{s}'=11 \\ 2\mu_{11} & \text{for } \bar{s}=11,\bar{s}'=36 \\ \mu_{21} & \text{for } \bar{s}=11,\bar{s}'=10 \\ 2\mu_{11} & \text{for } \bar{s}=12,\bar{s}'=39 \\ 2\mu_{22} & \text{for } \bar{s}=12,\bar{s}'=13 \\ 2\mu_{11} & \text{for } \bar{s}=13,\bar{s}'=34 \\ \mu_{21} & \text{for } \bar{s}=13,\bar{s}'=12 \\ 2\mu_{22} & \text{for } \bar{s}=13,\bar{s}'=10 \\ \mu_{11} & \text{for } \bar{s}=17,\bar{s}'=53 \\ \mu_{12} & \text{for } \bar{s}=17,\bar{s}'=34 \end{cases}$$

$$P_{a_{12}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases}
\mu_{21} & \text{for } \bar{s}=17, \bar{s}'=20 \\
\mu_{22} & \text{for } \bar{s}=17, \bar{s}'=18 \\
\mu_{11} & \text{for } \bar{s}=18, \bar{s}'=54 \\
\mu_{12} & \text{for } \bar{s}=18, \bar{s}'=35 \\
\mu_{21} & \text{for } \bar{s}=18, \bar{s}'=17 \\
\mu_{22} & \text{for } \bar{s}=18, \bar{s}'=19 \\
\mu_{11} & \text{for } \bar{s}=19, \bar{s}'=55 \\
\mu_{12} & \text{for } \bar{s}=19, \bar{s}'=36 \\
\mu_{21} & \text{for } \bar{s}=19, \bar{s}'=18 \\
\mu_{11} & \text{for } \bar{s}=20, \bar{s}'=56 \\
\mu_{12} & \text{for } \bar{s}=20, \bar{s}'=39 \\
2\mu_{22} & \text{for } \bar{s}=20, \bar{s}'=21 \\
\mu_{11} & \text{for } \bar{s}=21, \bar{s}'=53 \\
\mu_{12} & \text{for } \bar{s}=21, \bar{s}'=34 \\
\mu_{21} & \text{for } \bar{s}=21, \bar{s}'=20 \\
\mu_{22} & \text{for } \bar{s}=21, \bar{s}'=18 \\
2\mu_{11} & \text{for } \bar{s}=33, \bar{s}'=17 \\
\mu_{12} & \text{for } \bar{s}=33, \bar{s}'=9 \\
\mu_{21} & \text{for } \bar{s}=33, \bar{s}'=7 \\
2\mu_{22} & \text{for } \bar{s}=33, \bar{s}'=5 \\
2\mu_{11} & \text{for } \bar{s}=34, \bar{s}'=17 \\
\mu_{12} & \text{for } \bar{s}=34, \bar{s}'=9 \\
\mu_{21} & \text{for } \bar{s}=34, \bar{s}'=7 \\
2\mu_{22} & \text{for } \bar{s}=34, \bar{s}'=5 \\
2\mu_{11} & \text{for } \bar{s}=35, \bar{s}'=18 \\
\mu_{12} & \text{for } \bar{s}=35, \bar{s}'=10 \\
\mu_{21} & \text{for } \bar{s}=35, \bar{s}'=4 \\
\mu_{22} & \text{for } \bar{s}=35, \bar{s}'=6 \\
2\mu_{11} & \text{for } \bar{s}=36, \bar{s}'=19 \\
\mu_{12} & \text{for } \bar{s}=36, \bar{s}'=11 \\
\mu_{21} & \text{for } \bar{s}=36, \bar{s}'=5 \\
2\mu_{11} & \text{for } \bar{s}=38, \bar{s}'=20 \\
\mu_{12} & \text{for } \bar{s}=38, \bar{s}'=12 \\
2\mu_{22} & \text{for } \bar{s}=38, \bar{s}'=8
\end{cases}$$

$$P_{a_{12}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_{11} & \text{for } \bar{s}=39, \bar{s}'=20 \\ \mu_{12} & \text{for } \bar{s}=39, \bar{s}'=12 \\ 2\mu_{22} & \text{for } \bar{s}=39, \bar{s}'=8 \\ 2\mu_{11} & \text{for } \bar{s}=40, \bar{s}'=17 \\ \mu_{12} & \text{for } \bar{s}=40, \bar{s}'=9 \\ \mu_{21} & \text{for } \bar{s}=40, \bar{s}'=7 \\ 2\mu_{22} & \text{for } \bar{s}=40, \bar{s}'=5 \\ 2\mu_{11} & \text{for } \bar{s}=45, \bar{s}'=34 \\ \mu_{21} & \text{for } \bar{s}=45, \bar{s}'=12 \\ 2\mu_{22} & \text{for } \bar{s}=45, \bar{s}'=10 \\ 2\mu_{11} & \text{for } \bar{s}=46, \bar{s}'=35 \\ \mu_{21} & \text{for } \bar{s}=46, \bar{s}'=9 \\ \mu_{22} & \text{for } \bar{s}=46, \bar{s}'=11 \\ 2\mu_{11} & \text{for } \bar{s}=48, \bar{s}'=39 \\ 2\mu_{22} & \text{for } \bar{s}=48, \bar{s}'=13 \\ 2\mu_{11} & \text{for } \bar{s}=49, \bar{s}'=34 \\ \mu_{21} & \text{for } \bar{s}=49, \bar{s}'=12 \\ 2\mu_{22} & \text{for } \bar{s}=49, \bar{s}'=10 \\ \mu_{12} & \text{for } \bar{s}=53, \bar{s}'=69 \\ \mu_{21} & \text{for } \bar{s}=53, \bar{s}'=56 \\ 2\mu_{22} & \text{for } \bar{s}=53, \bar{s}'=54 \\ \mu_{12} & \text{for } \bar{s}=54, \bar{s}'=70 \\ \mu_{21} & \text{for } \bar{s}=54, \bar{s}'=53 \\ \mu_{22} & \text{for } \bar{s}=54, \bar{s}'=55 \\ \mu_{12} & \text{for } \bar{s}=55, \bar{s}'=71 \\ \mu_{21} & \text{for } \bar{s}=55, \bar{s}'=54 \\ \mu_{12} & \text{for } \bar{s}=56, \bar{s}'=74 \\ 2\mu_{22} & \text{for } \bar{s}=56, \bar{s}'=57 \\ \mu_{12} & \text{for } \bar{s}=57, \bar{s}'=69 \\ \mu_{21} & \text{for } \bar{s}=57, \bar{s}'=56 \\ 2\mu_{22} & \text{for } \bar{s}=57, \bar{s}'=54 \\ \mu_{11} & \text{for } \bar{s}=68, \bar{s}'=53 \\ \mu_{12} & \text{for } \bar{s}=68, \bar{s}'=34 \\ \mu_{21} & \text{for } \bar{s}=68, \bar{s}'=20 \\ \mu_{22} & \text{for } \bar{s}=68, \bar{s}'=18 \end{cases}$$

$$P_{a_{12}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases}
\mu_{11} & \text{for } \bar{s}=69, \bar{s}'=53\\
\mu_{12} & \text{for } \bar{s}=69, \bar{s}'=34\\
\mu_{21} & \text{for } \bar{s}=69, \bar{s}'=20\\
\mu_{22} & \text{for } \bar{s}=69, \bar{s}'=18\\
\mu_{11} & \text{for } \bar{s}=70, \bar{s}'=54\\
\mu_{12} & \text{for } \bar{s}=70, \bar{s}'=35\\
\mu_{21} & \text{for } \bar{s}=70, \bar{s}'=17\\
\mu_{22} & \text{for } \bar{s}=70, \bar{s}'=19\\
\mu_{11} & \text{for } \bar{s}=71, \bar{s}'=55\\
\mu_{12} & \text{for } \bar{s}=71, \bar{s}'=36\\
\mu_{21} & \text{for } \bar{s}=71, \bar{s}'=18\\
\mu_{11} & \text{for } \bar{s}=73, \bar{s}'=56\\
\mu_{12} & \text{for } \bar{s}=73, \bar{s}'=39\\
2\mu_{22} & \text{for } \bar{s}=73, \bar{s}'=21\\
\mu_{11} & \text{for } \bar{s}=74, \bar{s}'=56\\
\mu_{12} & \text{for } \bar{s}=74, \bar{s}'=39\\
2\mu_{22} & \text{for } \bar{s}=74, \bar{s}'=21\\
2\mu_{11} & \text{for } \bar{s}=75, \bar{s}'=34\\
\mu_{21} & \text{for } \bar{s}=75, \bar{s}'=12\\
2\mu_{22} & \text{for } \bar{s}=75, \bar{s}'=10\\
2\mu_{11} & \text{for } \bar{s}=76, \bar{s}'=36\\
\mu_{21} & \text{for } \bar{s}=76, \bar{s}'=10
\end{cases}$$

$$P_{a_{21}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases}
\mu_{11} & \text{for } \bar{s}=18, \bar{s}'=58\\
2\mu_{12} & \text{for } \bar{s}=18, \bar{s}'=30\\
2\mu_{21} & \text{for } \bar{s}=18, \bar{s}'=24\\
\mu_{22} & \text{for } \bar{s}=18, \bar{s}'=23\\
\mu_{11} & \text{for } \bar{s}=19, \bar{s}'=59\\
2\mu_{12} & \text{for } \bar{s}=19, \bar{s}'=31\\
2\mu_{21} & \text{for } \bar{s}=19, \bar{s}'=25\\
\mu_{11} & \text{for } \bar{s}=21, \bar{s}'=60\\
2\mu_{12} & \text{for } \bar{s}=21, \bar{s}'=33
\end{cases}$$

$$P_{a_{21}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases}
\mu_{21} & \text{for } \bar{s}=21, \bar{s}'=27\\
\mu_{22} & \text{for } \bar{s}=21, \bar{s}'=25\\
\mu_{11} & \text{for } \bar{s}=22, \bar{s}'=58\\
2\mu_{12} & \text{for } \bar{s}=22, \bar{s}'=30\\
2\mu_{21} & \text{for } \bar{s}=22, \bar{s}'=24\\
\mu_{22} & \text{for } \bar{s}=22, \bar{s}'=23\\
\mu_{11} & \text{for } \bar{s}=23, \bar{s}'=59\\
2\mu_{12} & \text{for } \bar{s}=23, \bar{s}'=31\\
2\mu_{21} & \text{for } \bar{s}=23, \bar{s}'=25\\
\mu_{11} & \text{for } \bar{s}=24, \bar{s}'=60\\
2\mu_{12} & \text{for } \bar{s}=24, \bar{s}'=33\\
\mu_{21} & \text{for } \bar{s}=24, \bar{s}'=27\\
\mu_{22} & \text{for } \bar{s}=24, \bar{s}'=25\\
\mu_{11} & \text{for } \bar{s}=25, \bar{s}'=58\\
2\mu_{12} & \text{for } \bar{s}=25, \bar{s}'=30\\
2\mu_{21} & \text{for } \bar{s}=25, \bar{s}'=24\\
\mu_{22} & \text{for } \bar{s}=25, \bar{s}'=23\\
\mu_{11} & \text{for } \bar{s}=26, \bar{s}'=59\\
2\mu_{12} & \text{for } \bar{s}=26, \bar{s}'=31\\
2\mu_{21} & \text{for } \bar{s}=26, \bar{s}'=25\\
\mu_{11} & \text{for } \bar{s}=27, \bar{s}'=62\\
2\mu_{12} & \text{for } \bar{s}=27, \bar{s}'=38\\
\mu_{22} & \text{for } \bar{s}=27, \bar{s}'=28\\
\mu_{11} & \text{for } \bar{s}=28, \bar{s}'=60\\
2\mu_{12} & \text{for } \bar{s}=28, \bar{s}'=33\\
\mu_{21} & \text{for } \bar{s}=28, \bar{s}'=27\\
\mu_{22} & \text{for } \bar{s}=28, \bar{s}'=25\\
\mu_{11} & \text{for } \bar{s}=30, \bar{s}'=22\\
\mu_{12} & \text{for } \bar{s}=30, \bar{s}'=42\\
2\mu_{21} & \text{for } \bar{s}=30, \bar{s}'=33\\
\mu_{22} & \text{for } \bar{s}=30, \bar{s}'=31\\
\mu_{11} & \text{for } \bar{s}=31, \bar{s}'=23\\
\mu_{12} & \text{for } \bar{s}=31, \bar{s}'=43\\
2\mu_{21} & \text{for } \bar{s}=31, \bar{s}'=35
\end{cases}$$

$$P_{a_{21}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases}
\mu_{11} & \text{for } \bar{s}=33, \bar{s}'=24\\
\mu_{12} & \text{for } \bar{s}=33, \bar{s}'=45\\
\mu_{21} & \text{for } \bar{s}=33, \bar{s}'=38\\
\mu_{22} & \text{for } \bar{s}=33, \bar{s}'=35\\
\mu_{11} & \text{for } \bar{s}=35, \bar{s}'=22\\
\mu_{12} & \text{for } \bar{s}=35, \bar{s}'=42\\
2\mu_{21} & \text{for } \bar{s}=35, \bar{s}'=33\\
\mu_{22} & \text{for } \bar{s}=35, \bar{s}'=31\\
\mu_{11} & \text{for } \bar{s}=36, \bar{s}'=23\\
\mu_{12} & \text{for } \bar{s}=36, \bar{s}'=43\\
2\mu_{21} & \text{for } \bar{s}=36, \bar{s}'=35\\
\mu_{11} & \text{for } \bar{s}=38, \bar{s}'=27\\
\mu_{12} & \text{for } \bar{s}=38, \bar{s}'=48\\
\mu_{22} & \text{for } \bar{s}=38, \bar{s}'=40\\
\mu_{11} & \text{for } \bar{s}=40, \bar{s}'=24\\
\mu_{12} & \text{for } \bar{s}=40, \bar{s}'=45\\
\mu_{21} & \text{for } \bar{s}=40, \bar{s}'=38\\
\mu_{22} & \text{for } \bar{s}=40, \bar{s}'=35\\
\mu_{11} & \text{for } \bar{s}=42, \bar{s}'=30\\
2\mu_{21} & \text{for } \bar{s}=42, \bar{s}'=45\\
\mu_{22} & \text{for } \bar{s}=42, \bar{s}'=43\\
\mu_{11} & \text{for } \bar{s}=43, \bar{s}'=31\\
2\mu_{21} & \text{for } \bar{s}=43, \bar{s}'=46\\
\mu_{11} & \text{for } \bar{s}=45, \bar{s}'=33\\
\mu_{21} & \text{for } \bar{s}=45, \bar{s}'=48\\
\mu_{22} & \text{for } \bar{s}=45, \bar{s}'=46\\
\mu_{11} & \text{for } \bar{s}=46, \bar{s}'=30\\
2\mu_{21} & \text{for } \bar{s}=46, \bar{s}'=45\\
\mu_{22} & \text{for } \bar{s}=46, \bar{s}'=43\\
\mu_{11} & \text{for } \bar{s}=48, \bar{s}'=38\\
\mu_{22} & \text{for } \bar{s}=48, \bar{s}'=49\\
\mu_{11} & \text{for } \bar{s}=49, \bar{s}'=33\\
\mu_{21} & \text{for } \bar{s}=49, \bar{s}'=48\\
\mu_{22} & \text{for } \bar{s}=49, \bar{s}'=46
\end{cases}$$

$$P_{a_{21}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_{12} & \text{for } \bar{s}=54, \bar{s}'=65 \\ 2\mu_{21} & \text{for } \bar{s}=54, \bar{s}'=60 \\ \mu_{22} & \text{for } \bar{s}=54, \bar{s}'=59 \\ 2\mu_{12} & \text{for } \bar{s}=55, \bar{s}'=66 \\ 2\mu_{21} & \text{for } \bar{s}=55, \bar{s}'=61 \\ 2\mu_{12} & \text{for } \bar{s}=57, \bar{s}'=68 \\ \mu_{21} & \text{for } \bar{s}=57, \bar{s}'=62 \\ \mu_{22} & \text{for } \bar{s}=57, \bar{s}'=61 \\ 2\mu_{12} & \text{for } \bar{s}=58, \bar{s}'=65 \\ 2\mu_{21} & \text{for } \bar{s}=58, \bar{s}'=60 \\ \mu_{22} & \text{for } \bar{s}=58, \bar{s}'=59 \\ 2\mu_{12} & \text{for } \bar{s}=59, \bar{s}'=66 \\ 2\mu_{21} & \text{for } \bar{s}=59, \bar{s}'=61 \\ 2\mu_{12} & \text{for } \bar{s}=60, \bar{s}'=68 \\ \mu_{21} & \text{for } \bar{s}=60, \bar{s}'=62 \\ \mu_{22} & \text{for } \bar{s}=60, \bar{s}'=61 \\ 2\mu_{12} & \text{for } \bar{s}=61, \bar{s}'=65 \\ 2\mu_{21} & \text{for } \bar{s}=61, \bar{s}'=60 \\ \mu_{22} & \text{for } \bar{s}=61, \bar{s}'=59 \\ 2\mu_{12} & \text{for } \bar{s}=62, \bar{s}'=73 \\ \mu_{22} & \text{for } \bar{s}=62, \bar{s}'=63 \\ 2\mu_{12} & \text{for } \bar{s}=63, \bar{s}'=68 \\ \mu_{21} & \text{for } \bar{s}=63, \bar{s}'=62 \\ \mu_{22} & \text{for } \bar{s}=63, \bar{s}'=61 \\ \mu_{11} & \text{for } \bar{s}=65, \bar{s}'=58 \\ 2\mu_{12} & \text{for } \bar{s}=65, \bar{s}'=30 \\ 2\mu_{21} & \text{for } \bar{s}=65, \bar{s}'=24 \\ \mu_{22} & \text{for } \bar{s}=65, \bar{s}'=23 \\ \mu_{11} & \text{for } \bar{s}=66, \bar{s}'=59 \\ 2\mu_{12} & \text{for } \bar{s}=66, \bar{s}'=31 \\ 2\mu_{21} & \text{for } \bar{s}=66, \bar{s}'=25 \\ \mu_{11} & \text{for } \bar{s}=68, \bar{s}'=60 \\ 2\mu_{12} & \text{for } \bar{s}=68, \bar{s}'=33 \\ \mu_{21} & \text{for } \bar{s}=68, \bar{s}'=27 \\ \mu_{22} & \text{for } \bar{s}=68, \bar{s}'=25 \end{cases}$$

$$P_{a_{21}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} \mu_{11} & \text{for } \bar{s}=70,\bar{s}'=58 \\ 2\mu_{12} & \text{for } \bar{s}=70,\bar{s}'=30 \\ 2\mu_{21} & \text{for } \bar{s}=70,\bar{s}'=24 \\ \mu_{22} & \text{for } \bar{s}=70,\bar{s}'=23 \\ \mu_{11} & \text{for } \bar{s}=71,\bar{s}'=59 \\ 2\mu_{12} & \text{for } \bar{s}=71,\bar{s}'=31 \\ 2\mu_{21} & \text{for } \bar{s}=71,\bar{s}'=25 \\ \mu_{11} & \text{for } \bar{s}=73,\bar{s}'=62 \\ 2\mu_{12} & \text{for } \bar{s}=73,\bar{s}'=38 \\ \mu_{22} & \text{for } \bar{s}=73,\bar{s}'=28 \\ \mu_{11} & \text{for } \bar{s}=76,\bar{s}'=31 \\ \mu_{21} & \text{for } \bar{s}=76,\bar{s}'=46 \end{cases}$$

$$P_{a_{22}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} \mu_{11} & \text{for } \bar{s}=29,\bar{s}'=64 \\ \mu_{12} & \text{for } \bar{s}=29,\bar{s}'=41 \\ 2\mu_{21} & \text{for } \bar{s}=29,\bar{s}'=32 \\ 2\mu_{22} & \text{for } \bar{s}=29,\bar{s}'=30 \\ \mu_{11} & \text{for } \bar{s}=30,\bar{s}'=65 \\ \mu_{12} & \text{for } \bar{s}=30,\bar{s}'=42 \\ 2\mu_{21} & \text{for } \bar{s}=30,\bar{s}'=34 \\ \mu_{22} & \text{for } \bar{s}=30,\bar{s}'=31 \\ \mu_{11} & \text{for } \bar{s}=31,\bar{s}'=66 \\ \mu_{12} & \text{for } \bar{s}=31,\bar{s}'=43 \\ 2\mu_{21} & \text{for } \bar{s}=31,\bar{s}'=35 \\ \mu_{11} & \text{for } \bar{s}=32,\bar{s}'=67 \\ \mu_{12} & \text{for } \bar{s}=32,\bar{s}'=44 \\ \mu_{21} & \text{for } \bar{s}=32,\bar{s}'=37 \\ 2\mu_{22} & \text{for } \bar{s}=32,\bar{s}'=33 \\ \mu_{11} & \text{for } \bar{s}=33,\bar{s}'=64 \\ \mu_{12} & \text{for } \bar{s}=33,\bar{s}'=41 \\ 2\mu_{21} & \text{for } \bar{s}=33,\bar{s}'=32 \\ 2\mu_{22} & \text{for } \bar{s}=33,\bar{s}'=30 \\ \mu_{11} & \text{for } \bar{s}=34,\bar{s}'=64 \\ \mu_{12} & \text{for } \bar{s}=34,\bar{s}'=41 \end{cases}$$

$$P_{a_{22}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} 2\mu_{21} & \text{for } \bar{s}=34, \bar{s}'=32 \\ 2\mu_{22} & \text{for } \bar{s}=34, \bar{s}'=30 \\ \mu_{11} & \text{for } \bar{s}=35, \bar{s}'=65 \\ \mu_{12} & \text{for } \bar{s}=35, \bar{s}'=42 \\ 2\mu_{21} & \text{for } \bar{s}=35, \bar{s}'=34 \\ \mu_{22} & \text{for } \bar{s}=35, \bar{s}'=31 \\ \mu_{11} & \text{for } \bar{s}=36, \bar{s}'=66 \\ \mu_{12} & \text{for } \bar{s}=36, \bar{s}'=43 \\ 2\mu_{21} & \text{for } \bar{s}=36, \bar{s}'=35 \\ \mu_{11} & \text{for } \bar{s}=37, \bar{s}'=72 \\ \mu_{12} & \text{for } \bar{s}=37, \bar{s}'=47 \\ 2\mu_{22} & \text{for } \bar{s}=37, \bar{s}'=38 \\ \mu_{11} & \text{for } \bar{s}=38, \bar{s}'=67 \\ \mu_{12} & \text{for } \bar{s}=38, \bar{s}'=44 \\ \mu_{21} & \text{for } \bar{s}=38, \bar{s}'=37 \\ 2\mu_{22} & \text{for } \bar{s}=38, \bar{s}'=33 \\ \mu_{11} & \text{for } \bar{s}=39, \bar{s}'=67 \\ \mu_{12} & \text{for } \bar{s}=39, \bar{s}'=44 \\ \mu_{21} & \text{for } \bar{s}=39, \bar{s}'=37 \\ 2\mu_{22} & \text{for } \bar{s}=39, \bar{s}'=33 \\ \mu_{11} & \text{for } \bar{s}=40, \bar{s}'=64 \\ \mu_{12} & \text{for } \bar{s}=40, \bar{s}'=41 \\ 2\mu_{21} & \text{for } \bar{s}=40, \bar{s}'=32 \\ 2\mu_{22} & \text{for } \bar{s}=40, \bar{s}'=30 \\ \mu_{11} & \text{for } \bar{s}=41, \bar{s}'=29 \\ 2\mu_{21} & \text{for } \bar{s}=41, \bar{s}'=44 \\ 2\mu_{22} & \text{for } \bar{s}=41, \bar{s}'=42 \\ \mu_{11} & \text{for } \bar{s}=42, \bar{s}'=30 \\ 2\mu_{21} & \text{for } \bar{s}=42, \bar{s}'=75 \\ \mu_{22} & \text{for } \bar{s}=42, \bar{s}'=43 \\ \mu_{11} & \text{for } \bar{s}=43, \bar{s}'=31 \\ 2\mu_{21} & \text{for } \bar{s}=43, \bar{s}'=46 \\ \mu_{11} & \text{for } \bar{s}=44, \bar{s}'=32 \\ 2\mu_{21} & \text{for } \bar{s}=44, \bar{s}'=47 \\ 2\mu_{22} & \text{for } \bar{s}=44, \bar{s}'=45 \end{cases}$$

$$P_{a_{22}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases}
\mu_{11} & \text{for } \bar{s}=45, \bar{s}'=29\\
2\mu_{21} & \text{for } \bar{s}=45, \bar{s}'=44\\
2\mu_{22} & \text{for } \bar{s}=45, \bar{s}'=42\\
\mu_{11} & \text{for } \bar{s}=46, \bar{s}'=30\\
2\mu_{21} & \text{for } \bar{s}=46, \bar{s}'=75\\
\mu_{22} & \text{for } \bar{s}=46, \bar{s}'=43\\
\mu_{11} & \text{for } \bar{s}=47, \bar{s}'=37\\
2\mu_{22} & \text{for } \bar{s}=47, \bar{s}'=48\\
\mu_{11} & \text{for } \bar{s}=48, \bar{s}'=32\\
2\mu_{21} & \text{for } \bar{s}=48, \bar{s}'=47\\
2\mu_{22} & \text{for } \bar{s}=48, \bar{s}'=45\\
\mu_{11} & \text{for } \bar{s}=49, \bar{s}'=29\\
2\mu_{21} & \text{for } \bar{s}=49, \bar{s}'=44\\
2\mu_{22} & \text{for } \bar{s}=49, \bar{s}'=42\\
\mu_{11} & \text{for } \bar{s}=64, \bar{s}'=29\\
2\mu_{21} & \text{for } \bar{s}=64, \bar{s}'=67\\
2\mu_{22} & \text{for } \bar{s}=64, \bar{s}'=65\\
\mu_{12} & \text{for } \bar{s}=65, \bar{s}'=30\\
2\mu_{21} & \text{for } \bar{s}=65, \bar{s}'=69\\
\mu_{22} & \text{for } \bar{s}=65, \bar{s}'=66\\
\mu_{12} & \text{for } \bar{s}=66, \bar{s}'=31\\
2\mu_{21} & \text{for } \bar{s}=66, \bar{s}'=70\\
\mu_{12} & \text{for } \bar{s}=67, \bar{s}'=32\\
\mu_{21} & \text{for } \bar{s}=67, \bar{s}'=72\\
2\mu_{22} & \text{for } \bar{s}=67, \bar{s}'=68\\
\mu_{12} & \text{for } \bar{s}=68, \bar{s}'=29\\
2\mu_{21} & \text{for } \bar{s}=68, \bar{s}'=67\\
2\mu_{22} & \text{for } \bar{s}=68, \bar{s}'=65\\
\mu_{12} & \text{for } \bar{s}=69, \bar{s}'=29\\
2\mu_{21} & \text{for } \bar{s}=69, \bar{s}'=67\\
2\mu_{22} & \text{for } \bar{s}=69, \bar{s}'=65\\
\mu_{12} & \text{for } \bar{s}=70, \bar{s}'=30\\
2\mu_{21} & \text{for } \bar{s}=70, \bar{s}'=69\\
2\mu_{22} & \text{for } \bar{s}=70, \bar{s}'=66
\end{cases}$$

$$P_{a_{22}}(\bar{s},\bar{s}') = \frac{1}{q}\begin{cases} \mu_{12} & \text{for } \bar{s}=71, \bar{s}'=31 \\ 2\mu_{21} & \text{for } \bar{s}=71, \bar{s}'=70 \\ \mu_{12} & \text{for } \bar{s}=72, \bar{s}'=37 \\ 2\mu_{22} & \text{for } \bar{s}=72, \bar{s}'=70 \\ \mu_{12} & \text{for } \bar{s}=73, \bar{s}'=32 \\ \mu_{21} & \text{for } \bar{s}=73, \bar{s}'=72 \\ 2\mu_{22} & \text{for } \bar{s}=73, \bar{s}'=68 \\ \mu_{12} & \text{for } \bar{s}=74, \bar{s}'=32 \\ \mu_{21} & \text{for } \bar{s}=74, \bar{s}'=72 \\ 2\mu_{22} & \text{for } \bar{s}=74, \bar{s}'=68 \\ \mu_{11} & \text{for } \bar{s}=75, \bar{s}'=29 \\ 2\mu_{21} & \text{for } \bar{s}=75, \bar{s}'=44 \\ 2\mu_{22} & \text{for } \bar{s}=75, \bar{s}'=42 \\ \mu_{11} & \text{for } \bar{s}=76, \bar{s}'=31 \\ 2\mu_{21} & \text{for } \bar{s}=76, \bar{s}'=46 \end{cases}$$

With the above set up, we ran the Policy Iteration algorithm (in MATLAB) to show that inequality (3.2) holds and hence the stated policy is optimal.

(Theorem 5.5) □

Conjecture 5.7. *When a line is faster, the optimal policy assigns both vertical flexible servers to the fast line. If one of the vertical flexible servers is busy at the slow line, the policy still assigns the other vertical flexible server to the fast line.*

Conjecture 5.8. *When the first station is faster in one of the lines, the optimal policy performs as follows: let the line with the fast station be line k. For the second pair, the policy chooses line k if it has a blocked server. If line k has no blocking and the other line does, the policy chooses the other line. If neither of the lines have a blocked server, it chooses line k. For the first pair, the policy allocates the vertical flexible server to the line not including the vertical flexible server in its second station. The only exception occurs when the vertical flexible server at the first pair is available*

and there is no blocking in any of the lines. In this case, the policy synchronizes and assigns both vertical flexible servers to line k.

Conjecture 5.9. *When the second station is faster in one of the lines, the optimal policy assigns both vertical flexible servers to the line with the fast station. The exceptions are $\bar{s} = \{21, 40, 49\}$, where the policy sends the vertical flexible server at the second pair of stations to the line with the slow station.*

Conjecture 5.10. *When the first pair of stations are faster in both lines, the optimal policy is the same as the policy described in Theorem 5.5.*

Conjecture 5.11. *When the second pair of stations are faster in both lines, the optimal policy is the same as the policy described in Theorem 5.5.*

Conjecture 5.12. *When the first and second stations are faster in two different lines, the optimal policy is similar to the policy described in Theorem 5.5. The exception is that when there are three busy/blocked servers in the line with the fast second station, the policy synchronizes and assigns vertical flexible servers to that line.*

We attempted to provide the MDP for $N = 3$, a dedicated server per station, and $\bar{F} = (1, 1, 1)$. However, we found it too tedious (and error-prone) to iterate through the large number of states to construct the MDP. We decided not to do this exercise at this point. The other alternative is to automate the steps of setting up the MDP. To study non-idling policies that perform hand-off, one needs to list all the possible states and eliminate those that include idle servers. The state space should further be refined by eliminating states where a hand-off can occur. However, there are no means to verify if all the resulting states are reachable, other than composing

transitions and navigating through them. We do not comment on the complexity of automatic generation of transitions, as we have not attempted to follow this approach.

5.4 Simulation

To address larger systems, we developed heuristics in the form of allocation policies and applied them over a number of configurations. The major tasks performed by policies are: clearing blocking, serving starved stations, synchronizing vertical flexible servers to one line, or distributing vertical flexible servers among lines. The policies that we consider are as follows (for each policy, priority is specified in parenthesis):

1. (Blocking) clear blocking from end to beginning. Stick to one line unless there is blocking in the other line and no blocking in the current line. Also, avoid creating additional blocking.

2. same as policy 1, except clear blocking from beginning to end.

3. (Distributing) clear blocked stations which are blocked only in one line (and free or busy in the other line). For the remaining vertical flexible servers, starting from the first line, allocate them alternately between the two lines.

4. (Synchronizing) clear blocked stations and allocate the remaining vertical flexible servers to the line with more vertical flexible servers.

5. (Blocking) clear blocking from end to beginning. Stick to one line unless there is blocking in the other line and no blocking in the current line.

6. same as policy 5, except clear blocking from beginning to end.

7. (Distributing) clear stations which are blocked only in one tandem line (and free or busy in the other line). Assign the remaining vertical flexible servers to the first line for the first half of stations and to the second line for the second half of stations.

8. (Starvation) clear starvation from end to beginning. Assign the remaining vertical flexible servers to one line.

9. same as policy 8, except serve starved stations from beginning to end.

We applied these policies to a number of configurations: $w = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$ with a dedicated server per station; $w = (1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1)$ with a dedicated server per station; $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$ with dedicated server assignment of $s = (12, 7, 13, 3, 5, 4, 1, 10, 9)$; $w = (12, 7, 13, 3, 5, 4, 1, 10, 9)$ with dedicated server assignment of $s = (11, 7, 12, 3, 5, 4, 1, 15, 14)$; $w = (1, 3, 4, 5, 7, 9, 10, 12, 13)$ with dedicated server assignment of $s = (3, 5, 6, 7, 9, 11, 12, 14, 15)$. In all of the configurations, a vertical flexible server is assigned to each pair of stations. However no clear ordering among the policies could be identified. Solving the MDP for longer lines would provide insights to better analyze and compare these heuristics.

5.5 Flexibility assessment

In this section we compare the throughput improvement gained by horizontal flexibility versus vertical flexibility. Originally we aimed to study systems that include both horizontal and vertical flexible servers. However as presented in this section, it turned out that horizontal flexibility is by far more beneficial than vertical flexibility. Hence we decided not to study these hybrid systems at this time.

5.5.1 Horizontal flexibility

As discussed in Section 3.6, we found that the highest improvements in throughput are gained from making the first few servers flexible.

5.5.2 Vertical flexibility

Presume two parallel tandem lines with equal numbers of stations, but different numbers of dedicated servers at each pair of stations. We aim to change a number of dedicated servers (of the tandem line with more servers for a pair of stations) to vertical flexible servers. We let $|\bar{F}|$ take all values from 0 to N. We would like to examine how much throughput is augmented as $|\bar{F}|$ increases.

In particular, we consider tandem lines with 10 stations, one line with one dedicated server per station and the other line with two dedicated servers per station. Service times of all stations are exponentially distributed with rate 1. The deployed policy allocates vertical flexible servers always to one tandem line and sends them to the other tandem line, only if the other tandem line has a blocked station and the current line does not. It clears blocking from the end to the beginning (i.e. policy 1 above).

Figure 5.7 depicts the throughput gained by choosing only one station to assign a vertical flexible server to. As the figure suggests, making a server flexible from the stations at the end of the line results in highest throughput improvement. The intuition is that clearing a blocked station towards the end of the line is more beneficial than clearing blocking from stations elsewhere in the line, as it also helps the upstream stations. The reduction in the throughput value is explained by the fact that the employed policy is not the optimal policy.

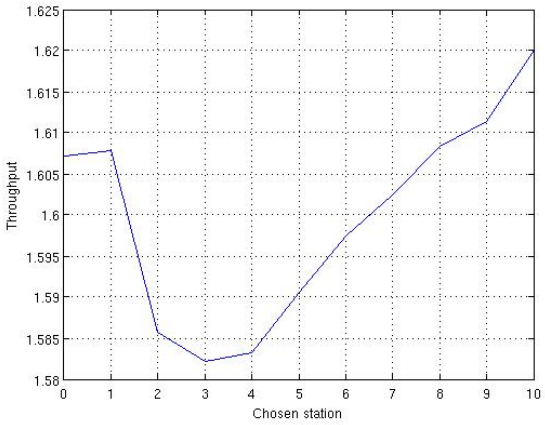

Figure 5.7: Throughput versus making the dedicated server at station $i \in \{1, \ldots, 10\}$ vertically flexible

Increasing $|\bar{F}|$ can be performed in various orders. Vertical flexible servers can be chosen from the end to the beginning, from the beginning to the end, or in a zig-zag manner around the middle station. Figure 5.8 compares the throughput gained by each of these selection orders when policy 1 is employed over the configuration described above. Similarly Figure 5.9 illustrates our observations, when policy 4 is applied.

Comparing Figures 5.8 and 5.9 for vertical flexibility and Figures 3.31, 3.32, and 3.33 for horizontal flexibility shows that the throughput improvement gained by horizontal

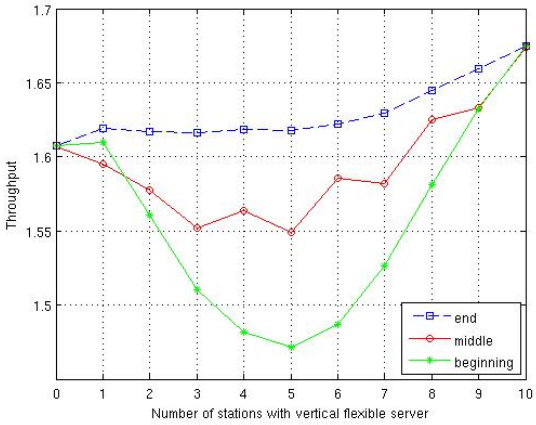

Figure 5.8: Throughput versus $|\bar{F}|$ - employing policy 1 for allocations

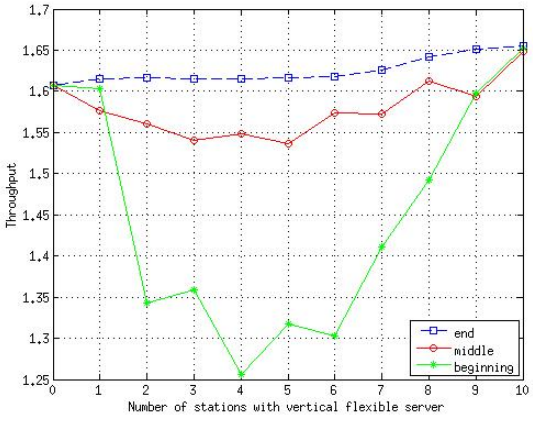

Figure 5.9: Throughput versus $|\bar{F}|$ - employing policy 4 for allocations

flexibility is far greater than that for vertical flexibility. Notice that in Figures 5.8 and 5.9, throughput values are not monotonically increasing under all policies, as the employed policies are not the optimal policies.

In the next chapter, we study a number of applications of all the models provided so far. The next chapter can be used to gain intuition about the problems we have discussed in this work.

Chapter 6

Applications

In this chapter we present a number of applications of this research. Our research is applicable to any situation where zero-buffer tandem lines appear. A zero-buffer (alternatively called a blocking) environment arises either from characteristics of the processing technology itself, or from the absence of storage capacity between stations. In the following, we briefly describe some of these applications from each of the two categories.

6.1 Bed management

In the domain of healthcare, the Department of Health and Aged Care (1999) defines Integrated Bed Management as "the management of all admissions, stays, transfers, and discharges by a hospital within a framework that integrates and coordinates all processes related to these activities". The issue is seemingly very simple: if the patient flows into and out of beds are optimized, there will be fewer long waits for emergency admissions and fewer cancellations of elective surgeries. There are multiple factors

working against the achievement of this simple goal, summarized as:

- cross-divisional boundaries (i.e. lack of coordination between units)
- lack of appropriate information systems
- outdated policies for the allocation of priority and management of beds.

We now present a real world example that includes a tandem line of care units. Assume a patient arrives to the Emergency unit. After an initial assessment, the patient is moved to the Express unit in which he could stay for hours or days, until a space becomes available in the next unit. The Express unit contains a number of beds called express (or intermediate) beds that are used to transfer patients among the Emergency, Express, and Medicine units. A specific nurse is assigned to the Express unit who is responsible for moving the express beds among units.

The patient is then transferred to the Medicine unit for a longer term and more extensive care. If required, the patient is called for surgery. Surgery itself is composed of pre-operation, operation, and post-operation. For less complicated surgeries no pre-operation monitoring is required. However the patient is sent to the recovery room after their operation for short term monitoring. Once discharged from the Medicine or Recovery unit, the patient is sent to the Complex unit for long term care. The patient can be discharged at any point, if no further care is required.

It is a known issue in the healthcare domain (identified for example by the Department of Health and Aged Care (1999)) that long waiting times happen due to long stays in the Complex (or the Alternative Level of Care) unit. Long stays are usually related to older patients who are afraid of not receiving proper care outside of a hospital and therefore prefer to stay in the hospital. It is observed that early discharge

of these patients would result in them coming back to the Emergency unit shortly after they leave. This introduces an extra cost of re-examination, which should be avoided. The reasonable solution is to deliver these patients to community care such as home care, a nursing home, a retirement home, palliative care, or a rehabilitation center.

Mapping the described setting to our model, units correspond to stations, beds to servers, and patients to jobs. Similar to our model, patient care in a unit is not pre-emptive. We assume that it is possible to extract an estimation of the service time distributions for the units. Distributions act as an abstraction of the lengths of stay of patients in the units, as there is little control over the discharge time of a patient. The zero-buffer restriction is analogous to the fact that there is no space between the units to accommodate patients who are discharged from a source unit and are to be admitted to a target unit. In other words, it is impossible to leave a patient on the ground in order to release their beds. If a patient is discharged from a source unit but the target unit has no free beds, the patient should remain in his bed in the source unit until a bed becomes available in the target unit. This corresponds to blocking in our model. In worst case, there are always patients waiting to be admitted to the first unit, which is equivalent to jobs queueing before the first station. Given the above analogy, we use our dedicated server allocation algorithm to assign beds to the units.

Flexible servers correspond to the express beds. Hand-off is when a patient is moved between a configurable bed and a fixed bed. Our policy for choosing and moving flexible servers can be applied to this problem. Also, the fact that our proposed policies clear blocking from end to the beginning is consistent with the observation that blocking in the Complex units causes long waiting times in the Emergency units

(and hence should be cleared before other blocked stations).

In addition to express beds, other ad-hoc usages of flexible beds have been reported. Van der Linden et al. (2012) report that in a level 1 trauma center in the Netherlands, patients were transferred from Emergency departments to other hospitals, due to long waiting times in the Emergency departments. This occurred despite sub-maximal hospital occupancy, mainly because specialists were reluctant to admit patients from other specialties to "their beds". For example, a patient with rectal carcinoma had to wait in the Emergency department until transfer to another hospital, despite enough available beds on other, non-oncology wards. In this study, Van der Linden et al. (2012) considered 15 beds as flexible beds, consisting of free beds from all the units, and allowed them to be shared during off-hours (4 PM to 8 AM). As a result, they observed improved throughput and reduced number of transfers to other hospitals. Baillie et al. (1997) also report that in order to deal with fluctuations in demands and high loads, free beds should be shared among all units. They specifically mention surgical specialties as recipients of excess patients from medical specialties.

Unlike horizontal flexibility, vertical flexibility is not used in the bed management problem. It might happen that a patient arrives to the Emergency unit of a hospital but after initial examination is sent to another hospital to receive specialized care (such as cardiac or pediatric). Another possibility is that a patient in the Medicine unit might be diagnosed with cancer and is transferred to a cancer center. However it is not a common practice to move patients in between hospitals with similar levels of expertise on a particular condition, as it introduces extra cost for re-examinations.

We note that assuming exponential distributions for service times might not be realistic in the healthcare environment. However in our model, we have considered

distributions with coefficients of variation other than one to model more generic service time distributions.

6.2 Assembly lines

An assembly line is an example where the lack of storage capacity introduces a zero-buffer production environment. Boysen et al. (2007) define an assembly line as a flow-oriented production system where the productive units performing the operations, referred to as stations, are aligned in a serial manner. Jobs visit stations successively as they are moved along the line, usually by some kind of transportation system, e.g., a conveyor belt, a fork lift, or even gravity. Stations could include workers, tools, or machines.

Boysen et al. (2007) describe Assembly Line Balancing, a major activity in this domain, as follows: a configuration planning activity which is related to equipping and aligning the productive units for a given production process, before the actual assembly can start. The question is how to set the system capacity (cycle time, number of stations, station equipment) and how to decide work contents of productive units (task assignment, sequence of operations) such that the throughput is maximized and the cost is minimized.

A portion of the assembly line balancing problem is the allocation of workers or facilities to stations. This resource allocation question can be modeled with our dedicated server allocation algorithm, given that transportation systems do not have extra spaces to hold completed pieces between stations. This assumption translates to the zero-buffer restriction. An example of such a configuration is an automobile assembly line. Different stations might be press shop, body shop, paint shop, and

final assembly area (each of these including smaller stations themselves). Each car is carried by a specific conveyor with no extra conveyors between stations (reported by Hu et al. (2011)). A more generic example is any production process that includes different stages of spot welding.

Note that service times of stations in assembly lines typically have low variability, due to application of automated machines or assigning single step tasks to human operators. Introducing horizontal or vertical flexibility to the system might not be as beneficial as assigning dedicated servers. However if there is limited budget for adding servers, one might need to share servers among stations using flexibility. In the context of assembly lines, usage of parallel stations has also been reported by Lia et al. (2009). Vertical flexibility can be used to model a resource shared among the parallel stations.

6.3 Steel production

This is an example of a case where the technology itself requires a zero-buffer production environment. In this production process, the temperature or other characteristics (such as viscosity) of the material requires that each operation follows the previous operation, immediately. In production of steel, molten steel undergoes a series of operations such as molding into ingots, unmolding, reheating, soaking, and preliminary rolling.

Smith and Hashemi (2006) describe the steel production process as follows. When iron is smelted, it contains more carbon than is desirable. To become steel, it must be melted and reprocessed to reduce the carbon to the correct amount, at which point other elements can be added. This liquid is then continuously cast into ingots. The

ingots are then heated in a soaking pit and hot rolled into slabs, blooms, or billets. In modern foundries, these processes often occur in one assembly line, with ore coming in and finished steel coming out .

Mapping to our model, jobs represent the molten steel. Servers are all of the facilities that shape or heat the metal. Our model only partially applies to the described situation. The reason is that adding servers is costly and therefore does not usually happen. However the initial server allocation problem can be aided with our proposed algorithm.

Similar concerns exist in the plastic molding and silverware production industries where operations must follow each other immediately to prevent degradation. Further examples occur in the chemical and pharmaceutical industries where the anodizing of aluminum products such as pipes, trim, and truck grilles has to be performed with the least possible gaps in between (see (Hall and Sriskandarajah, 1996)).

Such applications are closely related to a field of research that deals with scheduling jobs in a no-wait setting. In a no-wait environment, a job must be processed from start to completion, without any interruption either on or between stations. On the other hand, in a blocking environment a job that has completed processing can remain on the station until a downstream station becomes free. For a survey on scheduling under no-wait and blocking in tandem lines, refer to the work by Hall and Sriskandarajah (1996).

6.4 Semi-conductor production

Semi-conductors or more specifically silicon boards are frequently used in different electronic devices which contain many transistors. However, due to small sizes of

these transistors, the silicon bases on which they reside need to be absolutely flawless. For this reason, semi-conductor production is performed in clean rooms which contain far fewer dust particles than in the air. The first step is to melt raw polysicilon (in the presence of argon gas to eliminate any air) to get a single silicon crystal. The crystal is then sliced into thin disks. Then, in a process called photolithography, chip designs are imprinted to the disks. In the photolithography process, disks are first covered with photosensitive chemicals which harden, once exposed to ultraviolet light. In sealed dark rooms, light is shone through an image of the design. Then the chemical is washed away and the design remains. Each layer of the design requires different materials.

If as little as one particle lands in a critical area through the design, malfunctioning of a chip may result. Hence disks are not buffered through the process and they continuously go through the required steps. Mapping to our model, any of the steps in the clean room is considered a station and devices or experts performing the steps correspond to servers. Knowing that it is desirable to keep the number of workers in the clean room as low as possible, it is reasonable that each expert operates multiple devices. In such cases, an expert can be considered as a flexible server.

6.5 Food production

Several food materials spoil shortly after harvesting and hence need to be consumed while they are fresh. An alternative is to can food materials when they are fresh and use them at a later time. To keep food materials fresh, the canning process should not take long, meaning that food materials can not be kept in intermediate buffer spaces and should be passed seamlessly to subsequent stations. In particular,

no buffer space is allowed between the cooking operation and before the canning operation (Andriansyah et al., 2010). Similar issues can be found in producing juice and beer (e.g., see (Fey, 2000)). Another example is a study performed by Ramudhin and Ratliff (1995), about a condiments manufacturer, with mayonnaise and various types of salad dressing as the product. The nature of the product dictates hygienic consideration as critical.

Chapter 7

Conclusion

In this thesis we studied allocation of servers to stations in zero-buffer tandem lines, with the goal of maximizing the throughput. We provided an algorithm and policies to specify allocations for dedicated and flexible servers, respectively. Extensive numerical analysis was employed to study the algorithm and the policies. Simulations covered service time distributions with different coefficients of variation. Our work can be broken down to the following stages.

With dedicated servers only, we provided a server allocation algorithm. We considered a wide range of configurations, varying the number of stations, number of servers per station, and service time distributions. In contrast to infinite buffer settings where balancing total service rates of stations is the only concern, in zero-buffer tandem lines, under certain conditions, stations with lower mean service times should be prioritized. We introduced the multiplicity effect to explain that servers also play the role of buffers in the absence of real storage between stations. Studying the impact of variability in service times, we found that the algorithm performs well in the following combinations of service time distributions: coefficients of variation are

approximately one except for a small number of stations which are greater than one; coefficients of variation are less than one with equal values; coefficients of variation are greater than one with equal values. In other cases adjustments are required to the algorithm. These adjustments were discussed, but the final word still remains to be said on this issue.

With dedicated and flexible servers, we focused on configurations with only one flexible server. The optimal policy for two stations performs hand-off, clears blocking, and admits new jobs when there is no blocking. We used the Policy Iteration algorithm to show that for small tandem lines with equal workloads, the optimal policy clears blocking from end to beginning and avoids starvation. Simulation results illustrated that for longer tandem lines with varying workloads, the optimal policy tends to prioritize clearing blocking from stations with higher mean service times. We also identified optimal policies when flexibility was constrained to be between two neighbouring stations. We showed that in such cases, optimal policies follow the same structure as optimal policies for non-constrained cases.

With dedicated and vertical flexible servers between two parallel tandem lines, we considered similar tandem lines with two stations, with a dedicated server at each station. We proved that optimal policies are non-idling and perform hand-off whenever possible. We also found that optimal policies are rate dependent. With equal rates, and a vertical flexible server at each pair of stations, the optimal policy clears blocking, if it exists, and roughly equally distributes vertical flexible servers between the lines. We also specified optimal policies for unequal service rates. For cases with more than two stations, heuristic assignment policies are proposed.

7.1 Design decisions

From the perspective of overall system design, given the workload of a tandem line and the total number of servers available, one can use the proposed algorithm (Section 2.2) to allocate dedicated servers to stations. An analysis is then required to identify the trade-off between the cost and the throughput improvement of making servers flexible. Section 3.6 suggests that the highest throughput improvements are gained when the first few dedicated servers are made flexible. The improvement is diminished as the number of flexible servers increases. Once the optimum number of flexible servers is identified, the policies discussed on the selection order of flexible servers (Section 3.6) can be employed to make servers flexible. From there, the near optimal allocation policy for flexible servers, provided in Chapter 3, should be used to decide the assignment of flexible servers.

A similar paradigm should be followed for two parallel tandem lines. After dedicated servers are assigned, an analysis is required to identify the trade-off between the cost and the throughput improvement (Section 5.5.2) of making servers vertically flexible. Once the optimum number of vertical flexible servers is identified, the policies discussed for the selection order of vertical flexible servers (Section 5.5.2) can be employed to make servers vertically flexible. From there, the near optimal allocation policy for vertical flexible servers, provided in Chapter 5, should be used to decide the assignment of vertical flexible servers.

If making servers either vertically or horizontally flexible is a decision variable, the discussion in Section 5.5.2 shows that all things being equal, one should make servers horizontally flexible, as this leads to higher throughput improvements. Of course, the cost comparison of making servers horizontally flexible or vertically flexible should

also be taken into account.

As an example, consider the bed management problem. The capacity of a care unit is usually defined in terms of the number of beds in that unit. Given the total capacity of a hospital and estimates of average lengths of stay in the units of the hospital, managers assign beds to the units. Our fixed server allocation can be used to perform this initial allocation of beds.

Due to fluctuations in demand, it may be the case that some units will have free beds, while other units are congested. In such cases, idle beds could be shared among units (if physically possible), according to the discussion at the beginning of this section. From Section 3.6, we know that the higher the number of flexible beds, the higher throughput will be. However, bed management and transferring beds can involve collaboration among a number of staff members, namely clinical ward managers (or clinical nurse managers), business managers, chief executives, and duty nurse managers. In several hospitals, dedicated positions for managing beds are defined with titles such as Bed Manager or Bed Management Coordinator. Hence, the management and movement of beds require staff efforts which may be costly.

Bed managers need to decide about the trade-off between improved throughput and higher expenses, due to making beds flexible. The discussion in Section 3.6 shows that making the first few beds flexible results in the highest throughput improvements. Therefore, to increase the throughput and simultaneously keep cost as low as possible, it is potentially enough to make only a few beds flexible.

7.2 Future work

In this work we have considered non-collaborative homogeneous servers. A natural extension to this work is to study the case where servers are heterogeneous (i.e. have different service rates at different stations) or can work collaboratively at a station. Considering costs for making a server flexible and then minimizing the overall cost and maximizing the throughput, is also of interest. It is also possible to introduce costs for hand-off or for navigation through stations for flexible servers. Future work specific to different stages is:

- dedicated servers: additional simulations might lead to more concrete statements on how to justify visit periods for arbitrary coefficients of variation across all stations. Simulation variables would be the number of stations, distributions of service times, number of servers, and arrangement of coefficients of variation.

- horizontal flexible servers: it is desirable to run the Policy Iteration algorithm for $F > 1$ to identify optimal allocation policies. With $F > 1$, the allocation problem is more challenging, as the allocation of a flexible server is dependent on the assignment of other flexible servers. The problem becomes more interesting when long tandem lines are considered for $F > 1$, as more complicated collaboration among flexible servers is required.

- constrained flexibility: it is desirable to run the Policy Iteration algorithm for several other unbalanced workloads to identify optimal allocation policies. The collaboration mechanism among constrained flexible servers, when lines are unbalanced, is of interest.

- vertical flexible servers: it is possible to extend the analytical results to more than two stations and use the insight to improve current heuristics. Also it would be interesting to study parallel tandem queues, when tandem queues are not similar (i.e. have different mean service time vectors).

Bibliography

Ahn, H.-S., Duenyas, I., and Lewis, M. E. (2002). Optimal control of a two-stage tandem queuing system with flexible servers. *Probability in the Engineering and Informational Sciences*, 16(4):453 – 469.

Ahn, H.-S., Duenyas, I., and Zhang, R. Q. (2004). Optimal control of a flexible server. *Advances in Applied Probability*, 36(1):139 – 170.

Ahn, H.-S. and Lewis, M. E. (2012). Flexible server allocation and customer routing policies for two parallel queues when service rates are not additive. *Working Paper*.

Andradóttir, S. and Ayhan, H. (2005). Throughput maximization for tandem lines with two stations and flexible servers. *Operations Research*, 53(3):516 – 531.

Andradóttir, S., Ayhan, H., and Down, D. G. (2001). Server assignment policies for maximizing the steady-state throughput of finite queueing systems. *Management Science*, 47(10):1421 – 1439.

Andradóttir, S., Ayhan, H., and Down, D. G. (2003). Dynamic server allocation for queueing networks with flexible servers. *Operations Research*, 51(6):952 – 968.

Andradóttir, S., Ayhan, H., and Down, D. G. (2007). Dynamic assignment of dedicated and flexible servers in tandem lines. *Probability in the Engineering and Informational Sciences*, 21(4):497 – 538.

Andriansyah, R., Van Woensel, T., Cruz, F. R. B., and Duczmal, L. (2010). Performance optimization of open zero-buffer multi-server queueing networks. *Computers and Operations Research*, 37(8):1472 – 1487.

Baillie, H., Wright, W., McLeod, A., Craig, N., Leyland, A., Drummond, N., and Boddy, A. (1997). Bed occupancy and bed management. Department of Public Health, University of Glasgow.

Boysen, N., Fliedner, M., and Scholl, A. (2007). A classification of assembly line balancing problems. *European Journal of Operational Research*, 183(2):674 – 693.

Buyukkoc, C., Varaiya, P., and Walrand, J. (1985). The $c\mu$ rule revisited. *Advances in Applied Probability*, 17(1):237 – 238.

Cheng, D. W. and Zhu, Y. (1993). Optimal order of servers in a tandem queue with general blocking. *Queueing Systems*, 14(3):427 – 437.

Department of Health and Aged Care (1999). Managing beds better - balancing supply and demand the NDHP-2 experience. Commonwealth of Australia, Canberra.

Duenyas, I. and Van Oyen, M. P. (1995). Stochastic scheduling of parallel queues with set-up costs. *Queueing Systems Theory and Applications*, 19(4):421 – 444.

El-Rayah, T. E. (1979). The effect of inequality of interstage buffer capacities and operation time variability on the efficiency of production line systems. *International Journal of Production Research*, 17(1):77 – 89.

Feldman, R. M. and Valdez-Flores, C. (1996). *Applied Probability & Stochastic Processes*. PWS Publishing Company.

Fey, J. (2000). Design of a fruit juice blending and packaging plant. Ph.D. thesis, Eindhoven University of Technology.

Futamura, K. (2000). The multiple server effect: Optimal allocation of servers to stations with different service-time distributions in tandem queueing networks. *Annals of Operations Research*, 93(1):71 – 90.

Hajek, B. (1982). Optimal control of two interacting service stations. In *21st IEEE Conference on Decision and Control*, volume 21, pages 840 – 845.

Hall, N. G. and Sriskandarajah, C. (1996). A survey of machine scheduling problems with blocking and no-wait in process. *Operations Research*, 44(3):510 – 525.

Hasenbein, J. J. and Kim, B. (2011). Throughput maximization for two station tandem systems: a proof of the Andradóttir–Ayhan conjecture. *Queueing Systems*, 67(4):365 – 386.

Heavey, C., Papadopoulos, H. T., and Browne, J. (1993). The throughput rate of multistation unreliable production lines. *European Journal of Operational Research*, 68(1):69 – 89.

Hillier, F. S. and Boling, R. W. (1979). On the optimal allocation of work in symmetrically unbalanced production line systems with variable operation times. *Management Science*, 25(8):721 – 728.

Hillier, F. S. and So, K. C. (1989). The assignment of extra servers to stations

in tandem queueing systems with small or no buffers. *Performance Evaluation*, 10(3):219 – 231.

Hillier, F. S. and So, K. C. (1993). Some data for applying the bowl phenomenon to large production line systems. *International Journal of Production Research*, 31(4):811 – 822.

Hillier, F. S. and So, K. C. (1995). On the optimal design of tandem queueing systems with finite buffers. *Queueing Systems*, 21(3-4):245 – 266.

Hillier, F. S. and So, K. C. (1996). On the simultaneous optimization of server and work allocations in production line systems with variable processing times. *Operations Research*, 44(3):435 – 443.

Hillier, F. S., So, K. C., and Boling, R. W. (1993). Toward characterizing the optimal allocation of storage space in production line systems with variable processing times. *Management Science*, 39(1):126 – 133.

Himmelblau, D. M. (1972). *Applied Nonlinear Programming*. McGraw-Hill Book Company.

Hordijk, A. and Koole, G. (1992). On the shortest queue policy for the tandem parallel queue. *Probability in the Engineering and Informational Sciences*, 6(1):63 – 79.

Hu, S., Ko, J., Weyand, L., ElMaraghy, H., Lien, T., Koren, Y., Bley, H., Chryssolouris, G., Nasr, N., and Shpitalni, M. (2011). Assembly system design and operations for product variety. *CIRP Annals - Manufacturing Technology*, 60(2):715 – 733.

Iravani, S. M. R. (1997). *Tandem queues attended by a moving server*. PhD thesis, University of Toronto.

Kırkızlar, E., Andradóttir, S., and Ayhan, H. (2012). Flexible servers in understaffed tandem lines. *Production and Operations Management*, 21(4):761 – 777.

Lia, J., Blumenfeld, D. E., Huang, N., and Alden, J. M. (2009). Throughput analysis of production systems: recent advances and future topics. *International Journal of Production Research*, 47(14):3823 – 3851.

Lippman, S. A. (1975). Applying a new device in the optimization of exponential queuing systems. *Operations Research*, 23(4):687 – 710.

Magazine, M. J. and Stecke, K. E. (1996). Throughput for production lines with serial work stations and parallel services facilities. *Performance Evaluation*, 25(3):211 – 232.

Muth, E. J. and Alkaff, A. (1987). The bowl phenomenon revisited. *International Journal of Production Research*, 25(2):161 – 173.

Pandelis, D. G. (2008). Optimal stochastic scheduling of two interconnected queues with varying service rates. *Operations Research Letters*, 36(4):492 – 495.

Papadopoulos, H., Heavey, C., and O'Kelly, M. (1989). Throughput rate of multistation reliable production lines with inter station buffers (I) exponential case. *Computers in Industry*, 13(4):229 – 244.

Papadopoulos, H., Heavey, C., and O'Kelly, M. (1990). Throughput rate of multistation reliable production lines with inter station buffers (II) Erlang case. *Computers in Industry*, 13(4):317 – 335.

Puterman, M. L. (2005). *Markov Decision Processes - Discrete Stochastic, Dynamic Programming*. John Wiley & Sons, Inc.

Ramudhin, A. and Ratliff, H. D. (1995). Generating daily production schedules in process industries. *IIE Transactions*, 27(8):646 – 656.

Shanthikumar, J. G. and Yao, D. D. (1987). Optimal server allocation in a system of multi-server stations. *Management Science*, 33(9):1173 – 1180.

Sheu, R.-S. and Ziedins, I. (2010). Asymptotically optimal control of parallel tandem queues with loss. *Queueing Systems*, 65(3):211 – 227.

Smith, W. F. and Hashemi, J. (2006). *Foundations of Materials Science and Engineering*. McGraw-Hill.

Spicer, S. and Ziedins, I. (2006). User-optimal state-dependent routing in parallel tandem queues with loss. *Journal of Applied Probability*, 43(1):274 – 281.

Spinellis, D., Papadopoulos, C., and Smith, J. M. (2000). Large production line optimization using simulated annealing. *International Journal of Production Research*, 38(3):509 – 541.

Tassiulas, L. and Ephremides, A. (1993). Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, 39(2):466 – 478.

Van der Linden, C., Lucas, C., van der Linden, N., and Lindeboom, R. (2012). Evaluation of a flexible acute admission unit: Effects on transfers to other hospitals and patient throughput times. *Journal of Emergency Nursing*, 38(1).

Van Woensel, T., Andriansyah, R., Cruz, F. R. B., Smith, J. M., and Kerbache, L. (2010). Buffer and server allocation in general multi-server queueing networks. *International Transactions in Operational Research*, 17(2):257 – 286.

Winston, W. (1977). Optimality of the shortest line discipline. *Journal of Applied Probability*, 14(1):181 – 189.

Wu, C.-H., Down, D., and Lewis, M. (2008). Heuristics for allocation of reconfigurable resources in a serial line with reliability considerations. *IIE Transactions*, 40(6):595 – 611.

Wu, C.-H., Lewis, M., and Veatch, M. (2006). Dynamic allocation of reconfigurable resources in a two-stage tandem queueing system with reliability considerations. *Automatic Control, IEEE Transactions on*, 51(2):309 – 314.

Yarmand, M. H. and Down, D. G. (2012). Server allocation for zero buffer tandem queues. *Submitted August 2012 to European Journal of Operational Research*.

Yoneyama, K. and Ishii, H. (1997). The throughput rate of interchangeable parallel two-stage tandem queue with correlated service times. *Computers and Mathematics with Applications*, 33(7):29 – 37.