SCHEDULING IN QUEUEING SYSTEMS WITH SPECIALIZED OR ERROR-PRONE SERVERS

A Dissertation
Presented to
The Academic Faculty

By

Junqi Hu

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of H. Milton Stewart School of Industrial and Systems Engineering

Georgia Institute of Technology

August 2020

SCHEDULING IN QUEUEING SYSTEMS WITH SPECIALIZED OR ERROR-PRONE SERVERS

To my parents.

ACKNOWLEDGEMENTS

First of all, I would like to express my foremost and deepest gratitude to my advisors, Dr. Sigrún Andradóttir and Dr. Hayriye Ayhan, for their generous support and invaluable guidance throughout my doctoral studies. They have not only been my academic advisors, but also incredible mentors and role models to my professional and personal development. I am indebted to Dr. Sigrún Andradóttir, without whom I would never be able to start my Ph.D. at ISyE.

I would also like to thank my committee members, Dr. Robert Foley, Dr. Siva Theja Maguluri, and Dr. Tuğçe Işık, for generously offering their time to serve as committee members. Their insightful questions, constructive comments, and valuable suggestions have helped improve the quality of this thesis. Thanks to Bob, I was fortunate to learn the foundations of Stochastic processes from him. I am also very thankful to Siva for his generous help on my research and academic career.

I am also thankful to all the faculty and staff members in ISyE for their help and efforts. Special thanks to Warren Bell and Jonathan Etress, who have helped me fix the leaking ceiling in my office after each rainstorm five years in a row.

I thank all my friends accompanying me in the past years. Special thanks go to Shuang Li, Liyan Xie, Yining Cao, Yuting Zhang, Yun Ji, Chao Zhao, Yilin Li, Huimin Wei, Wenyi Xu, Huijun Zhang.

Finally, I would like to express my sincerest gratitude to my family, my mother Danli Cai, my father Baiqing Hu, my uncles and aunt, for their selfless love and endless support through my entire life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Consider a multi-server queueing system with tandem stations, finite intermediate buffers, and an infinite supply of jobs in front of the first station. Our goal is to maximize the long-run average throughput of the system by dynamically assigning the servers to the stations.

For the first part of this thesis, we analyze a form of server coordination named task assignment where each job is decomposed into subtasks assigned to one or more servers, and the job is finished when all its subtasks are completed. We identify the optimal task assignment policy of a queueing station when the servers are either static, flexible, or collaborative. Next, we compare task assignment approaches with other forms of server assignment, namely teamwork and non-collaboration, and obtain conditions for when and how to choose a server coordination approach under different service rates. In particular, task assignment is best when the servers are highly specialized; otherwise, teamwork or non-collaboration are preferable depending on whether the synergy level among the servers is high or not. Then, we provide numerical results that quantify our previous comparison. Finally, we analyze server coordination for longer lines, where there are precedence relationships between some of the tasks. We show that for static task assignment, internal buffers at the stations are preferable to intermediate buffers between the stations, and we present numerical results that suggest our comparisons for one station systems generalize to longer lines.

The second part of this thesis studies server allocation when the servers can work in teams and the team service rates can be arbitrary. Our objective is to improve the performance of the system by dynamically assigning servers to teams and teams to stations. We first establish sufficient criteria for eliminating inferior teams, and then we identify the optimal policy among the remaining teams for the two-station case. Next, we investigate the special cases with structured team service rates and with teams of specialists. Finally, we provide heuristic policies for longer lines with teams of specialists when the servers are generalists, and numerical results that suggest that our heuristic policies are near-optimal.

In the final part of this dissertation, we consider the scenario where a job might be broken and wasted when being processed by a server. Servers are flexible but non-collaborative, so that a job can be processed by at most one server at any time. We identify the dynamic server assignment policy that maximizes the long-run average throughput of the system with two stations and two servers. We find that the optimal policy is either a single or a double threshold policy on the number of jobs in the buffer, where the thresholds depend on the service rates and defect probabilities of the two servers. For larger systems, we provide a partial characterization of the optimal policy. In particular, we show that the optimal policy may involve server idling, and if there exists a distinct dominant server at each station, then it is optimal to always assign the servers to the stations where they are dominant. Finally, we propose heuristic server assignment policies motivated by experimentation with three-station lines and analysis of systems with infinite buffers. Numerical results suggest that our heuristics yield near-optimal performance for systems with more than two stations.

CHAPTER 1
INTRODUCTION AND BACKGROUND

Optimal control through dynamic resource allocation is commonly seen in production systems and in service systems (such as call centers, ridesharing systems, and healthcare systems). Given a real-world problem, we strive to model the system so as to make it solvable or analyzable with reasonable and realistic assumptions; and develop optimal or near-optimal control policies that are applicable in practice. More specifically, this thesis revolves around the dynamic allocation of cross-trained workforce in manufacturing and service systems, with the objective of maximizing the long-run average throughput of the system.

Cross-trained (flexible) servers are widely discussed as a useful tool to improve the performance of production and service systems [29]. Yet much remains to be done to advance the use of server flexibility. In a multi-server queueing system with tandem stations and finite intermediate buffers, our goal in this thesis is to improve the performance of the system by dynamically assign the servers to stations under scenarios which are practical in real life but seldom discussed in the literature. We refer to objects that are either being processed or being served as jobs. In particular, this work considers the following three cases: (1) a job can be decomposed into multiple subtasks with or without precedence relationship; (2) a job needs to be served by a group of servers as a team; (3) A job might be broken and wasted when being processed by a server. Assume that there is an infinite supply of jobs in front of the first station and infinite storage space after the last station. Servers are cross-trained and allowed to switch between stations with negligible time and cost. Unless specified otherwise, we assume the service requirements are independently and exponentially distributed. The system operates under manufacturing blocking, that is, a completed job is blocked from moving to the downstream buffer when that buffer is full.

In the first part of this dissertation, we consider a form of server coordination named task assignment. Consider a multi-server queueing system with tandem stations, and suppose that we need to determine how to deploy the servers at each station. The standard approach would be to let the servers assigned to each station work in parallel without collaboration. However, if a job can be decomposed into multiple subtasks and there are no precedence relationships among the subtasks, then it is possible we could improve the long-run average throughput of the system via other forms of server coordination. One form of server coordination involves assigning each subtask to one or more servers, and a job is completed when all of its subtasks are completed. We refer to this as *task assignment*, or as the *maximum model* since the service time of a job at a queueing station equals the maximum of the times it takes to complete the subtasks. We consider three types of task assignments based on their server flexibility and collaboration levels. Another form of server coordination is teamwork, where servers work together as a team with a combined service rate. We also consider the non-collaboration approach, where the servers work in parallel and will complete all the subtasks of a job by themselves.

First, we identify the optimal server assignment policy when the servers are either static, flexible, or collaborative. Next, we compare task assignment approaches with other forms of server assignment, namely teamwork and non-collaboration, and provide guidelines for on whether and to what extent we can improve the performance of the system via these server coordination methods. In particular, task assignment is best when the servers are highly specialized; otherwise, teamwork or non-collaboration are preferable depending on whether the synergy level among the servers is high or not. Moreover, we further investigate these methods when the servers are generalists or specialists, and provide the corresponding numerical results. Finally, we analyze server coordination for longer lines, where there are precedence relationships between some of the tasks. We obtain that for static task assignment, it is always better to allocate the available buffers within stations as internal buffers rather than after stations as intermediate buffers (however, this result does not hold for flexible or collaborative task assignment). Finally, our numerical results for the two-station case suggest that our one-station results can be generalized to longer lines.

Next, we shift our attention to server allocation in terms of teams. Most existing papers assumed a fixed synergy level whenever the servers collaborate. However, there are many situations when jobs need to be served by a group of servers as a team (e.g.a surgery), but the efficiencies of server collaboration between the team members are diversified. For example, the collaboration between servers A and B in a team could be efficient while the collaboration between servers A and C is inefficient. Thus, we do not restrict ourselves to some specific relationship between the team service rates and other factors, such as the individual service rates of team members, or the synergy between the servers of a team. To be more reasonable and include all possible types of server collaborations, we focus on the service rate of the team instead of the individual service rates of the servers at each station. First, we select the team assignments that are on the Pareto boundary and irreplaceable as the optimal assignment set. Then, we specify the optimal policy among the teams in the optimal assignment set for two stations case. Next, we apply our optimal policies to two special cases. In the first case, the team service rate is proportional to the sum of the service rates of team members. This kind of server collaboration has been analyzed in the past [6, 11, 13]. We validate our results by checking if the optimal policy we obtained is consistent with previous work under this special case. In the second case, we assume that there are different types of servers with different specialties, and the team formation is constrained in that each team must consist of exactly one server of each type. The optimal policy indicates that, for teams of specialized servers where the servers are generalists, we use a permanent set of teams that are formed based on their ability. Based on this result, we develop near-optimal heuristic policies that are validated by our numerical results for longer lines for teams of specialized servers when they are generalists .

The final part of this thesis studies the optimal server allocation in presence of defects. We consider a Markovian tandem line with an equal number of stations and flexible but non-collaborative servers. At any time, each server can work on at most one job, and a job can be processed by at most one server. Most of the existing papers that study queueing systems with flexible servers assumed that the servers are reliable with zero defect probabilities. To the best of our knowledge, this is the first paper that considers the dynamic scheduling of servers when they are flexible and error-prone. For systems with two stations and two servers, we formulate the system as a Markov decision process and characterize the optimal policy with respect to which server has the higher effectiveness overall. More specifically, we prove that the optimal server assignment policy is either a single or a double threshold policy on the number of jobs in the buffer, where the thresholds depend on the service rates and defect probabilities of the two servers. For larger systems, we provide a partial characterization of the optimal policy. First, we verify that the optimal policy may involve server idling (except for the server assigned to the first station); next, when a distinct server is the fastest and most reliable at each station, the optimal policy always assigns the server to the station where they are dominant. Finally, we propose heuristic server assignment policies motivated by experimentation with three-station lines and analysis of systems with infinite buffers. Numerical results suggest that our heuristics yield near-optimal performance for systems with more than two stations.

The rest of this dissertation is organized as follows. In Chapter 2, we provide an overview of the literature on queueing systems with flexible servers and/or error-prone servers. In Chapter 3, we investigate when and how to choose different forms of server coordination methods when a job can be decomposed into multiple subtasks with or without precedence relationship. In Chapter 4, we consider the optimal server assignment problem in terms of teams. In Chapter 5, we study the optimal scheduling of queueing systems with flexible, non-collaborative and error-prone servers. In Chapter 6, we summarize the main contributions of this thesis and present our future research directions. Finally, we provide supplementary materials for Chapters 3 and 5 in Appendices A and B, respectively.

CHAPTER 2
LITERATURE REVIEW

In Chapter 2, we provide an overview of the literature on queueing systems with flexible servers and/or error-prone servers. First, we review systems the flexible servers in Section 2.1. In Section 2.2, we focus on the literature on queueing systems with failure-prone or error-prone servers.

2.1 Flexible Servers

There is a significant amount of literature on queues with flexible servers. For a comprehensive review of the literature in this area, see Hopp and Van Oyen [29], and Qin, Nembhard, and Barnes [42].

2.1.1 Non-collaborative Servers

Several papers considered server allocation when the servers are non-collaborative. For example, Van Oyen, Gel, and Hopp [47] investigated the case when servers are working in parallel, and proved that the "pick-and-run" policy they proposed is not optimal. Işık, Andradóttir, and Ayhan [31] have studied server allocation in tandem queues with equal number of stations and servers when the servers are flexible and non-collaborative, and provided the optimal policy for two stations along with heuristic policies that were near-optimal for larger systems. Ahn, Duenyas, and Lewis [1] analyzed systems with two stations in tandem and two flexible servers when the individual service rates only depend on the station, and they considered the scenarios when the servers are collaborative with additive service rates (i.e., $\alpha = 1$), and when servers are non-collaborative. Argon and Andradóttir [15] considered systems with jobs that are divided into subtasks that would be processed in tandem and discussed how to improve the system throughput by partial pooling of the servers, stations, and subtasks of a job. Yarmand and Down [49, 50] investigated server allocation for tandem queues with zero buffers and homogeneous servers at each station. [49] proposed an allocation method that assigns servers to stations based on the mean service times and the current number of servers assigned to each station, and they validated their algorithm by simulation. [50] considered a mixture of dedicated and flexible servers, and studied server allocation policy for flexible servers that maximizes the throughput of the system. They concluded that the optimal policy for systems with two stations and one flexible server performs a hand-off (switch the job between flexible and dedicated server when the dedicated server is starved or blocked), clears blocking, and admits new jobs when there is no blocking. Pandelis and Van Oyen [38] analyzed a tandem queueing system with partially cross-trained servers, and they identified structural properties of worker allocation policies that maximize the throughput of the system.

2.1.2 Collaborative Servers

There is a significant amount of literature on optimal server allocation when the servers are flexible and collaborative with combined service rates that are additive. Thus the servers neither gain nor lose efficiency when they collaborate. Van Oyen, Gel, and Hopp [47] introduced the teamwork approach with identical servers (which they referred to as the "expedite policy") that maximizes the long-run average throughput and minimizes the work-in-process (WIP) of the system. Andradóttir, Ayhan, and Down [7] considered the dynamic scheduling policy when the servers are flexible only when their assigned stations are blocked or starved, while Andradóttir and Ayhan [6] and Kırkızlar, Andradóttir, and Ayhan [33] discussed the case when the servers are flexible all the time. Specifically, [6] focused on the system with the number of servers more than the number of stations (i.e. overstaffed), and [33] focused on the understaffed queueing system.

Other papers discussed server collaboration with non-additive combined service rates. Andradóttir, Ayhan, and Down [11] discussed the case when servers are synergistic (i.e.,

$\alpha > 1$) with a common synergy factor α for each station. Wang, Andradóttir, and Ayhan [48] further analyzed the case when servers are synergistic with different synergy factors for different stations. Moreover, Andradóttir, Ayhan, and Down [13] investigated the case when the server collaboration is inefficient (i.e., $\alpha < 1$). Ahn and Lewis [4] considered the problem of routing the arrivals and allocating the servers in a parallel queueing system with two types of customers and collaborative servers that can be either superadditive or subadditive.

2.1.3 Teams of Specialized Servers

Our work focuses on a continuous-time Markovian queueing system with preemptive service. Perron [40] investigated the discrete time-based server scheduling problem with teams of specialized servers and non-preemptive service, and provided experimental results suggest that the problem is not solvable without decomposition and decomposing is hard and error prone.

In this thesis, we consider on one type of jobs with a fixed requirement of team formation, and study the best team formation among heterogeneous servers. Some papers addressed multi-class jobs with different team formation requirements with homogeneous servers when offering service as a certain type of specialists. Courcoubetis and Reiman [21] studied a parallel queueing system with N identical servers and two types of jobs which they referred to as ordinary jobs and locking jobs. Ordinary jobs need one server to be processed, while locking jobs need all N servers to work together. They verified that to maximize the long run average reward of the system, they should prioritize the ordinary jobs until the number of locking jobs reaches some threshold. Gurvich and Van Mieghem[27] studied the capacity management problem of a network with teams of specialized servers and multi-class jobs. They showed that highest priority must be given to the tasks that require the most collaboration (i.e., largest number of specialized servers in the team), and a mismatch between the priority level and the collaboration level can lead to inevitable capacity loss. Lodree, Altay, and Cook [36] considered dynamic allocation of medical staff to casualties with random server arrivals and heterogeneous team requirements. They modeled this system as a discrete-time finite horizon stochastic dynamic programming problem and developed efficient heuristic policies for computational study.

2.1.4 Task assignment

There is limited work on the task assignment server coordination structure. Buzacott [19] studied task assignment for a single-stage queue where each job is split into parallel subtasks and the next job cannot begin until all the subtasks of the previous job are completed. He showed that, in terms of the mean total number of jobs in the system, this kind of server coordination is not superior to a series system with buffers between the two servers when utilization of servers is sufficiently high. However, in [19], the servers are static and identical, and there are no internal buffers after each subtask, which is restrictive. In our paper, we allow flexible and collaborative servers with general service rates and finite internal buffers of arbitrary size. Tsai and Argon [16, 46] also discussed the task assignment approach (which they called a "splitting system") for a single station with multiple subtasks and finite internal buffers. In [46], the servers are collaborative in that they can work together on the same subtasks with additive combined service rates. They proved that their splitting system is equivalent to a system with two tandem stations, and identify the optimal policy that maximizes the long-run average throughput by analyzing this equivalent system. In [16], they considered flexible servers with switching costs when servers transit among subtasks and holding costs for the jobs in the system, and provided a partial characterization of the policy that minimizes the long-run average costs. In our paper, we include the cases when servers are either static, flexible, or collaborative and find the policy that maximizes the long-run average throughput of the system directly. Thus, we both consider more general servers and use a different method to find the optimal policy compared to [16, 46]. In summary, the previous studies [16, 19, 46] focus on systems with one station and a specific type of servers. We consider different types of servers depending on their flexibility and collaboration levels, which is more general and practical. We also compare task assignment approaches with each other and with other forms of server coordination to identify the best server coordination approach.

2.2 Error-prone Servers

Some existing papers addressed server breakdowns, so that the service process is interrupted. For a thorough review of literature of queues with interruptions, see Krishnamoorthy, Pramod, and Chakravarthy [35]. Andradóttir, Ayhan, and Down [9, 10] considered the dynamic assignment of servers to maximize the long-run average throughput of queueing networks with failure-prone servers and stations. Specifically, [9] investigated the system with infinite buffers, while [10] analyzed the system with tandem stations and finite intermediate buffers. Özkan and Kharoufeh [37] studied routing problem of a Markovian queueing system with one reliable server and one faster but failure-prone server, with an objective of minimizing the long-run average number of customers in the system. They proved that it is always optimal to route customers to the faster server when it is available if the system is stable, and there exists an optimal threshold policy that depends on the queue length and the state of the faster server for the slower server. However, these server breakdowns do not cause any damage to the products but only postpone the service process.

Other works considered server breakdown with customer abandonment. Economou and Kapodistria [23] investigated a single server queue with server breakdown such that the current customer leaves the system, and the remaining customers become impatient as long as the server is down. Towsley and Tripathi [45] analyzed queueing systems with disasters such that the occurrence of disasters forces all customers to leave the system and causes the main server to fail. Yechiali [51] also considered this kind of system disaster while the new arrivals during server breakdown become impatient. All these papers analyzed various service measures of the system including system size distribution and the sojourn time distribution of the systems.

Some papers considered the planning and control of rework in production systems, see Flapper, Fransoo, Broekmeulen, and Inderfurth [26] for a review of the literature in this area. Specifically, Teunter and Flapper [44] considered the lot sizing problem for a production line with non-defective, reworkable defective, and non-reworkable defective items produced in lots, so that after producing a fixed number (N) of lots, they will switch to rework on the reworkable defective items until they are all fixed. They assumed that their products were perishable, and there were set-up times and costs attached to switching between producing new items and reworking of the defective items. They derived the average profit for any fixed N and then determined the optimal N numerically. However, they assumed a fixed defect probability of a production and focused on the lot sizing problem that maximizes the profit and only provided numerical examples of the optimal policy. Elshafei, Khan, and Duffuaa [24] also considered a dynamic programming model with products that are classified as non-defective, reworkable defective, and non-reworkable defective by an inspector. They proposed a dynamic programming algorithm that minimizes the total inspection cost, where the total cost includes the cost of false rejection of good items, the cost due to false acceptance of defective items which are either reworkable or non-reworkable, the cost of inspection, and the cost of rework. However, they focused on using inspection to eliminate the defective items, while we focus on identifying how servers should be assigned to tasks in the presence of defects.

CHAPTER 3

SERVER COORDINATION IN QUEUEING SYSTEMS: WHEN AND HOW?

In this chapter, we first consider a one-station system, and generalize to longer lines later. For simplicity, we assume that there are two servers at each station and that the task at each station is decomposed into two subtasks. We will discuss task assignment approaches with non-negative and finite internal buffer sizes. For one-station systems, let $0 \leq B_i < \infty$ denote the internal buffers after subtask i for $i = 1, 2$. We assume that the network operates under the manufacturing blocking mechanism, that is, a job that finishes its service at a subtask when the internal buffer after that subtask is full stays at the subtask and blocks other jobs from entering service there. Since a job will leave the system right after both of its subtasks are completed, only one of the two internal buffers can have jobs waiting inside at any given time. The service requirements for different subtasks and different jobs are independent; denote the service requirement for subtask i as S_i. Unless specified otherwise, we assume that S_i follows an exponential distribution with rate ξ_i for $i = 1, 2$. The service rate of server i working on subtask j is μ_{ij} for $i, j = 1, 2$. Assume there are infinitely many jobs waiting in front of the station and infinite room for completed jobs after the station. Without loss of generality, let $\xi_i = 1$ for $i = 1, 2$. Assume that $\sum_i \mu_{ij} > 0, \forall j = 1, 2$ (otherwise, the throughput of the system is zero), and $\sum_j \mu_{ij} > 0, \forall i = 1, 2$ (otherwise, the problem reduces to having only one server). See Figure 3.1 for the flow plot of the task assignment system with one station and two servers.

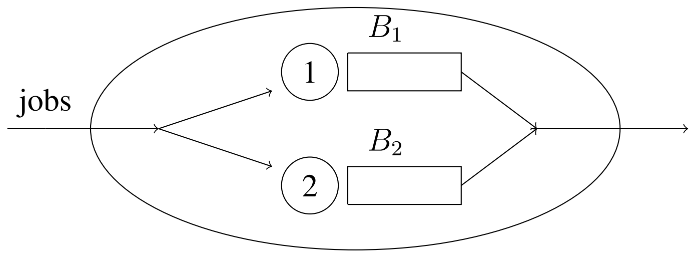


Figure 3.1: Task Assignment System with One Station

We investigate three types of task assignment based on their server flexibility and collaboration levels. In all three task assignment approaches, each server $i \in \{1, 2\}$ is assigned to a task j_i, where $\{j_1, j_2\} = \{1, 2\}$, at all times when there is work to be done at both subtasks. However, the task assignment approaches differ in the assignment of servers when there is no work to be done at one of the subtasks (due to blocking). In static task assignment, each server $i \in \{1, 2\}$ is at all times assigned to task j_i. In flexible task assignment, each server can be reassigned to the other subtask (replacing the server originally assigned there) when their assigned subtask is blocked. In collaborative task assignment, the servers can either stay, switch, or work as a team when one subtask is blocked. We investigate the performance of these different task assignment approaches and compare them with teamwork and non-collaboration. We assume that when the servers work as a team, their combined service rate is proportional to the sum of their service rates with a non-negative coefficient α in both of the collaborative task assignment and teamwork approaches. Note that when $\alpha > 1$, the servers are synergistic in that their combined service rate is larger than the sum of their own service rates. When $\alpha < 1$, their collaboration is inefficient.

The outline of this paper is as follows. In Sections 3.1, 3.2, and 3.3, we investigate the static, flexible, and collaborative task assignment approaches, respectively. Within each section, we first obtain the optimal task assignment policy with general internal buffer sizes, and then discuss two special cases, namely zero buffers and asymptotically infinite buffers. In Section 3.4, we introduce three other server coordination approaches, namely teamwork with or without task partitioning and non-collaboration, and compare these three methods. And in Section 3.5, we provide a comparison of collaborative task assignment, teamwork with or without task partition, and non-collaboration, and determine when and how to choose from these methods based on different server flexibility levels. The proofs of several of our results are provided in Appendices A.1, A.2, and A.3, and the comparisons that are not discussed in Section 3.5 are given in Appendix A.4. We also discuss two special cases, namely when servers are generalists, and when servers are specialists. In Section 3.6, we investigate server coordination for longer lines, where we identify desirable buffer allocation choices and provide numerical results for two stations in tandem. In Section 3.7, we summarize our findings and conclude the paper. Supplementary explanations of both teamwork approaches are given in Appendix A.5.

3.1 Static Task Assignment

In this section, we consider a single queueing station with two servers. We assume that each job involves two subtasks and that each server specializes in a fixed subtask, so the servers are static and will be idle after finishing their current subtask and before starting the next subtask if their internal buffer is full. In Section 3.1.1, we obtain the optimal static task assignment approach with general internal buffer sizes. In Sections 3.1.2 and 3.1.3, we discuss the special cases when the internal buffers are zero and when the sum of the buffer sizes goes to infinity, respectively.

3.1.1 Optimal Policy

In this section, since the server assignment is static, there are two feasible assignments:

(i) Server i is assigned to subtask i for $i = 1, 2$, with the corresponding throughput

$$T_{12}^{s} = \frac{\mu_{11}\mu_{22}\sum_{k=0}^{B_1+B_2+1}\mu_{11}^{k}\mu_{22}^{B_1+B_2+1-k}}{\sum_{k=0}^{B_1+B_2+2}\mu_{11}^{k}\mu_{22}^{B_1+B_2+2-k}}. \tag{3.1}$$

(ii) Server 1 is assigned to subtask 2 and server 2 is assigned to task 1, with corresponding throughput

$$T_{21}^s = \frac{\mu_{21}\mu_{12}\sum_{k=0}^{B_1+B_2+1}\mu_{21}^k\mu_{12}^{B_1+B_2+1-k}}{\sum_{k=0}^{B_1+B_2+2}\mu_{21}^k\mu_{12}^{B_1+B_2+2-k}}.$$

Denote these two static assignments as A_{12}^s, A_{21}^s; under A_{ij}^s, i is the assignment of server 1 and j is the assignment of server 2.

Observe that the sums in the expressions for T_{12}^s and T_{21}^s could be rewritten using the formula for geometric sums. For instance,

$$\sum_{k=0}^{B_1+B_2+1}\mu_{11}^k\mu_{22}^{B_1+B_2+1-k} = \begin{cases} \frac{\mu_{11}^{B_1+B_2+2}-\mu_{22}^{B_1+B_2+2}}{\mu_{11}-\mu_{22}} & \text{if } \mu_{11} \neq \mu_{22}, \\ (B_1+B_2+2)\mu_{11}^{B_1+B_2+1} & \text{if } \mu_{11} = \mu_{22}. \end{cases}$$

However, throughout this paper, we keep sums in our expressions for two reasons: (i) to reduce the number of cases we need to consider, especially when there are multiple distinct sums in one expression (e.g., equation (3.7)); (ii) to avoid minus signs so that it is easier to identify the sign of the expression (e.g., equation (A.4)).

It is clear that the throughputs T_{12}^s, T_{21}^s of the static task assignment approaches are non-decreasing in the buffer sizes B_1 and B_2. Moreover, the internal buffer allocation does not affect the results. Thus, two static maximum models with different internal buffer sizes would have the same throughput as long as the sums of the two internal buffers, $B_1 + B_2$, are equal. However, this property is the result of the birth-and-death structure of the system with two subtasks. In general, when there are more than two subtasks, the buffer allocation will affect the throughput of static task assignment. To illustrate, consider three subtasks, three servers. The service rate of subtask i is μ_i, and the internal buffer size of subtask i is B_i for $i = 1, 2, 3$. Assume $\mu_1 = 2, \mu_2 = \mu_3 = 1$. Then,

1. If $B_1 = 1, B_2 = B_3 = 0$, the long-run average throughput is $\frac{72}{127}$;

2. If $B_1 = 0, B_2 = 1, B_3 = 0$, the long-run average throughput is $\frac{144}{329} < \frac{72}{127}$.

We need to compare the two throughputs T_{12}^s and T_{21}^s to identify the optimal assignment for this model. Without loss of generality, assume that we number the servers so that $\mu_{11} \geq \mu_{21}$. Then we have:

Proposition 3.1.1. *Suppose that $\mu_{11} \geq \mu_{21}$. Then,*

(i) If $\mu_{22} = 0$, Assignment A^s_{21} is optimal.

(ii) If $\mu_{12} \times \mu_{21} = 0$, Assignment A^s_{12} is optimal.

(iii) If $\mu_{ij} > 0$ for $i, j = 1, 2$, then there exists a unique $\mu^*_{22} \in (0, \mu_{12}]$ such that Assignment A^s_{12} is optimal if $\mu_{22} \geq \mu^*_{22}$, Assignment A^s_{21} is optimal if $\mu_{22} < \mu^*_{22}$. Moreover,

(a) If $\mu_{11} > \mu_{21}$, then μ_{11} and μ^*_{22} are the only positive roots of

$$x^{B_1+B_2+3}A_1 - xA_2 + A_3 = 0, \tag{3.2}$$

where

$$A_1 = (\mu_{11} - \mu_{21}) \sum_{k=0}^{B_1+B_2+1} \mu_{21}^k \mu_{12}^{B_1+B_2+2-k} + \mu_{11}\mu_{21}^{B_1+B_2+2},$$

$$A_2 = \mu_{11}^{B_1+B_2+3} \sum_{k=0}^{B_1+B_2+2} \mu_{21}^k \mu_{12}^{B_1+B_2+2-k},$$

$$A_3 = \mu_{11}^{B_1+B_2+3} \sum_{k=0}^{B_1+B_2+1} \mu_{21}^{k+1} \mu_{12}^{B_1+B_2+2-k}.$$

(b) If $\mu_{11} = \mu_{21}$, then $\mu^*_{22} = \mu_{12}$.

Proof. Since $\mu_{11} \geq \mu_{21}$ and $\mu_{11}+\mu_{21} > 0$, we have $\mu_{11} > 0$. Moreover, since $\mu_{21}+\mu_{22} > 0$ and $\mu_{12} + \mu_{22} > 0$, it follows that $\mu_{22} = 0$ implies that μ_{12} and μ_{21} are both positive, and $\mu_{12} \times \mu_{21} = 0$ implies that $\mu_{22} > 0$. And since the servers are static, zero service rate at one subtask leads to zero throughput. Thus, our results (i) and (ii) are trivial.

For (iii),

$$\frac{\partial T_{12}^s}{\partial \mu_{22}} = \frac{\mu_{11}^{B_1+B_2+3} \sum_{k=0}^{B_1+B_2+1} (B_1+B_2+2-k) \mu_{22}^{B_1+B_2+1-k} \mu_{11}^k}{(\sum_{k=0}^{B_1+B_2+2} \mu_{11}^k \mu_{22}^{B_1+B_2+2-k})^2} > 0.$$

Thus, T_{12}^s is increasing with respect to μ_{22}. Moreover, $\mu_{22} \to 0 \Rightarrow T_{12}^s \to 0$. By the symmetry of μ_{22} and μ_{11} in T_{12}^s, we can obtain that T_{12}^s is increasing with respect to μ_{11}. Similarly, we can obtain that T_{21}^s is increasing with respect to both of μ_{21} and μ_{12}.

If we treat μ_{22} as variable and $\mu_{11}, \mu_{21}, \mu_{12} > 0$ as given, let $f(\mu_{22}) = T_{12}^s - T_{21}^s$. Then $f(\mu_{22})$ is increasing with respect to μ_{22} since T_{21}^s does not depend on μ_{22}. Moreover, $f(0) = -T_{21}^s < 0$, and $f(\mu_{12}) = \tilde{T}_{21}^s - T_{21}^s$, where $\tilde{T}_{21}^s$ is the throughput of A_{21}^s with μ_{21} replaced by μ_{11}. Since $\mu_{11} \geq \mu_{21}$ and T_{21}^s is increasing with respect to μ_{21}, $\tilde{T}_{21}^s \geq T_{21}^s$ and $f(\mu_{12}) \geq 0$. Thus, there exist only one value of $\mu_{22}^* \in (0, \mu_{12}]$ such that $f(\mu_{22}^*) = 0$, and when $\mu_{22} \geq \mu_{22}^*$, $f(\mu_{22}) \geq 0$, and hence $T_{12}^s \geq T_{21}^s$; when $\mu_{22} < \mu_{22}^*$, $f(\mu_{22}) < 0$, and hence $T_{12}^s < T_{21}^s$. Now we know that $f(x) = 0$ has one positive root μ_{22}^*, where

$$f(x) = \frac{\mu_{11} x \sum_{k=0}^{B_1+B_2+1} \mu_{11}^k x^{B_1+B_2+1-k}}{\sum_{k=0}^{B_1+B_2+2} \mu_{11}^k x^{B_1+B_2+2-k}} - T_{21}^s.$$

Note that for $n \geq 0$,

$$(x - \mu_{11}) \sum_{k=0}^{n} \mu_{11}^k x^{n-k} = (x^{n+1} - \mu_{11}^{n+1}). \tag{3.3}$$

Therefore, by multiplying $f(x)$ with $(x - \mu_{11})$ and reorganizing the equation $f(x)(x - \mu_{11}) = 0$, we obtain equation (3.2). And the fact that (2) corresponds to $(x - \mu_{11}) f(x)$ adds one more positive root, i.e., μ_{11}, to equation (3.2). Thus, μ_{11} and μ_{22}^* are the only positive roots of equation (3.2).

When $\mu_{11} = \mu_{21}$, by the symmetric structure of T_{12}^s and T_{21}^s we know that $T_{12}^s = T_{21}^s$ when $\mu_{22} = \mu_{12}$. That is, when $\mu_{11} = \mu_{21}$, $f(\mu_{12}) = 0$. Since $f(x) = 0$ has only one positive root, we have $\mu_{22}^* = \mu_{12}$. $\square$

Since $\mu_{22}^* \leq \mu_{12}$, Proposition 3.1.1 implies that when servers have different specialty on the subtasks (i.e., $\mu_{11} \geq \mu_{21}$ and $\mu_{22} \geq \mu_{12}$), we should assign them to the subtask that they are better at. When one server is better at both subtasks than the other one (i.e., $\mu_{11} \geq \mu_{21}$ and $\mu_{12} \geq \mu_{22}$), we will assign server 1 to subtask 1 as long as the service rate of server 2 at subtask 2 is not too small.

3.1.2 Special Case 1: No Buffers

In this section, we consider the special case with no buffers and $\mu_{ij} > 0$, for $i, j = 1, 2$. (If $\mu_{ij} = 0$ for some $i, j \in \{1, 2\}$, then the server allocation policy that assigns server i to station j would have zero throughput, and the optimal policy is trivial.) When $B_1 = B_2 = 0$, we can simplify the expressions of the long-run average throughput of the system as follows. For assignment A_{12}^s, the corresponding throughput is

$$T_{12}^s = \frac{1}{\frac{1}{\mu_{11}} + \frac{1}{\mu_{22}} - \frac{1}{\mu_{11}+\mu_{22}}};$$

and for assignment A_{21}^s, the throughput is

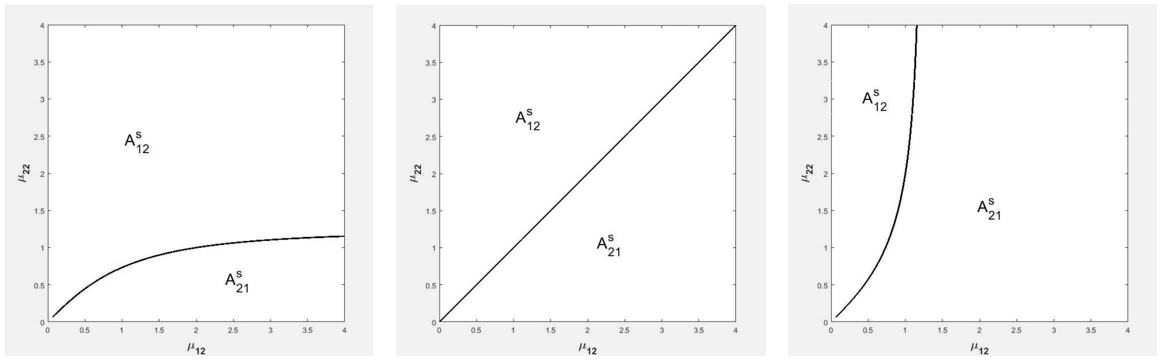
$$T_{21}^s = \frac{1}{\frac{1}{\mu_{21}} + \frac{1}{\mu_{12}} - \frac{1}{\mu_{21}+\mu_{12}}}.$$

Moreover, we can compute the precise value of μ_{22}^* as

$$\mu_{22}^* = \frac{-1 + \sqrt{1 + 4/M}}{2} \times \mu_{11}, \text{with } M = \frac{\mu_{11}}{\mu_{21}} + \frac{\mu_{11}}{\mu_{12}} - \frac{\mu_{11}}{\mu_{21} + \mu_{12}} - 1.$$

Note that since we label the servers so that $\mu_{11} \geq \mu_{21}$, we have $\frac{\mu_{11}}{\mu_{21}} \geq 1$ and $\frac{\mu_{11}}{\mu_{12}} > \frac{\mu_{11}}{\mu_{21}+\mu_{12}}$. Thus $M > 0$ and $\mu_{22}^* > 0$.

Example 3.1.1. *Consider the following three special cases of the service rates of subtask 1: (a) $\mu_{11} = 2, \mu_{21} = 1$; (b) $\mu_{11} = \mu_{21} = 1$; (c) $\mu_{11} = 1, \mu_{21} = 2$ (we include this case for symmetry even though it violates our convention that $\mu_{11} \geq \mu_{21}$). Figure 3.2 shows how the* *optimal assignment depends on μ_{12} and μ_{22}. For all three cases, Assignment A^s_{12} is optimal above the depicted line.*



(a) $\mu_{11} = 2, \mu_{21} = 1$ (b) $\mu_{11} = 1, \mu_{21} = 1$ (c) $\mu_{11} = 1, \mu_{21} = 2$

Figure 3.2: Optimal static task assignment with different service rates.

3.1.3 Special Case 2: Asymptotically Infinite Buffers

In this section, we consider the case when $B_1 + B_2 \to \infty$. Then,

1. In assignment A^s_{12}, $T^s_{12} \to \min\{\mu_{11}, \mu_{22}\}$.
2. In assignment A^s_{21}, $T^s_{21} \to \min\{\mu_{21}, \mu_{12}\}$.

Thus, when the sum of the buffer sizes goes to infinity, choose assignment A^s_{12} when $\min\{\mu_{11}, \mu_{22}\} \geq \min\{\mu_{21}, \mu_{12}\}$; otherwise, choose assignment A^s_{21}. And the throughput of the best static task assignment with asymptotically infinite buffers is:

$$T^s = \max\{\min\{\mu_{11}, \mu_{22}\}, \min\{\mu_{21}, \mu_{12}\}\}.$$

Intuitively, when the sum of the buffers goes to infinity, the throughput of the system is determined by the bottleneck subtask, that is, for any server assignment, the throughput is determined by the slower server. Thus, we will choose the assignment with the larger minimal service rate among both subtasks.

3.2 Flexible Task Assignment

In this section, we discuss the flexible task assignment approach. When servers are flexible, the server assignment involves two stages. When both servers are working, each server is assigned to a subtask based on a primary assignment; and when one of the subtasks is blocked, the servers may be reassigned according to a secondary assignment.

Similar to the structure of Section 3.1, in Section 3.2.1, we obtain the optimal flexible task assignment approach with general buffer sizes. In Sections 3.2.2 and 3.2.3, we discuss the special cases when the internal buffers are zero and when the sum of the buffer sizes goes to infinity, respectively.

3.2.1 Optimal Policy

For the primary assignment, we have two choices just like in static task assignment. And when one of the subtasks is blocked, for the secondary assignment we can either let the blocked server be idle or reassign this server to replace the other server and work on the unfinished subtask until it is completed.

Note that our queueing system can be modeled as a birth-and-death process, and the secondary assignment only applies to the states on the boundaries. And if we can increase the transition rates of leaving the states on the boundaries, we reduce the time the process spends in these boundary states without making any changes to the rest of the system, and thus will increase the throughput of this birth-and-death process. Hence, to improve the throughput by using flexible servers, we will only apply the secondary assignment when we can increase the service rate by replacing the current server. Specifically, if our primary assignment involves assigning server i to subtask i for $i = 1, 2$, we can improve the throughput by secondary assignment only if $\mu_{12} > \mu_{22}$ or $\mu_{21} > \mu_{11}$. Similarly, if our primary assignment involves assigning server i to subtask $3 - i$ for $i = 1, 2$, we can improve the throughput by secondary assignment only if $\mu_{22} > \mu_{12}$ or $\mu_{11} > \mu_{21}$. Also, the optimal flexible task assignment will involve at most one flexible server for the secondary assignment; otherwise we can improve the throughput by changing the primary assignment. When neither server is reassigned in the secondary assignment, flexible task assignment is equivalent to static task assignment.

Let A^f_{ij} be the flexible task assignment that assigns server 1 to subtask i and server 2 to subtask j for $i, j = 1, 2$ when the two servers are static and no secondary assignment is needed. Let A^{kf}_{ij} be the flexible task assignment when exactly one server is flexible for the secondary assignment, where for $i, j, k \in \{1, 2\}$, i is the primary assignment of server 1, j is the primary assignment of server 2, and k is the flexible server for the secondary assignment. For simplicity, we only consider the case when $\mu_{11} \geq \mu_{21}$; we can obtain the results of the other case by relabeling the servers. In this case, A^f_{21}, A^{1f}_{21}, and A^{2f}_{21} can only be optimal if $\mu_{12} \geq \mu_{22}$ (otherwise, it is better to change the primary assignment). By our previous analysis, since $\mu_{11} \geq \mu_{21}$, A^{2f}_{12} and A^{2f}_{21} cannot be optimal. Moreover, when subtask 2 is blocked, we will assign server 1 to subtask 1 to obtain a larger service rate on the boundary; thus A^f_{21} is not optimal. Then there are three available assignment policies.

A^f_{12}: Server 1 is assigned to subtask 1 and server 2 is assigned to subtask 2 for the primary assignment; when some subtask's buffer is full, the corresponding server will be idle. The corresponding throughput $T^f_{12} = T^s_{12}$ is given in equation (3.1).

A^{1f}_{12}: Server 1 is assigned to subtask 1 and server 2 is assigned to subtask 2 for the primary assignment; if subtask 1 runs out of buffer space earlier, server 1 will replace server 2 to finish subtask 2; if subtask 2 runs out of buffer space earlier, server 2 will be idle. The corresponding throughput is

$$T^{1f}_{12} = \frac{\mu_{11}\mu_{12} \sum_{k=0}^{B_1+B_2+1} \mu_{11}^k \mu_{22}^{B_1+B_2+1-k}}{\mu_{12} \sum_{k=0}^{B_1+B_2+1} \mu_{11}^k \mu_{22}^{B_1+B_2+1-k} + \mu_{11}^{B_1+B_2+2}}.$$

A^{1f}_{21}: Server 1 is assigned to subtask 2 and server 2 is assigned to subtask 1 for the primary assignment; if subtask 2 is blocked earlier, server 1 will replace server 2 to finish subtask 1; if subtask 1 is blocked earlier, server 2 will be idle. The corresponding throughput is

$$T_{21}^{1f} = \frac{\mu_{11}\mu_{12}\sum_{k=0}^{B_1+B_2+1}\mu_{21}^k\mu_{12}^{B_1+B_2+1-k}}{\mu_{11}\sum_{k=0}^{B_1+B_2+1}\mu_{21}^k\mu_{12}^{B_1+B_2+1-k}+\mu_{12}^{B_1+B_2+2}}.$$

Observe that the throughputs of flexible task assignment are non-decreasing in the buffer sizes B_1 and B_2. Moreover, the internal buffer allocation does not affect the results. Thus, two flexible task assignment models with different buffer sizes have the same throughput as long as the sums of the two buffers, B_1+B_2, are equal. Similar to static task assignment, this property only holds under our two-subtasks assumption.

The following proposition compares these policies when $\mu_{11} \geq \mu_{21}$. The proof is provided in Appendix A.1. It uses the notation $f \propto g$ if f is a positive multiple of g.

Proposition 3.2.1. *Suppose that $\mu_{11} \geq \mu_{21}$. Then*

(i) If $\mu_{22} \geq \mu_{12}$, Assignment A_{12}^f is optimal;

(ii) If $\mu_{22} < \mu_{12}$, there exists a unique $\mu_{22}^* \in [0, \mu_{12}]$ such that Assignment A_{12}^{1f} is optimal when $\mu_{22}^* \leq \mu_{22} < \mu_{12}$; Assignment A_{21}^{1f} is optimal when $0 \leq \mu_{22} < \mu_{22}^*$. Moreover,

a. If $\mu_{11} > \mu_{21} > 0$, then μ_{11} and μ_{22}^* are the only positive roots of

$$x^{B_1+B_2+2}A_4 - xA_5 + A_6 = 0, \tag{3.4}$$

where

$$A_4 = (\mu_{11}-\mu_{21})\sum_{k=0}^{B_1+B_2+1}\mu_{21}^k\mu_{12}^{B_1+B_2+1-k} + \mu_{21}^{B_1+B_2+2},$$

$$A_5 = \mu_{11}^{B_1+B_2+2}\sum_{k=0}^{B_1+B_2+1}\mu_{21}^k\mu_{12}^{B_1+B_2+1-k},$$

$$A_6 = \mu_{11}^{B_1+B_2+2}\sum_{k=0}^{B_1+B_2}\mu_{21}^{k+1}\mu_{12}^{B_1+B_2+1-k}.$$

b. If $\mu_{11} > \mu_{21} = 0$, then $\mu_{22}^* = 0$.

c. If $\mu_{11} = \mu_{21} > 0$, then $\mu_{22}^* = \mu_{12}$.

Note that, when $\mu_{22} = 0$ (which implies that $\mu_{12}, \mu_{21} > 0$), A_{12}^f yields zero throughput since it assigns server 2 to subtask 2 all the time, and the throughput of A_{12}^{1f} can be simplified as $T_{12}^{1f} = \frac{\mu_{11}\mu_{12}}{\mu_{11}+\mu_{12}}$. By multiplying both the numerator and denominator of T_{12}^{1f} by $\sum_{k=0}^{B_1+B_2+1} \mu_{21}^k \mu_{12}^{B_1+B_2+1-k}$ and comparing with T_{21}^{1f}, it is obvious that $T_{12}^{1f} < T_{21}^{1f}$. Thus, when $\mu_{22} = 0$, A_{21}^{1f} is optimal. When $\mu_{12} = 0$, A_{12}^f is optimal since both A_{12}^{1f} and A_{21}^{1f} yield zero throughputs. Finally, when $\mu_{21} = 0$, then A_{12}^{1f} is optimal for $\mu_{22} < \mu_{12}$ and A_{12}^f is optimal for $\mu_{22} \geq \mu_{12}$.

For the primary assignment, similar to the case of the static task assignment, if the servers have different specialty on the subtasks, we will assign them to the subtask that they are better at. On the other hand, if one server dominates the other one at both subtasks, we will use the better server as the flexible server, and assign the better server primarily to subtask 1 as long as the service rate of the dominated server at subtask 2 is not too low.

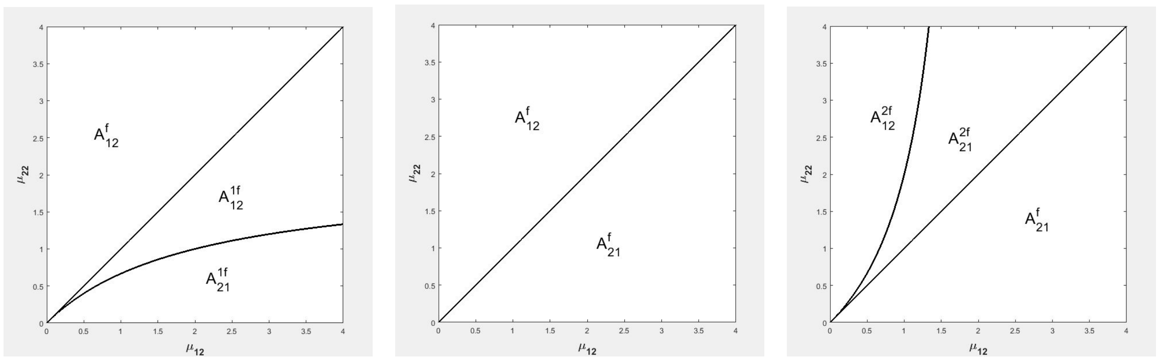
3.2.2 Special Case 1: No Buffers

In this section, we consider the special case with zero buffers. When $B_1 = B_2 = 0, \mu_{12} > 0$, we can compute the precise value of μ_{22}^* as

$$\mu_{22}^* = \begin{cases} \frac{1}{\frac{1}{\mu_{12}} + \frac{1}{\mu_{21}} - \frac{1}{\mu_{11}}}, & \text{if } \mu_{21} > 0; \\ 0, & \text{if } \mu_{21} = 0. \end{cases}$$

Since $\mu_{11} \geq \mu_{21}$ implies that $\frac{1}{\mu_{21}} \geq \frac{1}{\mu_{11}}$, it follows that $\mu_{22}^* \geq 0$.

Example 3.2.1. *Consider the following three special cases of the service rates of subtask 1: (a) $\mu_{11} = 2, \mu_{21} = 1$; (b) $\mu_{11} = \mu_{21} = 1$; (c) $\mu_{11} = 1, \mu_{21} = 2$ (we include this case for symmetry even though it violates our convention that $\mu_{11} \geq \mu_{21}$). Figure 3.3 shows* *how the optimal assignment changes with respect to the service rates μ_{12}, μ_{22} of subtask 2. When $\mu_{11} = 2, \mu_{21} = 1$, Assignment A_{12}^{f} is optimal when μ_{22} is large; A_{12}^{1f} is optimal when μ_{22} is moderate; and A_{21}^{1f} is optimal when μ_{22} is small. When $\mu_{11} = \mu_{21} = 1$, the optimal flexible task assignment is equivalent to the optimal static task assignment. And when $\mu_{11} = 1, \mu_{21} = 2$, the results are symmetric to case (a).*



(a) $\mu_{11} = 2, \mu_{21} = 1$ (b) $\mu_{11} = 1, \mu_{21} = 1$ (c) $\mu_{11} = 1, \mu_{21} = 2$

Figure 3.3: Optimal flexible task assignment with different service rates.

3.2.3 Special Case 2: Asymptotically Infinite Buffers

In this section, we consider the case when $B_1 + B_2 \to \infty$. Then,

1. In assignment A_{12}^{f}, if $\mu_{11} \leq \mu_{22}$, $T_{12}^{f} \to \mu_{11}$; and if $\mu_{11} > \mu_{22}$, $T_{12}^{f} \to \mu_{22}$.
2. In assignment A_{12}^{1f}, if $\mu_{11} \leq \mu_{22}$, $T_{12}^{1f} \to \mu_{11}$; and if $\mu_{11} > \mu_{22}$, $T_{12}^{1f} \to \frac{\mu_{11}\mu_{12}}{\mu_{11}+\mu_{12}-\mu_{22}}$.
3. In assignment A_{21}^{1f}, if $\mu_{21} \leq \mu_{12}$, $T_{21}^{1f} \to \frac{\mu_{11}\mu_{12}}{\mu_{11}+\mu_{12}-\mu_{21}}$; and if $\mu_{21} > \mu_{12}$, $T_{21}^{1f} \to \mu_{12}$.

Under the assumption that $\mu_{11} \geq \mu_{21}$, this yields the following results by algebra:

1. If $\mu_{12} \geq \mu_{21}$, $\mu_{11} \leq \mu_{22}$, then $T_{12}^{f} = T_{12}^{1f} \geq T_{21}^{1f}$;
2. If $\mu_{12} \geq \mu_{21}$, $\mu_{11} > \mu_{22}$,
   (a) when $\mu_{22} \geq \mu_{12}$, then $T_{12}^{f} \geq T_{12}^{1f} \geq T_{21}^{1f}$;

   (b) when $\mu_{21} \leq \mu_{22} < \mu_{12}$, then $T_{12}^{1f} > T_{12}^{f}, T_{12}^{1f} \geq T_{21}^{1f}$;

   (c) when $\mu_{22} < \mu_{21}$, then $T_{21}^{1f} > T_{12}^{1f} > T_{12}^{f}$;

3. If $\mu_{12} < \mu_{21}$, $\mu_{11} \leq \mu_{22}$, then $T_{12}^{f} = T_{12}^{1f} > T_{21}^{1f}$;

4. If $\mu_{12} < \mu_{21}$, $\mu_{11} > \mu_{22}$,

   (a) when $\mu_{22} \geq \mu_{12}$, then $T_{12}^{f} \geq T_{12}^{1f} \geq T_{21}^{1f}$;

   (b) when $\mu_{22} < \mu_{12}$, then $T_{21}^{1f} > T_{12}^{1f} > T_{12}^{f}$.

By summarizing the above result, we can obtain that

1. If $\mu_{12} \geq \mu_{21}$, choose assignment A_{12}^{f} when $\mu_{22} \geq \mu_{12}$, assignment A_{12}^{1f} when $\mu_{21} \leq \mu_{22} < \mu_{12}$, and assignment A_{21}^{1f} when $\mu_{22} < \mu_{21}$;

2. If $\mu_{12} < \mu_{21}$, choose assignment A_{12}^{f} when $\mu_{22} \geq \mu_{12}$, and choose assignment A_{21}^{1f} when $\mu_{22} < \mu_{12}$.

Comparing this result with Proposition 3.2.1 shows that when $B_1 + B_2 \to \infty$, $\mu_{22}^{*} \to \min\{\mu_{12}, \mu_{21}\}$. Note that assignment A_{21}^{1f} is better if and only if $\mu_{22} = \min\{\mu_{11}, \mu_{21}, \mu_{12}, \mu_{22}\}$. Thus, when server 1 is better at subtask 1, we will assign server 1 to subtask 1 and server 2 to subtask 2 for the primary assignment as long as the service rate of server 2 at subtask 2 is not lower than the service rate of the slower subtask under the other possible primary assignment. Thus, as $B_1 + B_2 \to \infty$, static and flexible task assignment both use the same primary assignment of servers.

3.3 Collaborative Task Assignment

In this section, we discuss collaborative task assignment. Similar to flexible task assignment, in this model, each server's assignment involves two stages. In particular, each server is assigned to a subtask based on a primary assignment, and when one of the subtasks is blocked, the servers are both flexible and collaborative, that is, we can assign either one server or two collaborating servers to the working subtask for the secondary assignment. When the servers work together on a subtask, their combined service rate is proportional to the sum of their service rates with synergy factor $\alpha > 0$.

In Sections 3.1 and 3.2, we assume that $\mu_{11} \geq \mu_{21}$ (without loss of generality) and fully characterize the optimal server assignment policy for general buffer sizes and static or flexible servers, respectively. By contrast, in this section, we do not identify the optimal task assignment approach for all service rates and general buffer sizes because the introduction of collaboration largely increases the number of possible assignments. Specifically, there are 18 possible collaborative task assignments since there are two choices for the primary assignment and three choices (assign server 1, 2, or both) for each subtask for the secondary assignment. Moreover, there are still 12 possible assignments to analyze even under the assumption that $\mu_{11} \geq \mu_{21}$. Instead, our arguments in Section 3.2.1 imply that in any optimal policy, we will assign the server or servers with the highest individual or combined service rate for the secondary assignment when one of the subtasks is blocked. And once we have determined the secondary assignment, there are only two possible primary assignments to choose from. Thus, to avoid a tedious description of the optimal approach, we fix the service rates for the secondary assignment first and discuss the primary assignment of the servers as a function of their secondary assignment. Moreover, in this section we do not assume that $\mu_{11} \geq \mu_{21}$ (because we will adopt a different convention for labeling the servers).

In Section 3.3.1, we provide the optimal primary assignment given the secondary assignment for the collaborative task assignment approach with general internal buffer sizes. In Sections 3.3.2 and 3.3.3, we fully characterize the optimal policy for the special cases when the internal buffers are zero and when the sum of the buffer sizes goes to infinity, respectively.

3.3.1 Optimal Policy

We introduce some notation to better illustrate the results in the following sections. Let $\Sigma_1 = \mu_{11} + \mu_{21}, \Sigma_2 = \mu_{12} + \mu_{22}$; then the combined service rate of servers working together at subtask i is $\alpha\Sigma_i$ for $i = 1, 2$. Let $\mu_{11} = \beta_1\Sigma_1, \mu_{21} = (1 - \beta_1)\Sigma_1, \mu_{12} = \beta_2\Sigma_2, \mu_{22} = (1 - \beta_2)\Sigma_2$; then $\beta_j \in [0, 1]$ is the fraction of server 1 of the total service rate on subtask j, for $j = 1, 2$. Moreover, let x_j be the service rate of the boundary state when subtask $3-j$ is blocked, for $j = 1, 2$. Then, $x_j = \max\{\mu_{1j}, \mu_{2j}, \alpha\Sigma_j\}$ in any optimal policy. Note that we will let the servers collaborate at subtask j if and only if $\alpha\Sigma_j \geq \max\{\mu_{1j}, \mu_{2j}\}$, i.e., when $\alpha \geq \max\{\beta_j, 1 - \beta_j\}$, for $j = 1, 2$. And if $\alpha < \min\{\max\{\beta_1, 1 - \beta_1\}, \max\{\beta_2, 1 - \beta_2\}\}$, then the servers will never work together, and the problem is equivalent to flexible task assignment.

We have two available assignment policies, one for each primary assignment:

A^c_{12}: Assign server $i \in \{1, 2\}$ to subtask i for the primary assignment. The corresponding throughput is:

$$T^c_{12} = \frac{x_1 x_2 \sum_{k=0}^{B_1+B_2+1} \mu_{11}^k \mu_{22}^{B_1+B_2+1-k}}{x_1 x_2 \sum_{k=0}^{B_1+B_2} \mu_{11}^k \mu_{22}^{B_1+B_2-k} + x_1 \mu_{11}^{B_1+B_2+1} + x_2 \mu_{22}^{B_1+B_2+1}}.$$

A^c_{21}: Assign server 1 to subtask 2, and server 2 to subtask 1 for the primary assignment. The corresponding throughput is:

$$T^c_{21} = \frac{x_1 x_2 \sum_{k=0}^{B_1+B_2+1} \mu_{21}^k \mu_{12}^{B_1+B_2+1-k}}{x_1 x_2 \sum_{k=0}^{B_1+B_2} \mu_{21}^k \mu_{12}^{B_1+B_2-k} + x_1 \mu_{21}^{B_1+B_2+1} + x_2 \mu_{12}^{B_1+B_2+1}}.$$

As in Sections 3.1 and 3.2, in this model, the buffer allocation does not affect the results. Thus, the throughputs of two collaborative task assignment models with different buffer sizes have the same throughput as long as the sums $B_1 + B_2$ of the two buffers are equal. As in Sections 3.1 and 3.2, this property only holds for our two subtasks assumption.

Remark 3.3.1. *The difference of T^c_{12} with buffer size $B_1 + B_2$ and T^c_{12} with buffer size B_1+B_2-1 is positively proportional to $x_1\mu_{22}+x_2\mu_{11}-x_1x_2$. Thus, T^c_{12} is non-decreasing in the buffer sizes B_1 and B_2 if and only if $x_1\mu_{22}+x_2\mu_{11}-x_1x_2 \geq 0$. Similarly, we can obtain that T^c_{21} is non-decreasing in the buffer sizes B_1 and B_2 if and only if $x_1\mu_{12}+x_2\mu_{21}-x_1x_2 \geq 0$. Moreover, when $\beta_1 = \beta_2$, both inequalities can be simplified as $\max\{\alpha, \beta_1, 1-\beta_1\} \leq 1$, and thus both T^c_{12} and T^c_{21} are non-decreasing in the buffer sizes B_1 and B_2 if and only if $\alpha \leq 1$.*

Remark 3.3.1 shows that unlike the static and flexible task assignment approaches, when the servers are collaborative, the long-run average throughputs are no longer always non-decreasing with respect to the sum of the buffers. Intuitively, when servers are not collaborative, blocking would cause a server to idle, and thus reduce the total service rate of the system. And a larger sum of the buffer sizes would reduce the occurrence of blocking, and therefore increase the long-run average throughput of the system. However, when the servers are collaborative with a moderate or high synergy level, we can not only avoid server idling but also take advantage of efficient server collaboration when one of the subtasks is blocked. Thus, the throughput of collaborative task assignment is increasing with respect to the sum of the buffers only if the server collaboration is not efficient (i.e., x_1, x_2 are small relative to the rates of the servers at their primary assignments). When collaboration is efficient (i.e., α is large), it is desirable to take advantage of collaboration to the extent possible, which occurs when B_1, B_2 are small.

Next, we choose the assignment with the larger throughput. For ease of exposition, let

$$C_1 \equiv \sum_{k=0}^{B_1+B_2} \mu_{11}^k \mu_{22}^{B_1+B_2-k} \sum_{j=0}^{k} \mu_{12}^j \mu_{21}^{B_1+B_2-j}; \tag{3.5}$$

$$C_2 \equiv \sum_{k=0}^{B_1+B_2} \mu_{12}^k \mu_{21}^{B_1+B_2-k} \sum_{j=0}^{k} \mu_{11}^j \mu_{22}^{B_1+B_2-j}. \tag{3.6}$$

Note that

$$C_1 - C_2 = \sum_{k=0}^{B_1+B_2} (\mu_{21}\mu_{22})^{B_1+B_2-k} \sum_{j=0}^{k} (\mu_{11}\mu_{12})^j \left[(\mu_{11}\mu_{21})^{k-j} - (\mu_{12}\mu_{22})^{k-j}\right].$$

Therefore, $C_1 \geq C_2$ when $\mu_{11}\mu_{21} \geq \mu_{12}\mu_{22}$.

The following proposition compares T_{12}^c and T_{21}^c.

Proposition 3.3.1. *Assignment A_{12}^c is no worse than A_{21}^c if and only if*

$$x_1x_2[(\mu_{11} - \mu_{21})C_1 + (\mu_{22} - \mu_{12})C_2] \geq (\mu_{11}\mu_{12} - \mu_{21}\mu_{22})(x_1C_1 - x_2C_2). \tag{3.7}$$

Proof. Note that

$$T_{12}^c - T_{21}^c \propto x_1x_2L_1 + x_1L_2 + x_2L_3,$$

where

$$\begin{aligned} L_1 = &\sum_{k=0}^{B_1+B_2+1} \mu_{11}^k \mu_{22}^{B_1+B_2+1-k} \sum_{j=0}^{B_1+B_2} \mu_{21}^j \mu_{12}^{B_1+B_2-j} \\ &- \sum_{k=0}^{B_1+B_2+1} \mu_{21}^k \mu_{12}^{B_1+B_2+1-k} \sum_{j=0}^{B_1+B_2} \mu_{11}^j \mu_{22}^{B_1+B_2-j}; \end{aligned} \tag{3.8}$$

$$L_2 = \mu_{21}^{B_1+B_2+1} \sum_{k=0}^{B_1+B_2+1} \mu_{11}^k \mu_{22}^{B_1+B_2+1-k} - \mu_{11}^{B_1+B_2+1} \sum_{k=0}^{B_1+B_2+1} \mu_{21}^k \mu_{12}^{B_1+B_2+1-k}; \tag{3.9}$$

$$L_3 = \mu_{12}^{B_1+B_2+1} \sum_{k=0}^{B_1+B_2+1} \mu_{11}^k \mu_{22}^{B_1+B_2+1-k} - \mu_{22}^{B_1+B_2+1} \sum_{k=0}^{B_1+B_2+1} \mu_{21}^k \mu_{12}^{B_1+B_2+1-k}. \tag{3.10}$$

Next, we treat L_1, C_1, C_2 as functions of $B_1 + B_2 = n$ and prove that $L_1 = (\mu_{11} - \mu_{21})C_1 + (\mu_{22} - \mu_{12})C_2$ by induction. Note that (3.5) and (3.6) yield

$$\begin{aligned} C_1(n) &= \sum_{k=0}^{n-1} \mu_{11}^k \mu_{22}^{n-k} \sum_{j=0}^{k} \mu_{12}^j \mu_{21}^{n-j} + \mu_{11}^n \sum_{j=0}^{n} \mu_{12}^j \mu_{21}^{n-j} \\ &= \mu_{21}\mu_{22}C_1(n-1) + \mu_{11}^n \sum_{j=0}^{n} \mu_{12}^j \mu_{21}^{n-j}; \end{aligned} \tag{3.11}$$

$$
\begin{aligned}
C_2(n) &= \sum_{k=0}^{n-1} \mu_{12}^k \mu_{21}^{n-k} \sum_{j=0}^{k} \mu_{11}^j \mu_{22}^{n-j} + \mu_{12}^n \sum_{j=0}^{k} \mu_{11}^n \mu_{22}^{n-j} \\
&= \mu_{21}\mu_{22} C_2(n-1) + \mu_{12}^n \sum_{j=0}^{n} \mu_{11}^j \mu_{22}^{n-j}. \qquad (3.12)
\end{aligned}
$$

When $B_1 + B_2 = 0$, then $C_1(0) = C_2(0) = 1$, and

$$
L_1(0) = (\mu_{11} + \mu_{22}) - (\mu_{21} + \mu_{12}) = (\mu_{11} - \mu_{21})C_1(0) + (\mu_{22} - \mu_{12})C_2(0).
$$

Suppose now that $L_1(B_1 + B_2) = (\mu_{11} - \mu_{21})C_1(B_1 + B_2) + (\mu_{22} - \mu_{12})C_2(B_1 + B_2)$ for $B_1 + B_2 = 0, 1, \ldots, n-1$. Then, for $B_1 + B_2 = n$, (3.8), (3.11), and (3.12) yield

$$
\begin{aligned}
L_1(n) &= \mu_{21}\mu_{22} L_1(n-1) + \mu_{11}^{n+1} \sum_{j=0}^{n} \mu_{21}^j \mu_{12}^{n-j} + \mu_{12}^n \mu_{22} \sum_{k=0}^{n} \mu_{11}^k \mu_{22}^{n-k} \\
&- \mu_{12}^{n+1} \sum_{j=0}^{n} \mu_{11}^j \mu_{22}^{n-j} - \mu_{11}^n \mu_{21} \sum_{k=0}^{n} \mu_{12}^k \mu_{21}^{n-k} \\
&= \mu_{21}\mu_{22} \Big[(\mu_{11} - \mu_{21}) C_1(n-1) + (\mu_{22} - \mu_{12}) C_2(n-1) \Big] \\
&+ (\mu_{11} - \mu_{21}) \mu_{11}^n \sum_{j=0}^{n} \mu_{12}^j \mu_{21}^{n-j} + (\mu_{22} - \mu_{12}) \mu_{12}^n \sum_{j=0}^{n} \mu_{11}^j \mu_{22}^{n-j} \\
&= (\mu_{11} - \mu_{21}) C_1(n) + (\mu_{22} - \mu_{12}) C_2(n).
\end{aligned}
$$

Thus, $L_1 = (\mu_{11} - \mu_{21})C_1 + (\mu_{22} - \mu_{12})C_2$.

Note that we can also present the recursive formulas for $C_1(n)$ and $C_2(n)$ as follows:

$$
\begin{aligned}
C_1(n) &= \sum_{k=1}^{n} \mu_{11}^k \mu_{22}^{n-k} \sum_{j=0}^{k-1} \mu_{12}^j \mu_{21}^{n-j} + \mu_{22}^n \mu_{21}^n + \sum_{k=1}^{n} \mu_{11}^k \mu_{22}^{n-k} \mu_{12}^k \mu_{21}^{n-k} \\
&= \mu_{11}\mu_{21} C_1(n-1) + \sum_{k=0}^{n} (\mu_{11}\mu_{12})^k (\mu_{21}\mu_{22})^{n-k}. \qquad (3.13) \\
C_2(n) &= \sum_{k=1}^{n} \mu_{12}^k \mu_{21}^{n-k} \sum_{j=0}^{k-1} \mu_{11}^j \mu_{22}^{n-j} + \mu_{21}^n \mu_{22}^n + \sum_{k=1}^{n} \mu_{12}^k \mu_{21}^{n-k} \mu_{11}^k \mu_{22}^{n-k}
\end{aligned}
$$

$$= \mu_{12}\mu_{22}C_2(n-1) + \sum_{k=0}^{n}(\mu_{11}\mu_{12})^k(\mu_{21}\mu_{22})^{n-k}. \tag{3.14}$$

Similarly, using (3.9), (3.10), (3.13) and (3.14), we can obtain the following equations by induction.

$$L_2(n) = -(\mu_{11}\mu_{12} - \mu_{21}\mu_{22})C_1(n); \tag{3.15}$$

$$L_3(n) = (\mu_{11}\mu_{12} - \mu_{21}\mu_{22})C_2(n). \tag{3.16}$$

When $B_1 + B_2 = 0$, (3.15) and (3.16) hold obviously. Suppose now that (3.15) and (3.16) hold for $B_1 + B_2 = 0, 1, \ldots, n-1$. Then, for $B_1 + B_2 = n$, equations (4.3),(3.13), and (3.14) yield

$$\begin{aligned}
L_2(n) &= \mu_{11}\mu_{21}L_2(n-1) + (\mu_{21}\mu_{22})^{n+1} - (\mu_{11}\mu_{12})^{n+1} \\
&= -(\mu_{11}\mu_{12} - \mu_{21}\mu_{22})\mu_{11}\mu_{21}C_1(n-1) - (\mu_{11}\mu_{12} - \mu_{21}\mu_{22})\sum_{k=0}^{n}(\mu_{11}\mu_{12})^k(\mu_{21}\mu_{22})^{n-k} \\
&= -(\mu_{11}\mu_{12} - \mu_{21}\mu_{22})C_1(n); \\
L_3(n) &= \mu_{12}\mu_{22}L_3(n-1) + (\mu_{11}\mu_{12})^{n+1} - (\mu_{21}\mu_{22})^{n+1} \\
&= (\mu_{11}\mu_{12} - \mu_{21}\mu_{22})\mu_{12}\mu_{22}C_2(n-1) + (\mu_{11}\mu_{12} - \mu_{21}\mu_{22})\sum_{k=0}^{n}(\mu_{11}\mu_{12})^k(\mu_{21}\mu_{22})^{n-k} \\
&= (\mu_{11}\mu_{12} - \mu_{21}\mu_{22})C_2(n).
\end{aligned}$$

The result follows. □

The next proposition shows the optimal server assignment policy for two special cases.

Proposition 3.3.2. *(a) When $\mu_{11} \geq \mu_{21}, \mu_{22} \geq \mu_{12}$, Assignment A_{12}^c is no worse than A_{21}^c.*

(b) When $\alpha = 1$, Assignment A_{12}^c is no worse than A_{21}^c if and only if $\beta_1 \geq \beta_2$.

Proof. For (a), note that $x_1 \geq \mu_{11}, x_2 \geq \mu_{22}$. When $\mu_{11} \geq \mu_{21}, \mu_{22} \geq \mu_{12}$,

$$x_2(\mu_{11} - \mu_{21})x_1C_1 \geq \mu_{22}(\mu_{11} - \mu_{21})x_1C_1 \geq (\mu_{12}\mu_{11} - \mu_{22}\mu_{21})x_1C_1,$$
$$x_1(\mu_{22} - \mu_{12})x_2C_2 \geq \mu_{11}(\mu_{22} - \mu_{12})x_2C_2 \geq (\mu_{21}\mu_{22} - \mu_{11}\mu_{12})x_2C_2.$$

Adding up the above two inequalities, we obtain (3.7), which means that inequality (3.7) always holds when $\mu_{11} \geq \mu_{21}, \mu_{22} \geq \mu_{12}$. Thus, in this case, $T^c_{12} \geq T^c_{21}$.
For (b), if $\alpha = 1$, then $x_1 = \Sigma_1, x_2 = \Sigma_2$, and inequality (3.7) can be simplified as

$$(C_1\Sigma_1 + C_2\Sigma_2)(\beta_1 - \beta_2) \geq 0.$$

Thus the result follows. □

Part (a) of Proposition 3.3.2 shows that when the servers specialize in different subtasks, the primary assignment is assign the servers to the subtasks they are specialized. For part (b), note that $\beta_1 \geq \beta_2 \Leftrightarrow \mu_{11}\mu_{22} \geq \mu_{21}\mu_{12}$. Thus, if the synergy level is 1, i.e., the server collaboration is additive, then we let the servers collaborate for the secondary assignment, and use the primary assignment with the higher product of service rates.

3.3.2 Special Case 1: No Buffers

When $B_1 = B_2 = 0$, then $C_1 = C_2 = 1$ and we can simplify inequality (3.7) as follows:

Corollary 3.3.1. *Assignment A^c_{12} is no worse than A^c_{21} if and only if*

$$x_1x_2(\mu_{11} + \mu_{22} - \mu_{21} - \mu_{12}) \geq (x_1 - x_2)(\mu_{11}\mu_{12} - \mu_{21}\mu_{22}). \tag{3.17}$$

Denote

$$\gamma = \frac{\Sigma_2}{\Sigma_1}$$

as the fraction of the total service capacity at subtask 2 over the total service capacity at subtask 1. Without loss of generality, label the servers so that $\beta_1 \geq \beta_2$. Then $\beta_1 > 0, \beta_2 < 1$

(otherwise the problem reduces to having only one server). Consider the following three cases: (1) $\beta_1 \geq \beta_2 > \frac{1}{2}$; (2) $\beta_1 \geq \frac{1}{2} \geq \beta_2$; (3) $\frac{1}{2} > \beta_1 \geq \beta_2$. The second case is equivalent to $\mu_{11} \geq \mu_{21}, \mu_{22} \geq \mu_{12}$, and the optimal policy is provided in Proposition 3.3.2. For cases (1) and (3), we have that $\beta_1 \neq \frac{1}{2} \neq \beta_2$. Let

$$\begin{aligned}
m_1 &= \frac{2\beta_1 - 1}{2\beta_2 - 1}; \\
m_2 &= \frac{(2\beta_2 - 1)(1 - \beta_1)\beta_1}{(2\beta_1 - 1)(1 - \beta_2)\beta_2}; \\
G_1 &= \frac{(\beta_1 + \beta_2 - 1)(1 - \gamma)}{(2\beta_1 - 1) - (2\beta_2 - 1)\gamma}, \text{ if } \gamma \neq m_1; \\
G_2 &= \frac{(\beta_1 + \beta_2 - 1)\beta_1}{(2\beta_1 - 1)[\beta_1 + (1 - \beta_2)\gamma]}; \\
G_3 &= \frac{(\beta_1 + \beta_2 - 1)(1 - \beta_2)\gamma}{(2\beta_2 - 1)[\beta_1 + (1 - \beta_2)\gamma]}.
\end{aligned}$$

Note that, when $\beta_1 \geq \beta_2 > \frac{1}{2}$, $0 \leq m_2 \leq 1 \leq m_1 < \infty$; when $\frac{1}{2} > \beta_1 \geq \beta_2$, $0 < m_1 \leq 1 \leq m_2 \leq \infty$. Propositions 3.3.3 and 3.3.4 present the results of the comparisons of A_{12}^c and A_{21}^c in cases (1) and (3), respectively. The proof of Proposition 3.3.3 is provided in Appendix A.2.1.

Proposition 3.3.3. *When $\beta_1 \geq \beta_2 > \frac{1}{2}$,*

(a) If $\gamma < m_2$, A_{12}^c is optimal if and only if $\alpha \geq \max\{G_1, G_2\}$.

(b) If $m_2 \leq \gamma \leq m_1$, A_{12}^c is always optimal.

(c) If $\gamma > m_1$, A_{12}^c is optimal if and only if $\alpha \leq G_1$.

When $\beta_1, \beta_2 > \frac{1}{2}$, then $\mu_{11} \geq \mu_{21}, \mu_{12} \geq \mu_{22}$, and hence server 1 is no worse than server 2 at both subtasks. And $\beta_1 \geq \beta_2$ suggests that server 1 is relatively better at subtask 1 than at subtask 2 (relative to server 2). Intuitively, when the synergy level α is high, we might want to take advantage of the high efficiency of the servers when they collaborate by pushing the system to the boundary states where only one subtask is available to work on. Thus, when γ is large (small), and hence there is more total capacity at subtask 2 (1), we will assign the relatively faster server, i.e., server 1, to the subtask with larger total capacity, i.e., subtask 2 (1), when the synergy level is high. Otherwise, when the synergy level is small, we will assign the relatively faster server to the subtask with the smaller total capacity to balance the speed of the two subtasks. Finally, when γ is moderate, so the total service capacities at the two stations are similar, then it is better to assign the faster server to the task the server is relatively better at whenever possible.

Proposition 3.3.4. *When $\frac{1}{2} > \beta_1 \geq \beta_2$,*

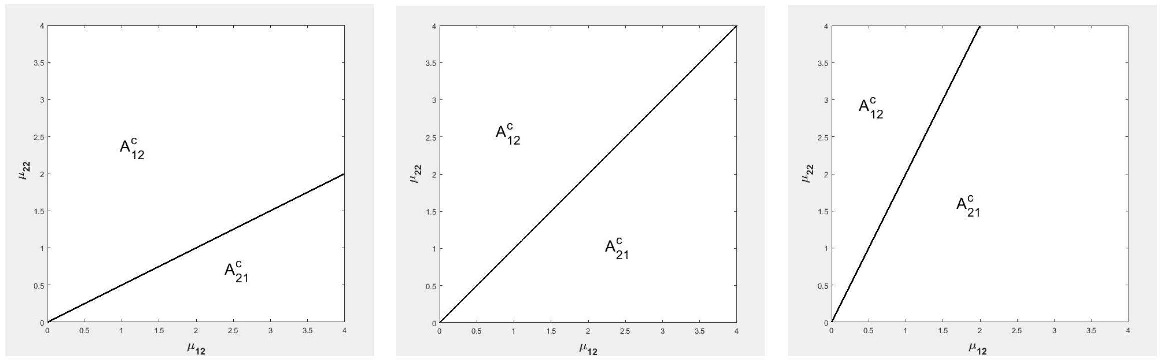
(a) If $\gamma < m_1$, A_{12}^c is optimal if and only if $\alpha \leq G_1$.

(b) If $m_1 \leq \gamma \leq m_2$, A_{12}^c is always optimal.

(c) If $\gamma > m_2$, A_{12}^c is optimal if and only if $\alpha \geq \max\{G_1, G_3\}$.

The interpretation and proof of Proposition 3.3.4 are very similar to that of Proposition 3.3.3 except that server 2 is now better than server 1 at both subtasks and server 2 is relatively better at subtask 2 than at subtask 1 (relative to server 1). Thus we omit the proof for brevity.

Example 3.3.1. *Figure 3.4 shows the optimal primary assignment of the servers for three cases when $\alpha = 1$: (a) $\mu_{11} = 2, \mu_{21} = 1$; (b) $\mu_{11} = \mu_{21} = 1$; (c) $\mu_{11} = 1, \mu_{21} = 2$. For all three cases, Assignment A_{12}^c is optimal if and only if the service rate of server 2 at subtask 2 is above the corresponding line; otherwise Assignment A_{21}^c is optimal.*



(a) $\mu_{11} = 2, \mu_{21} = 1$ (b) $\mu_{11} = 1, \mu_{21} = 1$ (c) $\mu_{11} = 1, \mu_{21} = 2$

Figure 3.4: Optimal collaborative task assignment for different service rates when $\alpha = 1$.

3.3.3 Special Case 2: Asymptotically Infinite Buffers

In this section, consider the case when $B_1 + B_2 \to \infty$. Then,

1. In assignment A^c_{12}, if $\mu_{11} \leq \mu_{22}$, $T^c_{12} \to \frac{x_1\mu_{22}}{x_1+\mu_{22}-\mu_{11}}$; and if $\mu_{11} > \mu_{22}$, $T^c_{12} \to \frac{x_2\mu_{11}}{x_2+\mu_{11}-\mu_{22}}$.
2. In assignment A^c_{21}, if $\mu_{21} \leq \mu_{12}$, $T^c_{21} \to \frac{x_1\mu_{12}}{x_1+\mu_{12}-\mu_{21}}$; and if $\mu_{21} > \mu_{12}$, $T^c_{21} \to \frac{x_2\mu_{21}}{x_2+\mu_{21}-\mu_{12}}$.

The following propositions provide the optimal collaborative task assignment policy when the sum of the buffers goes to infinity. Again, we label the servers so that $\beta_1 \geq \beta_2$, and consider three cases: (1) $\beta_1 \geq \beta_2 > \frac{1}{2}$; (2) $\beta_1 \geq \frac{1}{2} \geq \beta_2$; (3) $\frac{1}{2} > \beta_1 \geq \beta_2$. Case (2) follows from Proposition 3.3.2. For cases (1) and (3), we have $\beta_1 \neq \frac{1}{2} \neq \beta_2$. Let

$$\begin{aligned}
G_4 &= \frac{\beta_1+\beta_2-1}{2\beta_1-1};\\
G_5 &= \frac{\gamma\beta_2(1-\beta_2+\beta_1)-\beta_1(1-\beta_1+\beta_2)}{\gamma\beta_2-\beta_1}, \text{ if } \gamma \neq \frac{\beta_1}{\beta_2};\\
G_6 &= \frac{\beta_2[\beta_1-\gamma(1-\beta_2)]}{2\beta_1-1};\\
G_7 &= \frac{\beta_1+\beta_2-1}{2\beta_2-1};\\
G_8 &= \frac{\gamma(1-\beta_2)(1-\beta_1+\beta_2)-(1-\beta_1)(1-\beta_2+\beta_1)}{\gamma(1-\beta_2)-(1-\beta_1)}, \text{ if } \gamma \neq \frac{1-\beta_1}{1-\beta_2};\\
G_9 &= \frac{(1-\beta_1)[\beta_1-\gamma(1-\beta_2)]}{(2\beta_2-1)\gamma}.
\end{aligned}$$

Note that, when $\beta_1 \geq \beta_2$, we have $0 \leq \frac{1-\beta_1}{1-\beta_2} \leq 1 \leq \frac{\beta_1}{\beta_2} \leq \infty$. Propositions 3.3.5 and 3.3.6 present the results of the comparisons of A_{12}^c and A_{21}^c in cases (1) and (3), respectively. The proof of Proposition 3.3.5 is provided in Appendix A.2.2.

Proposition 3.3.5. *When $\beta_1 \geq \beta_2 > \frac{1}{2}$,*

1. *If $\gamma < \frac{1-\beta_1}{1-\beta_2}$, A_{12}^c is optimal if and only if $\alpha \geq \min\{G_4, \max\{G_5, G_6\}\}$.*
2. *If $\frac{1-\beta_1}{1-\beta_2} \leq \gamma \leq \frac{\beta_1}{\beta_2}$, A_{12}^c is always optimal.*
3. *If $\gamma > \frac{\beta_1}{\beta_2}$, A_{12}^c is optimal if and only if $\alpha \leq \max\{G_5, G_7\}$.*

Proposition 3.3.6. *When $\frac{1}{2} > \beta_1 \geq \beta_2$,*

1. *If $\gamma < \frac{1-\beta_1}{1-\beta_2}$, A_{12}^c is optimal if and only if $\alpha \leq \max\{G_4, G_8\}$.*
2. *If $\frac{1-\beta_1}{1-\beta_2} \leq \gamma \leq \frac{\beta_1}{\beta_2}$, A_{12}^c is always optimal.*
3. *If $\gamma \geq \frac{\beta_1}{\beta_2}$, A_{12}^c is optimal if and only if $\alpha \geq \min\{G_7, \max\{G_8, G_9\}\}$.*

The interpretation and proof of Proposition 3.3.6 are very similar to that of Proposition 3.3.5 except that server 2 is now better than server 1 at both subtasks and server 2 is relatively better at subtask 2 than at subtask 1 (relative to server 1). Thus we omit the proof for brevity.

We observe that the optimal policies for asymptotically infinite buffers have similar structure as in the zero buffer case, and hence have the same intuitions as in the previous section. However, they have more complex thresholds than the zero buffer case because $\frac{C_1}{C_2}$ has multiple possible limits when the sum of the buffers goes to infinity depending on how $\frac{\mu_{11}}{\mu_{22}}$ and $\frac{\mu_{12}}{\mu_{21}}$ compare with 1.

3.4 Other Forms of Server Coordination

In this section, we analyze the other two forms of server coordination mentioned in Section 5.1, namely teamwork and non-collaboration. Note that all of the results in this section hold without the assumption of exponentially distributed service requirement. Thus, the results in this section can be applied to a generalized system with independent and identically distributed service requirements. In Section 3.4.1, we investigate teamwork with or without task partitioning; in Section 3.4.2, we analyze the non-collaboration approach; and in Section 3.4.3, we compare the long-run average throughputs of these methods.

3.4.1 Teamwork

In this section, we analyze the teamwork server coordination approach. The service requirement of each job is $S_1 + S_2$, and the service rate of server i at the combined task is

$$\frac{1}{\frac{1}{\mu_{i1}} + \frac{1}{\mu_{i2}}}$$

for $i = 1, 2$. Thus, the corresponding throughput of teamwork is proportional to the sum of these service rates, namely

$$T^t = \frac{\alpha}{\frac{1}{\mu_{11}} + \frac{1}{\mu_{12}}} + \frac{\alpha}{\frac{1}{\mu_{21}} + \frac{1}{\mu_{22}}}.$$

We also consider teamwork with task partitioning, in which the server team will first complete subtask 1 with a combined service rate of $\alpha\Sigma_1$, and then complete subtask 2 with a combined service rate of $\alpha\Sigma_2$, and repeat the process in this order. The throughput of teamwork with task partitioning is

$$T^{tp} = \frac{\alpha}{\frac{1}{\mu_{11}+\mu_{21}} + \frac{1}{\mu_{12}+\mu_{22}}}.$$

The next result compares teamwork with or without task partitioning.

Proposition 3.4.1. *The throughput of teamwork with task partitioning is never smaller than that of teamwork (without task partitioning).*

Proof. Simple algebra yields that $T^{tp} \geq T^t$ if and only if $(\mu_{11}\mu_{22} - \mu_{12}\mu_{21})^2 \geq 0$. Thus $T^{tp} \geq T^t$ always holds. $\square$

Note that $T^{tp} = T^t$ if $\mu_{11}\mu_{22} = \mu_{12}\mu_{21}$, which holds if servers are identical or if service rates depend only on either the subtask or the station or if the servers are *generalists*, which means that the service rate of server i at subtask j is of the form $\mu_i\gamma_j$. Thus the difference between teamwork with and without task partitioning arises from server specialization. Teamwork with task partitioning takes advantage of the server specialization (and thus avoids extremely low service rates) by combining the service rates at each subtask first. Thus even when servers are not generalists, they do not spend excessive time at subtasks where they have relatively low service rates. Thus teamwork with task partitioning does a better job of neutralizing the servers' weaknesses and leads to higher throughput.

3.4.2 Non-collaboration

In this section, we consider a non-collaborative approach in which servers are working in parallel and each server will complete all subtasks of a job. The throughput of this parallel model is

$$T^{nc} = \frac{1}{\frac{1}{\mu_{11}} + \frac{1}{\mu_{12}}} + \frac{1}{\frac{1}{\mu_{21}} + \frac{1}{\mu_{22}}}.$$

Note that non-collaboration processes multiple jobs in parallel at the same time, while task assignment and teamwork with or without task partitioning process only one job at a time. Thus non-collaboration has more work in process (WIP) than the other server coordination approaches.

3.4.3 Comparison of Non-collaboration and Teamwork

Note that non-collaboration is a special case of teamwork with $\alpha = 1$ for one-station systems. Thus, we have the following proposition:

Proposition 3.4.2. *Teamwork (without task partitioning) is no worse than non-collaboration if and only if* $\alpha \geq 1$.

By comparing the long-run average throughputs of teamwork with task partitioning and non-collaboration, we have the following result:

Proposition 3.4.3. *Teamwork (with task partitioning) is no worse than non-collaboration if and only if*

$$\alpha \geq \frac{\Sigma_1 + \Sigma_2}{\Sigma_1 \Sigma_2} \times \frac{h_1}{h_2} \equiv h, \tag{3.18}$$

where

$$h_1 = \mu_{11}\mu_{21}\Sigma_2 + \mu_{12}\mu_{22}\Sigma_1,$$

$$h_2 = (\mu_{11} + \mu_{12})(\mu_{21} + \mu_{22}).$$

Note that, h can be reorganized as follows:

$$h = \frac{\beta_1(1-\beta_1)\Sigma_1^2 + \beta_2(1-\beta_2)\Sigma_2^2 + (\beta_1 + \beta_2 - \beta_1^2 - \beta_2^2)\Sigma_1\Sigma_2}{\beta_1(1-\beta_1)\Sigma_1^2 + \beta_2(1-\beta_2)\Sigma_2^2 + (\beta_1 + \beta_2 - 2\beta_1\beta_2)\Sigma_1\Sigma_2}. \tag{3.19}$$

It follows that $h \leq 1$, and the equality holds if and only if $\beta_1 = \beta_2 \Leftrightarrow \mu_{11}\mu_{22} = \mu_{12}\mu_{21} \Leftrightarrow T^{tp} = T^t$.

Intuitively, servers are working separately in non-collaboration, so when the server collaboration is not too inefficient (α is not too small), teamwork with or without task partitioning would be better since it takes the advantage of server collaboration. Moreover, teamwork with task partitioning can outperform non-collaboration even when collaboration is somewhat inefficient (when $h < \alpha < 1$) because it takes better advantage of server specialization. However, if the relative advantage of server 1 over server 2 at both subtasks are equal (i.e., $\beta_1 = \beta_2$), the benefit of server specialization for teamwork with task partitioning no longer exists, and the comparison of teamwork with task partitioning and non-collaboration depends solely on whether the server collaboration is efficient or not.

3.5 Best Server Coordination Methods

In this section, we compare the server coordination methods we discussed in the previous sections, and determine how we should choose from these approaches for one queueing station.

Recall that we separate the task assignment approaches based on server flexibility and collaboration levels, namely static, flexible but not collaborative, and flexible and collaborative. Moreover, servers need to be flexible and collaborative for teamwork with and without task partitioning and flexible for non-collaboration. Table 3.1 summarize the requirements on server flexibility and collaboration for the six server coordination methods we consider. A check-mark ✓ indicates that the method requires the specified server flexibility and collaboration levels.

Table 3.1: Applicability of server coordination approaches

Approach	Flexible Servers	Collaborative Servers
static task assignment		
flexible task assignment	✓	
collaborative task assignment	✓	✓
teamwork without task partitioning	✓	✓
teamwork with task partitioning	✓	✓
non-collaboration	✓	

As Table 3.1 shows, if the servers are static, only static task assignment is applicable; if the servers are flexible but not collaborative, only static and flexible task assignment and non-collaboration are applicable; and if the servers are flexible and collaborative, all six server coordination methods are applicable.

We have proved that in one-station systems, teamwork with task partitioning is never worse than teamwork without task partitioning. As these approaches are applicable in the same settings, we will only consider teamwork with task partitioning in this section. Moreover, static and flexible task assignment are special cases of collaborative task assignment. Thus, we will compare collaborative task assignment with teamwork with task partitioning and non-collaboration in Section 3.5.1 to identify the best possible server coordination approach. In Section 3.5.2, we compare flexible task assignment and non-collaboration to identify the best server coordination method when the servers are flexible but not collaborative. The comparison of static and flexible task assignment with other server coordination methods can be found in Appendices A.4.1 and A.4.2, respectively. Finally, in Sections 3.5.3 and 3.5.4, we consider two special cases, namely when the servers are generalists and when the servers are specialists, respectively.

3.5.1 Best Server Coordination Methods for Flexible and Collaborative Servers

In this section, we consider the case when servers are not only flexible, but also can work together with a combined service rate equal to the sum of individual service rates times the synergy factor $\alpha > 0$. We start by comparing teamwork with task partitioning and collaborative task assignment.

Proposition 3.5.1. *Teamwork with task partitioning is no worse than collaborative task assignment if and only if* $\alpha \geq 1 + |\beta_1 - \beta_2|$.

Proof. First, we prove that when $\alpha < \max\{\beta_1, 1-\beta_1, \beta_2, 1-\beta_2\}$, teamwork with task partitioning is worse than collaborative task assignment. Recall that $x_1 = \max\{\mu_{11}, \mu_{21}, \alpha\Sigma_1\}$, $x_2 = \max\{\mu_{12}, \mu_{22}, \alpha\Sigma_2\}$, and thus $\alpha\Sigma_1 \leq x_1$, $\alpha\Sigma_2 \leq x_2$. When $\alpha < \max\{\beta_1, 1 -$

$\beta_1, \beta_2, 1-\beta_2\}$, at least one of the inequalities $\alpha\Sigma_1 \leq x_1$ and $\alpha\Sigma_2 \leq x_2$ is strict. Thus,

$$T^{tp} = \frac{1}{\frac{1}{\alpha\Sigma_1} + \frac{1}{\alpha\Sigma_2}} < \frac{1}{\frac{1}{x_1} + \frac{1}{x_2}} = \frac{x_1 x_2}{x_1 + x_2}.$$

Moreover,

$$\frac{x_1 x_2}{x_1 + x_2} - T_{12}^c \propto (x_1 x_2 - x_1 \mu_{22} - x_2 \mu_{11}) \sum_{k=0}^{B_1+B_2} \mu_{11}^k \mu_{22}^{B_1+B_2-k}, \tag{3.20}$$

$$\frac{x_1 x_2}{x_1 + x_2} - T_{21}^c \propto (x_1 x_2 - x_1 \mu_{12} - x_2 \mu_{21}) \sum_{k=0}^{B_1+B_2} \mu_{21}^k \mu_{12}^{B_1+B_2-k}. \tag{3.21}$$

Since $\alpha < \max\{\beta_1, 1-\beta_1, \beta_2, 1-\beta_2\}$, at least one of the following four cases hold: (1) $x_1 = \mu_{11}$, (2) $x_1 = \mu_{21}$, (3) $x_2 = \mu_{12}$, or (4) $x_2 = \mu_{22}$. Thus, at least one of equations (3.20) and (3.21) is non-positive, and hence

$$T^{tp} < \frac{x_1 x_2}{x_1 + x_2} \leq \max\{T_{12}^c, T_{21}^c\}.$$

We conclude that teamwork with task partitioning is worse than collaborative task assignment.

Next, when $\alpha \geq \max\{\beta_1, 1-\beta_1, \beta_2, 1-\beta_2\}$, we have $x_1 = \alpha\Sigma_1$ and $x_2 = \alpha\Sigma_2$. Then, equations equations (3.20) and (3.21) yield

$$T^{tp} - T_{12}^c \propto (\alpha\Sigma_1\Sigma_2 - \Sigma_1\mu_{22} - \Sigma_2\mu_{11}) \sum_{k=0}^{B_1+B_2} \mu_{11}^k \mu_{22}^{B_1+B_2-k},$$

$$T^{tp} - T_{21}^c \propto (\alpha\Sigma_1\Sigma_2 - \Sigma_1\mu_{12} - \Sigma_2\mu_{21}) \sum_{k=0}^{B_1+B_2} \mu_{21}^k \mu_{12}^{B_1+B_2-k}.$$

Therefore,

$$T^{tp} - T_{12}^c \geq 0 \Leftrightarrow \alpha \geq \frac{\Sigma_1\mu_{22} + \Sigma_2\mu_{11}}{\Sigma_1\Sigma_2} = 1 - \beta_2 + \beta_1,$$
$$T^{tp} - T_{21}^c \geq 0 \Leftrightarrow \alpha \geq \frac{\Sigma_1\mu_{12} + \Sigma_2\mu_{21}}{\Sigma_1\Sigma_2} = 1 - \beta_1 + \beta_2.$$

It follows that

$$T^{tp} \geq \max\{T^c_{12}, T^c_{21}\} \Leftrightarrow \alpha \geq 1 + |\beta_1 - \beta_2|.$$

Hence, teamwork with task partitioning is no worse than collaborative task assignment if and only if $\alpha \geq 1 + |\beta_1 - \beta_2|$. $\square$

Note that $1 + |\beta_1 - \beta_2| \geq 1$ and a larger value of $|\beta_1 - \beta_2|$ indicates a higher specialization level of the servers. Therefore, Proposition 3.5.1 indicates that teamwork with task partitioning is preferable to collaborative task assignment if the server collaboration is efficient (i.e., $\alpha > 1$) and the servers are not heavily specialized.

Next, we compare non-collaboration and collaborative task assignment. The proof of Proposition 3.5.2 is provided in Appendix A.3.1.

Proposition 3.5.2. *Let*

$$\begin{aligned} D_1 \equiv & \Big[\mu_{11}^{B_1+B_2+2}(\mu_{21}+\mu_{22}) + \mu_{22}^{B_1+B_2+2}(\mu_{11}+\mu_{12}) \\ & + (\mu_{11}\mu_{22} - \mu_{21}\mu_{12})\mu_{11}\mu_{22} \sum_{k=0}^{B_1+B_2-1} \mu_{11}^k \mu_{22}^{B_1+B_2-1-k}\Big] \\ & \times \frac{x_1 x_2}{h_1(\mu_{11}^{B_1+B_2+1}x_1 + \mu_{22}^{B_1+B_2+1}x_2)}, \end{aligned}$$

$$\begin{aligned} D_2 \equiv & \Big[\mu_{21}^{B_1+B_2+2}(\mu_{11}+\mu_{12}) + \mu_{12}^{B_1+B_2+2}(\mu_{21}+\mu_{22}) \\ & + (\mu_{21}\mu_{12} - \mu_{11}\mu_{22})\mu_{21}\mu_{12} \sum_{k=0}^{B_1+B_2-1} \mu_{21}^k \mu_{12}^{B_1+B_2-1-k}\Big] \\ & \times \frac{x_1 x_2}{h_1(\mu_{21}^{B_1+B_2+1}x_1 + \mu_{12}^{B_1+B_2+1}x_2)}, \end{aligned}$$

where $x_1 = \max\{\mu_{11}, \mu_{21}, \alpha\Sigma_1\}, x_2 = \max\{\mu_{12}, \mu_{22}, \alpha\Sigma_2\}$. *Then, collaborative task assignment is no worse than non-collaboration if and only if*

$$\max\{D_1, D_2\} \geq 1.$$

Observe that $\max\{D_1, D_2\} > 0$, and D_1, D_2 can be either greater or less than 1. For instance, when $B_1 = B_2 = 0, \alpha = \frac{1}{2}$, if $\mu_{11} = \mu_{22} = 3, \mu_{21} = \mu_{12} = 1$, then collaborative task assignment is better; if $\mu_{11} = \mu_{22} = \mu_{21} = \mu_{12} = 1$, then non-collaboration is better. Intuitively, when the servers are highly specialized at different subtasks, we prefer task assignment even when server collaboration is not efficient since it takes advantage of the high server specialization level. However, if the servers are not highly specialized, we prefer non-collaboration since it avoids blocking.

Moreover, let $D_1(\alpha), D_2(\alpha)$ be the values of D_1, D_2 as functions of α. Then both $D_1(\alpha)$ and $D_2(\alpha)$ are non-decreasing in α, and when α is sufficiently large, both $D_1(\alpha)$ and $D_2(\alpha)$ will be linear in α. Thus, $\lim_{\alpha\to\infty} D_i(\alpha) \to \infty$ for $i = 1, 2$. Therefore, Proposition 3.5.2 implies that collaborative task assignment is better than non-collaboration when α is large enough since only collaborative task assignment takes advantage of efficient server collaboration.

The following proposition concludes the comparisons in this section. Its proof is provided in Appendix A.3.2.

Proposition 3.5.3. *Let* $\alpha_0 = \min\{\max\{\beta_1, 1-\beta_1\}, \max\{\beta_2, 1-\beta_2\}\}$, *then*

1. *when* $\alpha < 1 + |\beta_1 - \beta_2|$,
   (a) *if* $\max\{D_1(\alpha_0), D_2(\alpha_0)\} < 1$, *there exists a unique* $\alpha^* \in (\alpha_0, h]$ *such that* $\max\{D_1(\alpha^*), D_2(\alpha^*)\} = 1$, *and*
      i. *when* $\alpha < \alpha^*$, *non-collaboration is optimal;*
      ii. *when* $\alpha^* \leq \alpha < 1 + |\beta_1 - \beta_2|$, *collaborative task assignment is optimal;*
   (b) *if* $\max\{D_1(\alpha_0), D_2(\alpha_0)\} \geq 1$, *collaborative task assignment is optimal;*
2. *when* $\alpha \geq 1 + |\beta_1 - \beta_2|$, *teamwork with task partitioning is optimal.*

Intuitively, the value of $\max\{D_1(\alpha_0), D_2(\alpha_0)\}$ provides information on server specialty level. As we can see from our previous example, $\max\{D_1(\alpha_0), D_2(\alpha_0)\} > 1$ when the servers are highly specialized, and $\max\{D_1(\alpha_0), D_2(\alpha_0)\} \leq 1$ when the servers are not highly specialized. Thus, Proposition 3.5.3 indicates that when servers are not highly specialized, we prefer non-collaboration when the synergy level is low since it avoids inefficient server collaboration and blocking; we prefer collaborative task assignment when the synergy level is moderate since it takes the advantages of both server synergy and specialty (as long as the servers are not identical); and we prefer teamwork when the synergy level is high since it takes full advantage of efficient server collaboration. On the other hand, when the servers are highly specialized, we prefer collaborative task assignment when the synergy level is not high since it takes full advantage of server specialty and avoids assigning the servers to the subtasks they are not specialized in; and we prefer teamwork when the synergy level is high since it takes full advantage of efficient server collaboration.

3.5.2 Best Server Coordination Methods for Flexible and Non-collaborative Servers

In this section, we compare the two methods that are applicable when the servers are flexible but not collaborative, namely flexible task assignment and non-collaboration (static task assignment is also applicable, but it is a special case of flexible task assignment). Without loss of generality, label the servers such that $\mu_{11} \geq \mu_{21}$. Then the following proposition describes the optimal assignment of flexible but not collaborative servers.

Proposition 3.5.4. *Label the servers such that $\mu_{11} \geq \mu_{21}$. Let*

$$\begin{aligned}
D_3 \equiv & \frac{h_2 \mu_{11}\mu_{22} \sum_{k=0}^{B_1+B_2+1} \mu_{11}^k \mu_{22}^{B_1+B_2+1-k}}{h_1 \sum_{k=0}^{B_1+B_2+2} \mu_{11}^k \mu_{22}^{B_1+B_2+2-k}}; \\
D_4 \equiv & \frac{h_2 \mu_{11}\mu_{12} \sum_{k=0}^{B_1+B_2+1} \mu_{11}^k \mu_{22}^{B_1+B_2+1-k}}{h_1 \big(\sum_{k=0}^{B_1+B_2} \mu_{11}^k \mu_{22}^{B_1+B_2-k} \mu_{11}\mu_{12} + \mu_{11}^{B_1+B_2+2} + \mu_{12}\mu_{22}^{B_1+B_2+1} \big)}; \\
D_5 \equiv & \frac{h_2 \mu_{11}\mu_{12} \sum_{k=0}^{B_1+B_2+1} \mu_{21}^k \mu_{12}^{B_1+B_2+1-k}}{h_1 \big(\sum_{k=0}^{B_1+B_2} \mu_{21}^k \mu_{12}^{B_1+B_2-k} \mu_{11}\mu_{12} + \mu_{11}\mu_{21}^{B_1+B_2+1} + \mu_{12}^{B_1+B_2+2} \big)}.
\end{aligned}$$

Flexible task assignment is no worse than non-collaboration if and only if

$$\max\{D_3, D_4, D_5\} \geq 1.$$

Proof. When $\mu_{11} \geq \mu_{21}$, there are three available flexible task assignment policies, namely $A_{12}^{f}, A_{12}^{1f}, A_{21}^{1f}$. Thus, for flexible task assignment to be no worse than non-collaboration, the throughput of the optimal flexible task assignment needs to be no lower than the throughput of non-collaboration, i.e., $\max\{T_{12}^{f}, T_{12}^{1f}, T_{21}^{1f}\} \geq T^{nc}$. Reorganizing this inequality yields the desired result. We omit the details for reasons of brevity. □

Note that D_3, D_4, D_5 can be either greater or less than 1. For example, when $B_1 = B_2 = 0$, if $\mu_{11} = \mu_{12} = 2, \mu_{22} = 1, \mu_{21} = \frac{1}{8}$, then flexible task assignment is better; if $\mu_{11} = \mu_{12} = 2, \mu_{22} = 1, \mu_{21} = \frac{1}{2}$, then non-collaboration is better. Intuitively, when one server has an extremely low service rate at some station compared to the other server, flexible task assignment is better since it can avoid this low service rate; otherwise, non-collaboration is better since it avoids blocking.

3.5.3 Special Case: Generalists

In this section, we consider the special case when the servers are generalists. Recall that servers are called *generalists* if $\mu_{ij} = \mu_i \gamma_j$ for any i, j, where μ_i can be regarded as the ability of server i, and γ_j represents the difficulty of subtask j. When the servers are generalists, $\mu_{11}\mu_{22} = \mu_{21}\mu_{12}$, and thus $\beta_1 = \beta_2$. The following proposition provides the best server coordination method for one-station systems when $\beta_1 = \beta_2$ and servers are flexible and collaborative (so that all six server coordination methods are applicable). Its proof can be found in Appendix A.3.3.

Proposition 3.5.5. *When $\beta_1 = \beta_2$,*

(1) The long-run average throughputs of teamwork with task partitioning and teamwork without task partitioning are equal;

(2) If $\alpha > 1$, teamwork is optimal; if $\alpha < 1$, non-collaboration is optimal;

(3) If $\alpha = 1$, collaborative task assignment, teamwork with or without task partitioning,

and non-collaboration are equivalent, and are no worse than static and flexible task assignment.

Intuitively, when the servers are generalists, a server's rate at any subtask is proportional to his individual ability that is unaffected by the subtask. That is, if one server is better at some subtask than the other server, he will also be better at the other subtask. Thus, there is no advantage to assign the servers to different subtasks (i.e., task assignment). When the synergy level is higher than 1, we want to take advantage of this efficient server collaboration, and thus we prefer teamwork; and when the synergy level is less than 1, we let the servers work in parallel (i.e., non-collaboration), since they do not have specialties at different subtasks and the faster server will not be blocked by the slower server. When the synergy level is 1, the combined service rate is additive, and there is no loss or gain from either server collaboration or server speciality. Thus, collaborative task assignment, teamwork with or without task partitioning, and non-collaboration are equivalent in this case.

In order to quantify the comparisons of the six server coordination methods, we provide numerical results. Specifically, we compute the throughputs of the different approaches we have discussed with different values of the synergy factor α and service rates with the servers being generalists. We choose four sets of service rates, namely the cases where (i) servers are identical and the task difficulties at both subtasks are the same ($\mu_1 = \mu_2, \gamma_1 = \gamma_2$), (ii) servers are identical but the task difficulties are not the same ($\mu_1 = \mu_2, \gamma_1 \neq \gamma_2$), (iii) one server is faster than the other at both subtasks while both subtasks have the same difficulty ($\mu_1 \neq \mu_2, \gamma_1 = \gamma_2$), and (vi) the server abilities and task difficulties are different for different servers and subtasks ($\mu_1 \neq \mu_2, \gamma_1 \neq \gamma_2$). We also choose five values for α, including cases when server collaboration is synergistic, additive, and inefficient. Recall that B_i is the internal buffer size of subtask i for $i = 1, 2$, and that the buffer allocation does not affect the throughput as long as the sum $B_1 + B_2$ remains unchanged. Thus, we choose three values of $B_1 + B_2$ representing small, medium, and large buffer sizes. The results are given in Table 5.5, where S, F, C stand for static, flexible, collaborative task assignment, respectively, T for teamwork, and NC for non-collaboration. Since in this case, the throughputs of the two teamwork methods are equal, we do not include a separate column for teamwork with task partitioning in Table 5.5.

Table 3.2: Throughputs of server coordination methods with generalists (the highest throughputs in each row are in bold).

	service rates				$B_1 + B_2 = 0$			$B_1 + B_2 = 10$			$B_1 + B_2 \to \infty$				
α	μ_{11}	μ_{21}	μ_{12}	μ_{22}	S	F	C	S	F	C	S	F	C	T	NC
1.5	1	1	1	1	0.67	0.67	1.20	0.92	0.92	1.03	1.00	1.00	1.00	**1.50**	1.00
1.5	2	2	1	1	0.86	0.86	1.64	1.00	1.00	1.50	1.00	1.00	1.50	**2.00**	1.33
1.5	2	1	2	1	0.86	1.20	1.80	1.00	1.33	1.64	1.00	1.33	1.64	**2.25**	1.50
1.5	1	2	2	4	1.33	1.60	2.50	1.85	1.92	2.40	2.00	2.00	2.40	**3.00**	2.00
1.2	1	1	1	1	0.67	0.67	1.09	0.92	0.92	1.01	1.00	1.00	1.00	**1.20**	1.00
1.2	2	2	1	1	0.86	0.86	1.47	1.00	1.00	1.41	1.00	1.00	1.41	**1.60**	1.33
1.2	2	1	2	1	0.86	1.20	1.64	1.00	1.33	1.57	1.00	1.33	1.57	**1.80**	1.50
1.2	1	2	2	4	1.33	1.60	2.22	1.85	1.92	2.18	2.00	2.00	2.18	**2.40**	2.00
1.0	1	1	1	1	0.67	0.67	**1.00**	0.92	0.92	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
1.0	2	2	1	1	0.86	0.86	**1.33**	1.00	1.00	**1.33**	1.00	1.00	**1.33**	**1.33**	**1.33**
1.0	2	1	2	1	0.86	1.20	**1.50**	1.00	1.33	**1.50**	1.00	1.33	**1.50**	**1.50**	**1.50**
1.0	1	2	2	4	1.33	1.60	**2.00**	1.85	1.92	**2.00**	**2.00**	**2.00**	**2.00**	**2.00**	**2.00**
0.8	1	1	1	1	0.67	0.67	0.89	0.92	0.92	0.98	**1.00**	**1.00**	**1.00**	0.80	**1.00**
0.8	2	2	1	1	0.86	0.86	1.17	1.00	1.00	1.23	1.00	1.00	1.23	1.07	**1.33**
0.8	2	1	2	1	0.86	1.20	1.33	1.00	1.33	1.41	1.00	1.33	1.41	1.20	**1.50**
0.8	1	2	2	4	1.33	1.60	1.78	1.85	1.92	1.96	**2.00**	**2.00**	**2.00**	1.60	**2.00**
0.5	1	1	1	1	0.67	0.67	0.67	0.92	0.92	0.92	**1.00**	**1.00**	**1.00**	0.50	**1.00**
0.5	2	2	1	1	0.86	0.86	0.86	1.00	1.00	1.00	1.00	1.00	1.00	0.67	**1.33**
0.5	2	1	2	1	0.86	1.20	1.20	1.00	1.33	1.33	1.00	1.33	1.33	0.75	**1.50**
0.5	1	2	2	4	1.33	1.60	1.60	1.85	1.92	1.92	**2.00**	**2.00**	**2.00**	1.00	**2.00**

Observe that the results in Table 5.5 are consistent with our results in Proposition 3.5.5. That is, when $\alpha \geq 1$, teamwork is the best; when $\alpha \leq 1$, non-collaboration is the best; and when $\alpha = 1$, collaborative task assignment is the best. In addition, Table 5.5 yields the following new observations about static, flexible and collaborative task assignment with generalist servers:

1. When the servers are identical (cases (i) and (ii)), static task assignment is equivalent to flexible task assignment. When servers are not identical (cases (iii) and (iv)), by comparing the static and flexible task assignment, we conclude that server flexibility increases the throughput of the task assignment approach significantly (from 20% to

40% in our examples), especially when the buffer sizes are small.

2. We know that the long-run average throughputs of the static and flexible task assignment approaches are increasing with respect to the buffer sizes. Table 5.5 shows that the convergence speeds for both task assignment approaches are fast. Specifically, for cases (ii) and (iii), the throughputs of static and flexible task assignments already reach the maximum values (as when $B_1 + B_2 \to \infty$) when $B_1 + B_2 = 10$; and for cases (i) and (iv), letting $B_1 + B_2 = 10$ increases the throughputs of static and flexible task assignments by 20% to 39% relative to $B_1 + B_2 = 0$, and yields throughputs that are very close to their upper bounds (with deviation less than 8% relative to $B_1 + B_2 \to \infty$).

3. Allowing collaboration increases the throughput of the task assignment approach when server collaboration is not too inefficient (i.e., when α is not too small) as long as the combined service rate exceeds the maximum of individual service rates (e.g., $\alpha = 0.8$). And this improvement can be large, and increases with the synergy level. Specifically, the throughputs of collaborative task assignment are over 10% higher than for flexible task assignment, even when the synergy level is less than 1 (i.e., $\alpha = 0.8$) when the buffers are zero. However, this improvement decreases as the sum of the buffer sizes increases (because blocking is less frequent for larger buffer sizes).

4. The throughput of collaborative task assignment is non-decreasing with respect to the buffer sizes when $\alpha \leq 1$, and non-increasing when $\alpha > 1$. This result is consistent with Remark 3.3.1, since larger internal buffers lead to less collaboration of the two servers. And the convergence speed with respect to $B_1 + B_2$ is fast. In particular, for cases (ii), (iii), and the $\alpha \geq 1$ cases of (iv), the throughput of collaborative task assignment already reaches the maximum (minimum) value (as when $B_1 + B_2 \to \infty$) when $B_1 + B_2 = 10$; and for the other cases, the throughput of collaborative task

assignment is very close to its extreme value when $B_1 + B_2 = 10$ (with deviation less than 8% relative to $B_1 + B_2 \to \infty$).

5. When $\alpha \leq 1$ and $B_1 + B_2 \to \infty$, all three task assignment approaches are the best for cases (i) and (iv). Indeed, when the synergy level of server collaboration is low, the advantage of server collaboration through secondary assignment vanishes. Therefore, we prefer more balanced service rates at different subtasks for the primary assignment to avoid blocking. That is, for all three task assignment approaches, we will assign server i to subtask $3 - i$ for $i = 1, 2$ for the optimal primary assignments of case (iv). Then, the service rates of all subtasks are equal in cases (i) and (iv) when both subtasks are working, which reduces the occurrence of blocking. When the sum of the buffer sizes goes to infinity, the possibility of blocking is further reduced, and servers will work individually at the same rate and remain static almost all of the time. Thus, the three task assignment approaches all yield the same throughputs as non-collaboration as $B_1 + B_2 \to \infty$.

3.5.4 Special Case: Specialists

We have discussed the special case when servers are generalists in Section 3.5.3. In this section, we want to consider another special case, namely when servers are specialists. Servers are called *specialists* if they have higher service rates at different subtasks. In this section, the servers are labeled so that $\mu_{11} \geq \mu_{21}, \mu_{12} \leq \mu_{22}$.

Note that by Propositions 3.1.1 and 3.2.1, when $\mu_{11} \geq \mu_{21}, \mu_{12} \leq \mu_{22}$, then $A^s_{12} = A^f_{12}$ are the optimal static and flexible task assignment approaches. Specifically, we will assign server i to subtask i for $i = 1, 2$ all the time in both the static and the flexible task assignments.

We now compare our six server coordination approaches with specialist servers via numerical results. We choose four sets of service rates with the specialization level of servers at different subtasks from low to high, and use the case with identical service rates as a benchmark in our comparison. And we also choose five values for α, including cases where server collaboration is synergistic, additive, and inefficient. The results are given in Table 5.6. The notation and abbreviations are as defined in Section 3.5.3. And we have an extra column for teamwork with task partitioning (TP) since it is no longer equivalent to teamwork without task partitioning (as in the previous section).

Table 3.3: Throughputs of server coordination methods with specialists (the highest throughputs in each row are in bold, and we put the best task assignment methods for the finite buffers cases, i.e., when $B_1 + B_2 = 0$, and $B_1 + B_2 = 10$, in italics if they beat teamwork with and without task partitioning and non-collaboration).

	service rates				$B_1+B_2=0$			$B_1+B_2=10$			$B_1+B_2\to\infty$					
α	μ_{11}	μ_{21}	μ_{12}	μ_{22}	S	F	C	S	F	C	S	F	C	TP	T	NC
1.5	1	1	1	1	0.67	0.67	1.20	0.92	0.92	1.03	1.00	1.00	1.00	**1.50**	**1.50**	1.00
1.5	2	1	1	2	1.33	1.33	2.12	1.85	1.85	2.02	2.00	2.00	2.00	**2.25**	2.00	1.33
1.5	3	1	1	3	2.00	2.00	**3.00**	2.77	2.77	**3.00**	**3.00**	**3.00**	**3.00**	**3.00**	2.25	1.50
1.5	4	1	1	4	2.67	2.67	*3.87*	3.69	3.69	*3.98*	**4.00**	**4.00**	**4.00**	3.75	2.40	1.60
1.2	1	1	1	1	0.67	0.67	1.09	0.92	0.92	1.01	1.00	1.00	1.00	**1.20**	**1.20**	1.00
1.2	2	1	1	2	1.33	1.33	*1.89*	1.85	1.85	*1.98*	**2.00**	**2.00**	**2.00**	1.80	1.60	1.33
1.2	3	1	1	3	2.00	2.00	*2.67*	2.77	2.77	*2.94*	**3.00**	**3.00**	**3.00**	2.40	1.80	1.50
1.2	4	1	1	4	2.67	2.67	*3.43*	3.69	3.69	*3.89*	**4.00**	**4.00**	**4.00**	3.00	1.92	1.60
1.0	1	1	1	1	0.67	0.67	**1.00**	0.92	0.92	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
1.0	2	1	1	2	1.33	1.33	*1.71*	1.85	1.85	*1.95*	**2.00**	**2.00**	**2.00**	1.50	1.33	1.33
1.0	3	1	1	3	2.00	2.00	*2.40*	2.77	2.77	*2.88*	**3.00**	**3.00**	**3.00**	2.00	1.50	1.50
1.0	4	1	1	4	2.67	2.67	*3.08*	3.69	3.69	*3.81*	**4.00**	**4.00**	**4.00**	2.50	1.60	1.60
0.8	1	1	1	1	0.67	0.67	0.89	0.92	0.92	0.98	**1.00**	**1.00**	**1.00**	0.80	0.80	**1.00**
0.8	2	1	1	2	1.33	1.33	*1.50*	1.85	1.85	*1.89*	**2.00**	**2.00**	**2.00**	1.20	1.07	1.33
0.8	3	1	1	3	2.00	2.00	*2.09*	2.77	2.77	*2.80*	**3.00**	**3.00**	**3.00**	1.60	1.20	1.50
0.8	4	1	1	4	*2.67*	*2.67*	*2.67*	*3.69*	*3.69*	*3.69*	**4.00**	**4.00**	**4.00**	2.00	1.28	1.60
0.5	1	1	1	1	0.67	0.67	0.67	0.92	0.92	0.92	**1.00**	**1.00**	**1.00**	0.50	0.50	**1.00**
0.5	2	1	1	2	*1.33*	*1.33*	*1.33*	*1.85*	*1.85*	*1.85*	**2.00**	**2.00**	**2.00**	0.75	0.67	1.33
0.5	3	1	1	3	*2.00*	*2.00*	*2.00*	*2.77*	*2.77*	*2.77*	**3.00**	**3.00**	**3.00**	1.00	0.75	1.50
0.5	4	1	1	4	*2.67*	*2.67*	*2.67*	*3.69*	*3.69*	*3.69*	**4.00**	**4.00**	**4.00**	1.25	0.80	1.60

We have the following conclusions from Table 5.6:

1. Collaborative task assignment is the best *except* when server synergy is high, server specialization is low, and buffer sizes are small. Intuitively, when servers are specialists, we would like to take advantage of their specialty by assigning them to the subtasks they are better at unless the server synergy level is high enough to outweigh their specialization.

2. Teamwork with task partitioning is the best when the synergy level is high and the

specialization level is low.

3. Server collaboration improves the long-run average throughput of task assignment when the collaboration is not too inefficient (i.e., when $\alpha \geq 0.8$ and thus $\alpha \sum_j \mu_{ij} \geq \max_j\{\mu_{ij}\}$ for $i = 1, 2.$), and this improvement gets larger as the synergy level gets higher. However, unlike in the generalists case, this improvement vanishes as the sum of the buffers goes to infinity. Intuitively, when the service rate at both subtasks are the same for the primary assignment and the sum of the buffer sizes is large, the probability of any of the subtasks getting blocked is small, and thus the differences among the static, flexible, and collaborative task assignment are small. Furthermore, the throughputs of all three task assignment approaches do not depend on the synergy level when the sum of the buffers goes to infinity. The intuition is similar to that of cases (i) and (iv) when the servers are generalists as the possibility of blocking vanishes with large buffers and balanced service rates.

4. The throughput of collaborative task assignment is decreasing with respect to the buffer sizes when the synergy level is high and the specialty level of servers is not large, which coincides with the cases when teamwork with task partitioning is the best among all methods. Otherwise, the throughput of collaborative task assignment is non-decreasing with respect to the buffer sizes even when server collaboration is efficient. Intuitively, a moderate and balanced service rate at both subtasks for the primary assignment is preferable to a single high service rate at one subtask for the secondary assignment for collaborative task assignment. When $\alpha = 1.5$, and the specialty level $\frac{\mu_{11}}{\mu_{21}} = 3$, the throughput of collaborative task assignment is a constant with respect to the buffer sizes. This result is consistent with Remark 3.3.1 since in this case, we have $x_1 = x_2 = 6$, and thus $x_1\mu_{22} + x_2\mu_{11} - x_1x_2 = 0$. Moreover, the convergence speed is fast (with deviation less than 8% for $B_1 + B_2 = 10$ relative to $B_1 + B_2 \to \infty$) especially when the synergy level is not too low.

5. Similar to the generalists case, the throughputs of static and flexible task assignment increase quickly with respect to the sum of the internal buffer sizes. Specifically, by increasing the sum of the buffers from 0 to 10, we increase the throughputs of static and flexible task assignment approaches significantly (around 35%); and when $B_1 + B_2 = 10$, the long-run average throughputs of the three task assignment approaches are already close to the asymptotically-infinite-buffer throughputs (with deviations less than 10%).

6. The advantage of teamwork with task partitioning over teamwork without task partitioning increases as the specialty level increases. Specifically, when the specialty level $\frac{\mu_{11}}{\mu_{21}}$ equals 2, 3, 4, the throughputs of teamwork with task partitioning are $12.5\%, 33.3\%$, 56.3% higher than for teamwork without task partitioning, respectively.

7. Non-collaboration can be as good or better than static and flexible task assignments when the specialty level is low and buffer sizes are small, and better than teamwork with or without task partitioning when the specialty and synergy levels are low.

3.6 Server Coordination in Longer Lines

When there are no precedence relationships among tasks, then all tasks can be completed at a single station. This is the model considered so far in this paper. In this section, we will study systems where there are precedence relationships among certain tasks. Specifically, in this section, we consider a system of $M \geq 2$ tandem stations with two subtasks and two servers at each station. Denote S_{jk} as the service requirement of subtask j at station k, and assume that $E[S_{jk}] = 1$ for $j = 1, 2, k = 1, \ldots, M$. The service rate of server i working on subtask j at station k is μ_{ijk} for $i, j = 1, 2, k = 1, \ldots, M$. We allow internal buffers after each subtask and intermediate buffers between stations. Let $B_{j,k}$ be the internal buffers of the jth subtask at station k for $j = 1, 2, k = 1, \ldots, M$, and B_k be the intermediate buffers between station k and $k+1$ for $k = 1, \dots, M-1$ (a job will occupy an entire intermediate buffer space as long as processing of at least one of its subtasks has not commenced at the next station).

First, we describe the task assignment approaches in longer lines. As soon as both subtasks of a job at station $k \in \{1, \dots, M\}$ are completed, the job is ready to be assembled (and split again for the service at the next station if $k < M$). However, the job will not enter service at the next station until at least one of the subtasks of its previous job at the next station is completed. Note that, if $k = M$ or $k < M$ and the intermediate buffer between stations k and $k+1$ is not full, then at most one of the internal buffers at station k can have jobs in it (since otherwise, the two completed subtasks of a job will be combined and leave station k immediately). Assume that when $k < M$ and the intermediate buffer is full, the completed two subtasks of a job at station k (if such a job exists) will stay at station k until the intermediate buffer has room for the job.

When there are multiple stations, the three task assignment approaches can no longer be modeled as birth-and-death processes. As a result, it is more difficult to identify the optimal task assignment approaches for longer lines. Hence, we will focus on numerical results in this section.

In Section 3.6.1, we discuss buffer allocation for longer lines, and show that for the static task assignment approach, we can focus on the case when there are no intermediate buffers between stations. Moreover, when there are multiple stations in tandem with infinite intermediate buffers, stations will not be blocked by downstream stations. Then, the long-run average throughput of the system boils down to analyzing stations on their own, and is determined by the bottleneck station with the minimum individual station throughput. Since we have analyzed the best server coordination approaches for one-station systems in the previous sections, the optimal server coordination method can be obtained spontaneously. The difficulty in generalizing the one-station results in Sections 3.1 through 3.5 to longer lines arises due to the blocking of the stations. Therefore, we will focus on the most extreme case with no intermediate or internal buffers, which leads to the highest risk of blocking. In Section 3.6.2, we provide numerical results for the server coordination methods for two tandem stations with no buffers.

3.6.1 Buffer Allocation for Static Task Assignment in Longer Lines

In this section, we investigate the buffer allocation for longer lines with static servers when there exist both internal buffers within stations and intermediate buffers between stations. The following proposition and corollary show that we can focus on the case with no intermediate buffers between stations when the servers are static.

Proposition 3.6.1. *Consider a system with M tandem stations, two servers at each station, and service requirements with general distributions. Assume that the servers are static. If there exists some station $k_0 \in \{1, \dots, M-1\}$ such that $B_{k_0} > 0$, then the maximum long-run average throughput of this system is no more than the maximum throughput of another system with $B_{k_0} - 1$ intermediate buffers after station k_0, $B_{j,k_0} + 1$ internal buffers for subtask $j \in \{1, 2\}$ at station k_0, and the same number of buffers for the other stations.*

The following lemma (part (i) of Lemma 1 of Argon and Andradóttir [15]) will be useful in the proof of Proposition 3.6.1. We present it without proof.

Lemma 3.6.1. *Let a_i, b_i be any real numbers for $i = 1, \dots, n$, where n is a positive integer. Then, $\max_{i=1,\dots,n}\{a_i\} - \max_{i=1,\dots,n}\{b_i\} \geq \min_{i=1,\dots,n}\{a_i - b_i\}$.*

Proof of Proposition 3.6.1. Consider two processes with the same initial system state. Suppose B_k^l is the intermediate buffer after station $k \in \{1, \dots, M-1\}$ in process $l \in \{1, 2\}$, $B_{j,k}^l$ is the internal buffer of the jth subtask at station k in process l for $l, j \in \{1, 2\}, k \in \{1, \dots, M\}$. Then, for $j = 1, 2$, $B_k^2 = B_k^1, B_{j,k}^2 = B_{j,k}^1$, $k \in \{1, \dots, M-1\} \setminus \{k_0\}$, $B_{j,M}^2 = B_{j,M}^1$, $B_{k_0}^1 > 0$, $B_{k_0}^2 = B_{k_0}^1 - 1, B_{j,k_0}^2 = B_{j,k_0}^1 + 1$. That is, process 2 has one less intermediate buffer after station k_0 and one more internal buffer for both subtasks at station k_0 than process 1. Assume that the jobs are labeled according to the order in which they depart from the system. Therefore, if the system is not empty at time zero, then the job with a subtask closest to the end of the line is labeled as job 1. Let $D^l_{j,k}(i)$ be the departure time of job $i \geq 1$ from subtask $j \in \{1, 2\}$ at station $k \in \{1, \dots, M\}$ in process $l \in \{1, 2\}$ (so that job i will be in the internal buffer of subtask j at station k, or in the intermediate buffer between stations k and $k+1$, or at station $k+1$, or out of the system if $k = M$ right after time $D^l_{j,k}(i)$). Let $C^l_k(i)$ be the completion time of both subtasks of job $i \geq 1$ from station $k \in \{1, \dots, M\}$ in process $l \in \{1, 2\}$, i.e.,

$$C^l_k(i) = \max\{D^l_{1,k}(i), D^l_{2,k}(i)\}.$$

Also, let $X^l_{j,k}(i)$ be the service time of job $i > 0$ at subtask $j \in \{1, 2\}$ at station $k \in \{1, \dots, M\}$ in process $l \in \{1, 2\}$. We use the same server assignment at each subtask of each station for job $i \geq 1$ in both processes; hence $X^1_{j,k}(i) = X^2_{j,k}(i)$ for $j \in \{1, 2\}, k \in \{1, \dots, M\}$, and we suppress the superscripts in $X^1_{j,k}(i)$,$X^2_{j,k}(i)$.

First, we give recursive formulas that the departure times $D^l_{j,k}(i)$ must satisfy. Assume that $D^l_{j,k}(i) = X_{j,k}(i) = 0$ if $k \notin \{1, \dots, M\}, j, l \notin \{1, 2\}$, or $i \leq 0$. Then, the departure time of subtask j of a job i at station $k = 1, \dots, M-1$ depends on the following four cases:

1. If there are no jobs in subtask j at station k when job i departs from both subtasks at station $k-1$, and job i is not blocked by the time of its service completion at subtask j at station k, then $D^l_{j,k}(i) = C^l_{k-1}(i) + X_{j,k}(i)$.

2. If job i has waited to be served at subtask j of station k until the service completion of its previous job, and the internal buffer after subtask j is not blocked upon its own service completion at subtask j, then $D^l_{j,k}(i) = D^l_{j,k}(i-1) + X_{j,k}(i)$.

3. If the internal buffer after subtask j is blocked at the time of the service completion, this job will leave subtask j upon a new service completion at subtask $3-j$ of this station. Since in this case the internal buffer after subtask $3-j$ is empty, the

next departure at that subtask should be job $i - B^l_{j,k}$. Thus, in this case, $D^l_{j,k}(i) = D^l_{3-j,k}(i - B^l_{j,k})$.

4. If both the internal buffer after subtask j at station k and the intermediate buffer after station k are full at the time of the service completion, job i will leave subtask j upon a new service completion (i.e., of job $i - B^l_k - B^l_{j,k} - 1$) at station $k+1$. In this case, $D^l_{j,k}(i) = C^l_{k+1}(i - B^l_k - B^l_{j,k} - 1)$.

Note that for $k = M$, $D^l_{j,M}(i)$ only depends on the first three cases since the fourth case does not apply to the last station. Moreover, if job i is being served in subtask j at station k at time zero, then $D^l_{j',k'}(i) = 0$ for $j' = 1, 2, k' < k$, and the first two cases become $D^l_{j,k}(i) = X_{j,k}(i)$, where $X_{j,k}(i)$ stands for the remaining service time of job i at subtask j at station k.

Then for all $k \in \{1, \dots, M\}$, we have that, for $i \geq 1, l, j = 1, 2$,

$$D^l_{j,k}(i) = \max\{C^l_{k-1}(i)+X_{j,k}(i), D^l_{j,k}(i-1)+X_{j,k}(i), D^l_{3-j,k}(i-B^l_{j,k}), C^l_{k+1}(i-B^l_k-B^l_{j,k}-1)\}.$$

As $C^l_m(i) = \max\{D^l_{1,m}(i), D^l_{2,m}(i)\}$ for $m = k-1, k+1$, we have

$$\begin{aligned}D^l_{j,k}(i) = \max\{&D^l_{1,k-1}(i) + X_{j,k}(i), D^l_{2,k-1}(i) + X_{j,k}(i), D^l_{j,k}(i-1) + X_{j,k}(i),\\ &D^l_{3-j,k}(i - B^l_{j,k}), D^l_{1,k+1}(i - B^l_k - B^l_{j,k} - 1), D^l_{2,k+1}(i - B^l_k - B^l_{j,k} - 1)\}.\end{aligned}$$

Note that $B^1_k + B^1_{j,k} = B^2_k + B^2_{j,k}$ for any $k = 1, \dots, M-1, j = 1, 2$, $B^1_{j,M} = B^2_{j,M}$ for $j = 1, 2$.

Let $\Delta_{j,k}(i) = D^1_{j,k}(i) - D^2_{j,k}(i)$. Then by Lemma 3.6.1 and the inequalities above, for $j \in \{1, 2\}, i \geq 1, k = \{1, \dots, M-1\} \setminus \{k_0\}$,

$$\begin{aligned}\Delta_{j,k}(i) \geq \min\{&\Delta_{1,k-1}(i), \Delta_{2,k-1}(i), \Delta_{j,k}(i-1), \Delta_{3-j,k}(i - B^1_{j,k}),\\ &\Delta_{1,k+1}(i - B^1_k - B^1_{j,k} - 1), \Delta_{2,k+1}(i - B^1_k - B^1_{j,k} - 1)\}.\end{aligned}$$

For $k = k_0$, we have

$$\begin{aligned}\Delta_{j,k_0}(i) \geq \min\{&\Delta_{1,k_0-1}(i), \Delta_{2,k_0-1}(i), \Delta_{j,k_0}(i-1), D^1_{3-j,k_0}(i - B^1_{j,k_0}) - D^2_{3-j,k_0}(i - B^1_{j,k_0} - 1),\\ &\Delta_{1,k_0+1}(i - B^1_{k_0} - B^1_{j,k_0} - 1), \Delta_{2,k_0+1}(i - B^1_{k_0} - B^1_{j,k_0} - 1)\}\\ \geq \min\{&\Delta_{1,k_0-1}(i), \Delta_{2,k_0-1}(i), \Delta_{j,k_0}(i-1), \Delta_{3-j,k_0}(i - B^1_{j,k_0} - 1),\\ &\Delta_{1,k_0+1}(i - B^l_{k_0} - B^1_{j,k_0} - 1), \Delta_{2,k_0+1}(i - B^1_{k_0} - B^1_{j,k_0} - 1)\}.\end{aligned}$$

Finally, for $k = M$,

$$\Delta_{j,M}(i) \geq \min\{\Delta_{1,M-1}(i), \Delta_{2,M-1}(i), \Delta_{j,M}(i-1), \Delta_{3-j,M}(i - B^1_{j,M})\}.$$

Note that $\Delta_{j,k}(i) = 0$ when $j \notin \{1, 2\}, k \notin \{1, \ldots, M\}$, or $i \leq 0$. It is easy to see $\Delta_{j,k}(i) \geq 0$ for all $j \in \{1, 2\}, k = 1, \ldots, M$ and $i \geq 1$ by induction. Thus, $D^1_{j,M}(i) \geq D^2_{j,M}(i)$ for $j = 1, 2$, and the departure time of job i from station M in process 1 is later than in process 2, for $\forall i \geq 1$. It follows that process 2 has no smaller long-run average throughput than process 1. $\square$

Note that, we can generalize Proposition 3.6.1 to arbitrary number of subtasks at each station. For instance, if there are J subtasks at some station k, then the departure times $D^l_{j,k}(i)$ still depend on the four cases we discussed earlier, but the completion time becomes $C^l_k(i) = \max\{D^l_{1,k}(i), \ldots, D^l_{J,k}(i)\}$, and case 3 will be revised as $D^l_{j,k}(i) = \max_{j' \in \{1,\ldots,J\} \setminus \{j\}} \{D^l_{j',k}(i - B^l_{j,k})\}$.

The following corollary follows from Proposition 3.6.1.

Corollary 3.6.1. *For the static task assignment approach, if* $B_k > 0$*, where* $k \in \{1, \ldots, M-1\}$*, then the maximum long-run average throughput of this system is no more than the maximum throughput of another system with no intermediate buffer between stations* k *and* $k+1$*,* $B_{j,k} + B_k$ *internal buffers for subtask* $j \in \{1, 2\}$ *at station* k*, and the same number of buffers for the other stations.*

Note that Proposition 3.6.1 and Corollary 3.6.1 do not hold if the servers are flexible or collaborative. The following example shows that intermediate buffers can be preferable to internal buffers when the servers are flexible or collaborative.

Example 3.6.1. *Consider two systems, each with two stations in tandem and two servers at each station. System A has one internal buffer for each subtask at station 1, and no buffers anywhere else; system B has one intermediate buffer between stations 1 and 2, and no buffers anywhere else. Suppose that the service requirement S_{jk} is exponentially distributed for $\forall j, k \in \{1, 2\}$.*

(i) When the servers are flexible and non-collaborative, if the service rates are $\mu_{111} = \mu_{121} = 2, \mu_{211} = \mu_{221} = \mu_{112} = \mu_{212} = \mu_{122} = \mu_{222} = 1,$ then the optimal flexible task assignment of system A yields a throughput of 0.6425, and the optimal flexible task assignment of system B yields a throughput of 0.6437.

(ii) When the servers are flexible and collaborative, if the service rates are $\mu_{ijk} = 1$ for $i, j, k \in \{1, 2\}$, and the synergy level is $\alpha = 1$, the optimal collaborative task assignment of system A yields a throughput of 0.8023, and the optimal collaborative task assignment of system B yields a throughput of 0.8214.

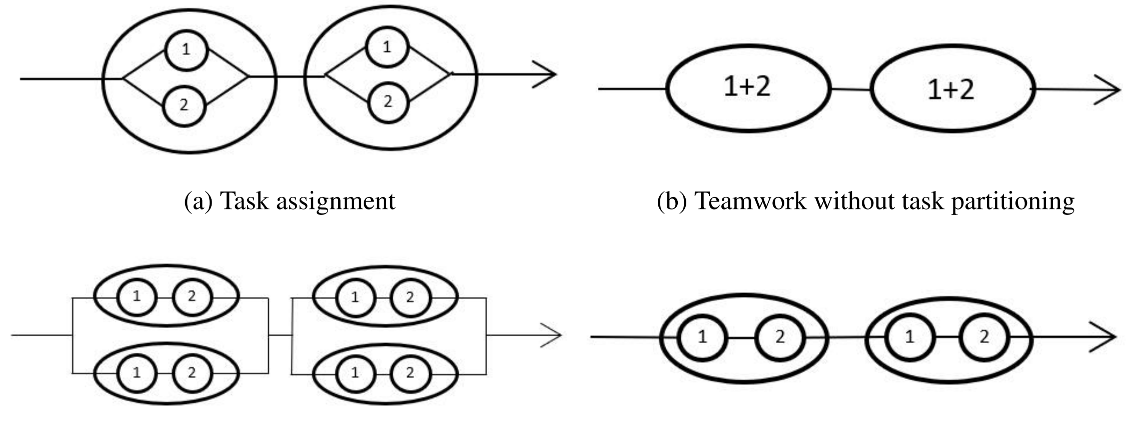
In both cases, system B yields a higher throughput than system A.

Intuitively, allocating the buffers within a station as internal buffers rather than outside of the station as intermediate buffers reduces blocking. When the servers are static, server assignments are identical for both the primary and secondary assignments, and less blocking yields higher efficiency of the system. However, when the servers are flexible and one server dominates the other server at a station (like in station 1 of Example 3.6.1(i)), or when the servers are flexible and collaborative (like in Example 3.6.1(ii)), we get greater benefit from the high service rates of the secondary assignment when there is blocking at that station. Therefore, when the servers are flexible and not equally well trained, or collaborative with synergy level that is not too low, we may prefer intermediate buffers to internal buffers.

3.6.2 Numerical Results for Two Tandem Stations

In this section, we will provide numerical results for the system with two tandem stations, two subtasks and servers at each station, and no internal or intermediate buffers between the stations ($B_1 = B_{j,k} = 0$ for $j, k \in \{1, 2\}$). Assume that the service requirement S_{jk} is exponentially distributed for $\forall j, k \in \{1, 2\}$. For teamwork without task partitioning, the two servers work together on a combined job with a combined service rate at each station; while for teamwork with task partitioning, the two servers work together on the two subtasks of a job in tandem at each station (see Appendix A.5 for details on the random service times under teamwork with or without task partitioning). For non-collaboration, since the two servers work in parallel at each station, we need to determine the priority scheme for the arrivals from the previous station. For simplicity, we assume that when both servers at station 1 are blocked and a server becomes available at station 2, we will serve the job from server 1 at station 1 first; when both servers are starved at station 2 and a job is completed at station 1, the incoming job will go to server 1 at station 2. Note that unlike static task assignment, there is no advantage in having internal buffers relative to intermediate buffers for the teamwork with or without task partitioning and non-collaboration approaches. Moreover, non-collaboration is no longer a special case of teamwork without task partitioning when $\alpha = 1$. See Figure 3.5 for the flow plot of the server coordination methods with two stations and two servers at each station.

Since the systems under consideration can all be modeled as continuous-time Markov chains, we compute the long-run average throughput of all the server coordination methods we discussed earlier for this two-station case by solving the balance equations. Similar to Sections 3.5.3 and 3.5.4, we consider two types of servers, namely generalists and specialists. We consider the same sets of service rates at each station as in Sections 3.5.3 and



(a) Task assignment (b) Teamwork without task partitioning

(c) Non-collaboration (d) Teamwork with task partitioning

Figure 3.5: Server coordination approaches for two stations with no buffers

3.5.4, and the numerical results are shown in Tables 3.4 and 5.11, respectively.

Table 3.4: Throughputs of server coordination methods for two stations and generalists (the highest throughputs in each row are in bold).

α	μ_{111}	μ_{211}	μ_{121}	μ_{221}	μ_{112}	μ_{212}	μ_{122}	μ_{222}	S	F	C	TP	T	NC
1.5	1	1	1	1	1	1	1	1	0.480	0.480	0.870	**1.091**	**1.091**	0.789
1.5	2	2	1	1	2	2	1	1	0.606	0.606	1.181	1.373	**1.455**	1.046
1.5	2	1	2	1	2	1	2	1	0.606	0.870	1.304	**1.636**	**1.636**	1.185
1.5	1	2	2	4	1	2	2	4	0.960	1.145	1.809	2.059	**2.182**	1.543
1.2	1	1	1	1	1	1	1	1	0.480	0.480	0.793	**0.873**	**0.873**	0.789
1.2	2	2	1	1	2	2	1	1	0.606	0.606	1.060	1.098	**1.164**	1.046
1.2	2	1	2	1	2	1	2	1	0.606	0.870	1.189	**1.309**	**1.309**	1.185
1.2	1	2	2	4	1	2	2	4	0.960	1.145	1.605	1.648	**1.745**	1.543
1.0	1	1	1	1	1	1	1	1	0.480	0.480	0.727	0.727	0.727	**0.789**
1.0	2	2	1	1	2	2	1	1	0.606	0.606	0.960	0.915	0.970	**1.046**
1.0	2	1	2	1	2	1	2	1	0.606	0.870	1.091	1.091	1.091	**1.185**
1.0	1	2	2	4	1	2	2	4	0.960	1.145	1.440	1.373	1.455	**1.543**
0.8	1	1	1	1	1	1	1	1	0.480	0.480	0.646	0.582	0.582	**0.789**
0.8	2	2	1	1	2	2	1	1	0.606	0.606	0.839	0.732	0.776	**1.046**
0.8	2	1	2	1	2	1	2	1	0.606	0.870	0.969	0.873	0.873	**1.185**
0.8	1	2	2	4	1	2	2	4	0.960	1.145	1.277	1.098	1.164	**1.543**
0.5	1	1	1	1	1	1	1	1	0.480	0.480	0.480	0.364	0.364	**0.789**
0.5	2	2	1	1	2	2	1	1	0.606	0.606	0.606	0.458	0.485	**1.046**
0.5	2	1	2	1	2	1	2	1	0.606	0.870	0.870	0.545	0.545	**1.185**
0.5	1	2	2	4	1	2	2	4	0.960	1.145	1.145	0.686	0.727	**1.543**

Comparing the results for one station when $B_1 = B_2 = 0$ (in Tables 5.5, 5.6) and for two stations (in Tables 3.4, 5.11), we can see that:

1. When the servers are generalists,

Table 3.5: Throughputs of server coordination methods for two stations and specialists (the highest throughputs in each row are in bold).

α	μ_{111}	μ_{211}	μ_{121}	μ_{221}	μ_{112}	μ_{212}	μ_{122}	μ_{222}	S	F	C	TP	T	NC
1.5	1	1	1	1	1	1	1	1	0.480	0.480	0.870	**1.091**	**1.091**	0.789
1.5	2	1	1	2	2	1	1	2	0.960	0.960	1.540	**1.636**	1.455	1.046
1.5	3	1	1	3	3	1	1	3	1.440	1.440	**2.182**	**2.182**	1.636	1.168
1.5	4	1	1	4	4	1	1	4	1.920	1.920	**2.815**	2.727	1.745	1.239
1.2	1	1	1	1	1	1	1	1	0.480	0.480	0.793	**0.873**	**0.873**	0.789
1.2	2	1	1	2	2	1	1	2	0.960	0.960	**1.378**	1.309	1.164	1.046
1.2	3	1	1	3	3	1	1	3	1.440	1.440	**1.937**	1.746	1.309	1.168
1.2	4	1	1	4	4	1	1	4	1.920	1.920	**2.489**	2.182	1.396	1.239
1.0	1	1	1	1	1	1	1	1	0.480	0.480	0.727	0.727	0.727	**0.789**
1.0	2	1	1	2	2	1	1	2	0.960	0.960	**1.244**	1.091	0.970	1.046
1.0	3	1	1	3	3	1	1	3	1.440	1.440	**1.739**	1.455	1.091	1.168
1.0	4	1	1	4	4	1	1	4	1.920	1.920	**2.227**	1.818	1.164	1.239
0.8	1	1	1	1	1	1	1	1	0.480	0.480	0.646	0.582	0.582	**0.789**
0.8	2	1	1	2	2	1	1	2	0.960	0.960	**1.085**	0.873	0.776	1.046
0.8	3	1	1	3	3	1	1	3	1.440	1.440	**1.505**	1.164	0.873	1.168
0.8	4	1	1	4	4	1	1	4	**1.920**	**1.920**	**1.920**	1.455	0.931	1.239
0.5	1	1	1	1	1	1	1	1	0.480	0.480	0.480	0.364	0.364	**0.789**
0.5	2	1	1	2	2	1	1	2	0.960	0.960	0.960	0.546	0.485	**1.046**
0.5	3	1	1	3	3	1	1	3	**1.440**	**1.440**	**1.440**	0.727	0.545	1.168
0.5	4	1	1	4	4	1	1	4	**1.920**	**1.920**	**1.920**	0.909	0.582	1.239

(a) Teamwork without task partitioning is no longer always equivalent to teamwork with task partitioning. In fact, it is strictly better than teamwork with task partitioning when the subtasks are of different difficulties (i.e., cases (ii) and (iv)). Intuitively, there is no blocking for the one-station system since there is infinite space at the end of the line. However, for systems with multiple stations and finite (zero in this example) intermediate buffers between stations, blocking becomes a serious issue for the throughput. Different task difficulties yield unbalanced service rates at the subtasks, and increase the possibility of blocking for teamwork with task partitioning. Meanwhile, teamwork without task partitioning neutralizes this effect by combining the tasks together. Moreover, when the servers are generalists, teamwork with task partitioning loses its advantage of server specialty. In fact, the variance of teamwork without task partitioning is lower than that of teamwork with task partitioning in this case, as we show in Appendix A.5. Thus, unlike our results for one-station systems, teamwork without task partitioning is now no worse than teamwork with task partitioning with generalist servers for the two-station systems we consider.

   (b) When server synergy is high (i.e., $\alpha \geq 1.2$), teamwork without task partitioning is the best method; when server synergy level is moderate and low (i.e., $\alpha \leq 1$) non-collaboration is the best. These results are consistent with our one-station results except that when server collaboration is additive (i.e., $\alpha = 1$), non-collaboration is strictly better than teamwork for two-station systems while these two methods perform the same in one-station systems.

   (c) When $\alpha = 1$, for one-station system, collaborative task assignment, teamwork, and non-collaboration are equivalent; but for two-station systems, non-collaboration is strictly better than the other two methods since the probability of blocking is highly increased with multiple stations and zero internal and intermediate buffers while non-collaboration has more WIP and thus less chance of blocking.

   (d) When the servers are identical (i.e., cases (i), (ii)), static task assignment is still equivalent to flexible task assignment. And when servers are not identical, server flexibility again increases the throughput significantly (from 20% to 40%, as in one-station systems).

2. When the servers are specialists,

   (a) Teamwork with task partitioning is the best when the synergy level is high and the specialty level is not high; non-collaboration is the best when both the specialty level and the synergy level are small; otherwise, the server specialization is not small and collaborative task assignment is best. These results are consistent with our results for the one-station system when the buffers are zero.

   (b) Flexible task assignment is still equivalent to static task assignment.

3. Collaborative task assignment is better than static and flexible task assignment as long as the synergy level is not too small (i.e., $\alpha \geq 0.8$).
4. The throughputs of all server coordination methods for two-station systems are lower than for the corresponding one-station systems since more stations with no buffers increases the probability of blocking and reduces the throughputs.
5. For two-station systems, non-collaboration is now strictly better than teamwork without task partitioning when $\alpha = 1$. Intuitively, non-collaboration has more WIP than teamwork. This additional WIP does not improve throughput for one-station systems since there is no blocking; however, for systems with multiple stations and zero intermediate buffers, this additional WIP is crucial to the long-run average throughput. Thus, the performance of non-collaboration for two-station systems is better than for one-station systems relative to other server coordination methods.

In conclusion, the numerical results for two stations suggest that our comparison of different server coordination methods for one station case provided in Sections 3.5.3 and 3.5.4 generalize to longer lines in most cases. By combining the results for one- and two-station systems, we have the following conclusions:

1. When the servers are generalists,
   (a) if $\alpha \leq 1$, we prefer non-collaboration;
   (b) if $\alpha > 1$, we prefer teamwork without task partitioning.
2. When the servers are specialists,
   (a) if the specialty levels of the servers are high, we prefer collaborative task assignment;
   (b) if the specialty levels of the servers are moderate or low, and
      i. the synergy level is high, we prefer teamwork with task partitioning;
      ii. the synergy level is moderate or low, we prefer non-collaboration.

3.7 Conclusions

For a queueing system with servers that are either static, flexible, or collaborative, we considered different server coordination methods when each job can be decomposed into multiple subtasks and there are no precedence relationships among the subtasks within each station. The objective is to maximize the long-run average throughput of the system. We first characterized the optimal static, flexible, and collaborative task assignment approaches, and further analyzed the optimal policies for two special cases, namely when buffers are zero and when the sum of the buffers goes to infinity. Then, we investigated three other server coordination methods, namely teamwork with or without task partitioning and non-collaboration, compared them to task assignment approaches, and determined when and how to choose a server coordination methods under different circumstances. Moreover, we further investigated these methods when the servers are generalists or specialists, and provided corresponding numerical results. Then, we analyzed server coordination for longer lines. We proved that for static task assignment, it is always better to allocate the available buffers within stations as internal buffers rather than after stations as intermediate buffers (however, this result does not hold for flexible or collaborative task assignment). Finally, we provided numerical results for the two-station case that suggested our one-station results can be generalized to longer lines.

Based on our theoretical and numerical analyses, we obtained the following additional insights:

1. Teamwork with task partitioning is no worse than teamwork without task partitioning when the servers are specialists, and it is no better than teamwork without task partitioning when the servers are generalists.

2. For one-station systems, non-collaboration has the same throughput as teamwork without task partitioning when the synergy level is one. However, this property no longer holds for longer lines with finite intermediate buffers.

3. When the servers are generalists, we prefer non-collaboration, and then teamwork without task partitioning as the synergy level among servers goes from low to high.

4. When the servers are specialists, we prefer collaborative task assignment if the servers are highly specialized, otherwise, we prefer teamwork with task partitioning if the synergy level is high, non-collaboration if the synergy level is moderate or low.

CHAPTER 4
DYNAMIC CONTROL OF SERVICE SYSTEMS WITH TEAMS

Consider a tandem queueing network with $N \geq 1$ stations and $M \geq 1$ servers. There are infinitely many jobs awaiting in front of the first station, and each job will be processed by stations $1, 2, \ldots, N$ before leaving the system. There is a finite buffer of size $0 \leq B_j < \infty$ between stations j and $j+1$, for $j = 1, 2, \ldots, N-1$, and infinitely large storage space after station N. Let T be the set of all possible team assignments and γ_{ij} be the service rate of team i at station j, where $i \in T$,$j \in \{1, 2, \ldots, N\}$. Moreover, we assume that $|T| < \infty$, $\gamma_{ij} \geq 0$ for all $i \in T, j \in \{1, 2, \ldots, N\}$, and restrict our attention to teams such that $\sum_{j=1}^{N} \gamma_{ij} > 0, \forall i \in T$ (otherwise, this team assignment is trivial and should be eliminated from T). Furthermore, without loss of generality, assume that $\sum_{i \in T} \gamma_{ij} > 0$ for any $j \in \{1, 2, \ldots, N\}$ (otherwise, the throughput of the system is zero).

Our objective is to determine the dynamic server assignment policy that maximizes the long-run average throughput of this queueing system. We first establish sufficient criteria for eliminating inferior teams, and then we identify the optimal policy among the remaining teams for two stations case. Next, we apply our optimal policies to two special cases. In the first case, the team service rate is proportional to the sum of the service rates of team members with factor $\alpha > 0$. In the second case, we assume that there are K different types of servers with different specialties and the team formation is constrained in that each team must consist of exactly one server of each type. For example, during a surgery, a team of medical staff helps the surgeon during the operation which may include an anesthesiologist, an operating room nurse, etc. We provide the optimal team assignment for systems with two stations for both cases. We put more effort on the second case. Finally, we explore heuristics for longer systems with constrained team formations when the servers are generalists.

The outline of this chapter is as follows. In Section 4.1, we formulate the team-assignment problem considered in this chapter, and provide a preliminary criterion to eliminate teams that are not on the Pareto boundary. In Section 4.2, we provide a secondary criterion to further eliminate inferior teams, and also provide the team assignment policy that maximizes the long-run average throughput of the systems with two stations. Moreover, we briefly discuss the optimal policy for this model when the servers are static. In Section 4.3, we first present a numerical example that illustrates the optimal policy obtained in Section 4.2, and then investigate the special case with proportional team service rates, and finally, we explore the optimal policy for systems with constrained team formations. In Section 4.4.1, we study heuristic policies that appear to yield near-optimal performance with teams of specialized servers when the servers are generalists for systems with more than two tandem stations. Section 4.5 concludes the chapter.

4.1 Problem Formulation and Team Selection

In this section, we first present a detailed description of our model, and then provide a primary criterion to select the teams we consider in the optimal policy. Let Π be the set of server assignment policies under consideration. Under policy $\pi \in \Pi$, the network state at time $t \in [0, \infty)$ is $X^{\pi}(t)$, where the jth component of $X^{\pi}(t)$ is the number of jobs in the system that have completed service at station j but have not yet completed service at station $j+1$ for $j \in 1, 2, \ldots, N-1$. Then $\{X^{\pi}(t) : t \geq 0\}$ is a continuous time Markov chain. The state space of $\{X^{\pi}(t) : t \geq 0\}$ is $S \subseteq \{(s_1, s_2, \ldots, s_{N-1}) : s_j \in \{0, 1, \ldots, B_j + 2\}, \forall j \in 1, 2, \ldots, N-1\}$. The action sets are given by the possible server allocations. Thus, for $s \in S$, the action set $A_s \subseteq T$. Let $\pi(s)$ be a projection from the state space to the action set (i.e., team assignment), $\pi : S \to T$. For all $\pi \in \Pi$, let $D_s^{\pi}(t)$ be the number of departures from the last station under policy π by time t with initial state $s \in S$, and let

$$g_s^{\pi} = \limsup_{t \to \infty} \frac{\mathbb{E}[D_s^{\pi}(t)]}{t} \tag{4.1}$$

be the long-run average throughput corresponding to server allocation policy π starting from state $s \in S$. Our objective is to solve the following optimization problem.

$$\max_{\pi \in \Pi} g_s^\pi, \forall s \in S. \tag{4.2}$$

Let $\{q^\pi(x, x')\}$ be the transition rates of $\{X^\pi(t)\}$, then there exists a finite uniformization constant $q \leq \sum_i \sum_j \gamma_{i,j} < \infty$ such that $\{q^\pi(x, x')\}$ satisfy $\sum_{x' \in S, x' \neq x} q^\pi(x, x') \leq q$ for all $x \in S, \pi \in \Pi$. Thus, $\{X^\pi(t)\}$ is uniformizable. Let $\{Y^\pi(k)\}$ be the corresponding discrete-time Markov chain, so that $\{Y^\pi(k)\}$ has state space S and transition probabilities $p^\pi(x, x') = q^\pi(x, x')/q$ if $x' \neq x$ and $p^\pi(x, x) = 1 - \sum_{x' \in S, x' \neq x} q^\pi(x, x')/q$ for all $x \in S$. Using a similar argument as in Section 3 of Andradóttir, Ayhan and Down [7], we can show that the original optimization problem in (5.2) can be translated into an equivalent discrete-time Markov decision problem. Thus, maximizing the long-run average throughput of the original queueing system is equivalent to maximizing the long-run average departure rate for the associated embedded discrete-time Markov chain.

A policy $\pi^* \in \Pi$ is called *optimal* if

$$g_s^{\pi^*} \geq g_s^\pi \text{ for all } \pi \in \Pi \text{ and } s \in S.$$

Note that since $\sum_{i \in T} \gamma_{ij} > 0$ for any $j \in \{1, 2, \ldots, N\}$, the Markov decision process is communicating and there exists a π^* such that $g_s^{\pi^*}$ is the same for all $s \in S$. Let g^* denote this common value. Since $T < \infty$, the state space and the buffers are finite, by Theorem 9.1.8 in [41], there exists a deterministic stationary optimal policy. Therefore, from now on, we assume that the class Π of server assignment policies under consideration consists of all Markovian stationary deterministic policies corresponding to the state space S of the stochastic processes $\{X^\pi(t)\}$. Let Π^* be the set of all optimal stationary deterministic policies.

Note that, if the servers are flexible and are trained to work at all the stations, they could be assigned to any one of the N stations. That is, $|T|$ could be up to N^M, which will get extremely large as N and M grow. To reduce the number of team assignments under consideration, we introduce the following concept and theorem.

Definition 4.1.1. *(Revised Pareto Boundary) Team assignment $i \in T$ is on the revised Pareto boundary of T if there is no team assignment $k \neq i, k \in T$, such that $\gamma_{k,1} > \gamma_{i,1}, \ldots, \gamma_{k,N} > \gamma_{i,N}$.*

Note that, the set of team assignments on the revised Pareto boundary is nonempty. Indeed, if $i' = \arg\max_{i \in T} \gamma_{ij} > 0$ for some station j, and i' is unique, then i' is on the revised Pareto boundary by definition; and if such i' is not unique for j, at least one of such i' should be on the revised Pareto boundary by definition.

Theorem 4.1.1. *Any policy π that uses a team assignment that is not on the revised Pareto boundary in any recurrent state under that policy is not optimal.*

Proof. Suppose team assignment $i \in T$ is not on the revised Pareto boundary. Then $\exists k \in T$, and $\alpha_1, \ldots, \alpha_N > 1$, s.t. $\gamma_{k,1} = \alpha_1 \gamma_{i,1}, \ldots, \gamma_{k,N} = \alpha_N \gamma_{i,N}$. Without loss of generality, assume that $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_N > 1$.

Suppose we use team assignment i in some recurrent state s_0 under policy $\pi \in \Pi$.

Step 1: We first prove the result when $\alpha_1 = \alpha_2 = \ldots = \alpha_N > 1$. Consider policy π' such that $\pi'(s) = \pi(s)$ for $\forall s \neq s_0$, and $\pi'(s_0) = k$. Then if we replace policy π by π', we increase service rates at all stations by the same proportion α_1. So, we can reduce the time spent in state s_0 without making any changes to all the other states. Hence, throughput of policy π' is higher than π and policy π cannot be optimal.

Step 2: Next we show the result when $\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_N > 1$ and at least one of the inequalities is strict. For any team assignment $t \in T$, let t_{-j} be the team assignment that assigns the servers to stations according to team assignment t, but let the servers at stations $1, \ldots, j$ be idle, where $j \in \{1, \ldots, N-1\}$. Then, $t_{-j} \in T$, for $j \in \{1, \ldots, N-1\}$.

Consider policy π'' such that $\pi''(s) = \pi(s)$ for $\forall s \neq s_0$, and $\pi''(s_0) = k$ with probability $p_1 = \frac{\alpha_N}{\alpha_1}$; $\pi''(s_0) = k_{-(j-1)}$ with probability $p_j = \alpha_N(\frac{1}{\alpha_j} - \frac{1}{\alpha_{j-1}})$ for $j = 2, \ldots, N$. That is, assign the server according to team assignment k for policy π'', but let the servers at the jth station idle with probability $\sum_{r=j+1}^{N} p_r = \sum_{r=j+1}^{N} \alpha_N(\frac{1}{\alpha_r} - \frac{1}{\alpha_{r-1}}) = \alpha_N(\frac{1}{\alpha_{j+1}} - \frac{1}{\alpha_j} + \cdots + \frac{1}{\alpha_N} - \frac{1}{\alpha_{N-1}}) = \alpha_N(\frac{1}{\alpha_N} - \frac{1}{\alpha_j}) = 1 - \frac{\alpha_N}{\alpha_j}$. Then in state s_0, the expected service rate of station 1 is $\gamma_{k,1} * p_1 = \gamma_{i,1}\alpha_1\frac{\alpha_N}{\alpha_1} = \alpha_N\gamma_{i,1}$; and the expected service rate of station $j = 2, \ldots, N$ is $\gamma_{k,j}(1 - \sum_{r=j+1}^{N} p_r) = \alpha_j\gamma_{i,j}\frac{\alpha_N}{\alpha_j} = \gamma_{i,j}\alpha_N$. Thus, if we replace policy π by π'', the average service rates of all stations in state s_0 increased by the same proportion and the service rates remain unchanged in other states. Therefore, π'' yields a higher throughput than π, and π cannot be optimal. $\square$

Note that, the Pareto boundary by convention is slightly different from our revised Pareto boundary, and we provide the definition of teams on Pareto boundary as follows.

Definition 4.1.2. *(Pareto Boundary) Team assignment $i \in T$ is on the Pareto Boundary of T if there is no team assignment $k \neq i, k \in T$, such that $\gamma_{k,1} \geq \gamma_{i,1}, \ldots, \gamma_{k,N} \geq \gamma_{i,N}$, and at least one of these inequalities is strict.*

Remark 4.1.1. *Note that, the Pareto boundary set defined in Definition 4.1.2 is a smaller set than the revised Pareto boundary set defined in Definition 4.1.1. Indeed, if $i, k \in T$ such that $\gamma_{k,1} = \gamma_{i,1}, \ldots, \gamma_{k,N-1} = \gamma_{i,N-1}, \gamma_{k,N} > \gamma_{i,N}$, then it is possible for both i, k to be on the revised Pareto boundary, but i is definitely not on the Pareto boundary.*

We refer to a team assignment as *replaceable* if we can find a stationary deterministic policy that does not use this team assignment but can still achieve at least the same throughput of the policies using this team assignment in any recurrent state. By Theorem 4.1.1, any team assignment that is not on the revised Pareto boundary is replaceable. Moreover, the following proposition shows that any team assignment that is on the revised Pareto Boundary but not on the Pareto Boundary is also replaceable.

Proposition 4.1.1. *Any team assignment that is not on the Pareto Boundary is replaceable.*

Proof. We only need to check the team assignments that are on the revised Pareto Boundary but not on the Pareto Boundary. If team assignment i is on the revised Pareto Boundary but not on the Pareto Boundary, then there exists a team k on the Pareto Boundary such that $\gamma_{k,n} = \gamma_{i,n}$ for $n \in Q$, where $Q \subset \{1, 2, \ldots, N\}$ and $\gamma_{k,n} > \gamma_{i,n}$ for $n \in \{1, 2, \ldots, N\} \setminus Q$. We consider the case when $|Q| = N - 1$ via a sample path argument, and the other cases can be solved by induction.

Assume that $\gamma_{k,j} > \gamma_{i,j}$, for some $j \in \{1, 2, \ldots, N\}$, and $\gamma_{k,n} = \gamma_{i,n}$ for $n \in \{1, 2, \ldots, N\} \setminus \{j\}$. Consider two processes with the same initial system. We use common random numbers to generate the service times at each station for both processes at the beginning of each time epoch (that is, the service time is proportional to the service rate with a common factor for both processes at each station). Recall that $X^{\pi}(t)$ indicates the state of the system at time t under policy π. Let $D^{\pi}(t)$ be the number of departures from the last station under policy π by time t. Suppose that Process 1 uses a policy $\pi \in \Pi$ that uses team assignment i in some state $\tilde{s} \in S$. Let τ_0 be the first time that Process 1 enters state $\tilde{s}$, and τ_1 be the first time after time τ_0 that Process 1 departs from state $\tilde{s}$. Suppose that Process 2 uses a policy $\tilde{\pi}$ such that it uses the same team assignment as π until time τ_0, and uses team assignment k right after τ_0. The next event among these two processes is either a service completion at station $n \neq j$ (in both processes) at τ_1 (since the same service rates yield the same service times for any station $n \neq j$), or a service completion at station j in Process 2, denote the time of this event as τ_2. Note that, $< \tau_1$ since process 2 has a higher service rate at station j than process 2. For the first case, the two processes are still in the same state by time τ_1. Suppose Process 2 uses the same team assignment as Process 1 thereafter, then there is no difference in the reward for both processes, and $D^{\tilde{\pi}}(t) = D^{\pi}(t)$ for $t \geq 0$. For the second case, we have

$$X^{\tilde{\pi}}(\tau_2^+) = X^{\pi}(\tau_2^+) + e_j 1\{j < N\} - e_{j-1} 1\{j > 1\}.$$

Note that, when $j = N$, $D^{\tilde{\pi}}(\tau_2^+) = D^{\pi}(\tau_2^+) + 1$; when $j \in \{1, \ldots, N_1\}$, $D^{\tilde{\pi}}(\tau_2^+) = D^{\pi}(\tau_2^+)$. Thus, $D^{\tilde{\pi}}(t) \geq D^{\pi}(t)$ for $t \in [0, \tau_2]$. From τ_2 onwards, $\tilde{\pi}$ uses the same team assignment as π until one of the following events occurs:

1. For $j < N$,

    (i) Whenever station j is starved in Process 1, let $\tilde{\pi}$ idle the server at station j in Process 2 until the occurrence of the next event, and let $\tilde{\pi}$ use the same team assignment as π until either (i) or (ii) happens.

    (ii) If station j is blocked in Process 2 but still working in Process 1 and the next event is a service completion at station j in Process 1, then the two processes couple by the beginning of the next event, and let $\tilde{\pi}$ use the same team assignment as π thereafter, we have $D^{\tilde{\pi}}(t) = D^{\pi}(t)$ for $t \geq 0$.

2. For $j = N$,

(i) Whenever station $j - 1$ is blocked in Process 1 but not blocked in Process 2, let $\tilde{\pi}$ idle the server at station $j - 1$ in Process 2 until the occurrence of the next event, and let $\tilde{\pi}$ uses the same team assignment as π until either (i) or (ii) happens.

(ii) If station j is starved in Process 2 but still working in Process 1 and the next event is a service completion at station j in Process 1, then the two processes couples by the beginning of the next event, and let $\tilde{\pi}$ uses the same team assignment as π thereafter, we have $D^{\tilde{\pi}}(t) \geq D^{\pi}(t)$ for $t \geq 0$.

Since in each case, $D^{\tilde{\pi}}(t) \geq D^{\pi}(t)$ for $t \geq 0$, the long-run average throughput under $\tilde{\pi}$ is no less than under π. Repeating this process, we can replace team assignment i by k and obtain a policy $\tilde{\pi}_* \in \Pi$ that never uses i but yields a throughput no less than policy that uses i. Thus, team assignment i is replaceable. □

Theorem 4.1.1 and Proposition 4.1.1 imply that we can only consider the team assignments on the Pareto boundary when seeking for an optimal policy. The following section characterizes the optimal policy for a two station system.

4.2 Optimal Policy for Two Stations

In this section, we provide the optimal policy for the two-station case, i.e., $N = 2$. For simplicity, let $B_1 = B$. Then the corresponding state space is $S = \{0, 1, \dots, B + 2\}$. And the system can be regarded as a birth-death process. We've already restricted our choice to team assignments on the Pareto boundary, but not all of them will be used in the optimal policy. In Section 4.2.1, we further remove the team assignments that are dominated or replaceable by other team assignments and obtain an optimal assignment set. In Section 4.2.2, we show how to find an optimal policy within this optimal assignment set.

4.2.1 Removing the Dominated and Replaceable Team Assignments

We first introduce the definition of the dominated team assignments that we will never use in any of the optimal policies as follows.

Definition 4.2.1. *(Dominated Team Assignment) Team assignment $i \in T$ is a dominated team assignment, if $\gamma_{i,j} > 0$ for $j = 1, 2$ and there exist two other assignments k, l such that*

(i) $\gamma_{k,1} \geq \gamma_{i,1} \geq \gamma_{l,1}$,

(ii) $\gamma_{k,2} \leq \gamma_{i,2} \leq \gamma_{l,2}$,

(iii) $(\gamma_{i,1} - \gamma_{l,1})(\gamma_{i,2} - \gamma_{k,2}) < (\gamma_{k,1} - \gamma_{i,1})(\gamma_{l,2} - \gamma_{i,2})$.

Note that constraints (i) and (ii) indicate that the service rates of team assignment i is in between team assignments k, l at both stations, the left hand side of constraint (iii) can be interpreted as the advantage of using i instead of k, l at both stations, and the right hand side of constraint (iii) is the advantage of using k, l instead of i at both stations. Intuitively, we define i be dominated by k and l if the gain of using team i is less than the loss of not using k and l, since as shown in the following theorem, we can achieve a higher long-run average throughput by using k, l instead of i.

Theorem 4.2.1. *Any policy π that forms an irreducible Markov chain and uses a dominated team assignment in any state under policy π is not optimal.*

Proof. Let $\{Z(t) : t > 0\}$ be a birth-death process with state space $\{0, \ldots, B+2\}$. Let $\mu_{s,1}$ and $\mu_{s,2}$ denote the birth and death rates in state $s \in S$. Suppose that $\{Z(t) : t > 0\}$ forms an irreducible Markov chain. Then the long-run average throughput is:

$$g = \frac{\Theta_1}{\Theta_2} := \frac{\mu_{01} + \mu_{01}\frac{\mu_{11}}{\mu_{12}} + \ldots + \mu_{01}\frac{\mu_{11}\cdots\mu_{B+1,1}}{\mu_{12}\cdots\mu_{B+1,2}}}{1 + \frac{\mu_{01}}{\mu_{12}} + \ldots + \frac{\mu_{01}\cdots\mu_{B+1,1}}{\mu_{12}\cdots\mu_{B+2,2}}}.$$

If for some state $s_0 \in \{1, \ldots, B+1\}$, $\mu_{s_0,1} \to \mu_{s_0,1} + \Delta_1 \geq 0, \mu_{s_0,2} \to \mu_{s_0,2} - \Delta_2 \geq 0$, then the corresponding long-run average throughput is:

$$g' = \frac{(\mu_{s_0,2} - \Delta_2)C_1 + (\mu_{s_0,1} + \Delta_1)C_2}{(\mu_{s_0,2} - \Delta_2)C_3 + C_4 + (\mu_{s_0,1} + \Delta_1)C_5} = \frac{\Theta_1 - \Delta_2 C_1 + \Delta_1 C_2}{\Theta_2 - \Delta_2 C_3 + \Delta_1 C_5},$$

where

$$C_1 = \mu_{01} + \mu_{01}\frac{\mu_{11}}{\mu_{12}} + \ldots + \mu_{01}\frac{\mu_{11}\cdots\mu_{s_0-1,1}}{\mu_{12}\cdots\mu_{s_0-1,2}},$$

$$C_2 = \mu_{01}\frac{\mu_{11}\cdots\mu_{s_0-1,1}}{\mu_{12}\cdots\mu_{s_0-1,2}}(1 + \frac{\mu_{s_0+1,1}}{\mu_{s_0+1,2}} + \ldots + \frac{\mu_{s_0+1,1}\cdots\mu_{B+1,1}}{\mu_{s_0+1,2}\cdots\mu_{B+1,2}}),$$

$$C_3 = 1 + \frac{\mu_{01}}{\mu_{12}} + \ldots + \frac{\mu_{01}\cdots\mu_{s_0-2,1}}{\mu_{12}\cdots\mu_{s_0-1,2}},$$

$$C_4 = \frac{\mu_{01}\cdots\mu_{s_0-1,1}}{\mu_{12}\cdots\mu_{s_0-1,2}},$$

and

$$C_5 = \frac{\mu_{01}\cdots\mu_{s_0-1,1}}{\mu_{12}\cdots\mu_{s_0-1,2}}(\frac{1}{\mu_{s_0+1,2}} + \ldots + \frac{\mu_{s_0+1,1}\cdots\mu_{B+1,1}}{\mu_{s_0+1,2}\cdots\mu_{B+2,2}}).$$

Comparing the difference of these two throughputs, we have:

$$g' - g = \frac{(C_2C_3 - C_1C_5)(\Delta_1\mu_{s_0,2} + \Delta_2\mu_{s_0,1}) + \Delta_1 C_2 C_4 - \Delta_2 C_1 C_4}{\Theta_2(\Theta_2 - \Delta_2 C_3 + \Delta_1 C_5)}. \tag{4.3}$$

Now suppose we use a dominated team assignment i in state $s_0 \in S$ under an optimal policy $\pi \in \Pi$ which forms an irreducible Markov chain, then the corresponding throughput g^π is constant. Then there exist two other assignments $k, l \in T$ such that $\gamma_{k,1} \geq \gamma_{i,1} \geq \gamma_{l,1}, \gamma_{k,2} \leq \gamma_{i,2} \leq \gamma_{l,2},$. Assume $\gamma_{k,1} - \gamma_{i,1} = \delta_1, \gamma_{i,1} - \gamma_{l,1} = \delta_2, \gamma_{l,2} - \gamma_{i,2} = \delta_3, \gamma_{i,2} - \gamma_{k,2} = \delta_4$. From definition 4.2.1 (iii), we have $0 \leq \delta_2\delta_4 < \delta_1\delta_3$. Consider policies $\pi', \pi'' \in \Pi$ such that $\pi'(s) = \pi(s), \pi''(s) = \pi(s)$ for $\forall s \neq s_0$, and $\pi'(s_0) = k, \pi''(s_0) = l$. Then the Markov chains generated by π' and π'' have a single set of recurrent states within $\{0, \ldots, B+2\}$, and have constant throughputs. Denote the corresponding throughputs as $g^{\pi'}$ and $g^{\pi''}$, respectively.

If $\gamma_{i,1} = 0$, then $s_0 = B+2$ (otherwise π would have two recurrent classes and can not be optimal). Moreover, when $\gamma_{i,1} = 0$, then $\gamma_{l,1} = 0, 0 < \gamma_{i,2} \leq \gamma_{l,2}$. Since $s_0 = B+2$ is on the boundary of the birth-death process, by using policy π'' instead of π, we can reduce the time we spend in that state and increase the reward (i.e., service rate at station 2) without changing anything else. Thus, $g^\pi < g^{\pi''}$, π is not optimal. Similarly, we can obtain this result when $\gamma_{i,2} = 0$.

Next, consider the case when $\gamma_{i,1}, \gamma_{i,2} > 0$. If we substitute $\mu_{s,j}$ with the service rate at station $j \in \{1,2\}$ in state $s \in S$ under policy π, then $\mu_{s_0,j} = \gamma_{i,j}$ for $j \in \{1,2\}$, and g^π is given in equation (4.3). Moreover, we can obtain the following equations (A.1) and (4.5) by plugging in $\Delta_1 = \delta_1, \Delta_2 = \delta_4$ and $\Delta_1 = -\delta_2, \Delta_2 = -\delta_3$ to equation (4.3), respectively.

$$g^{\pi'} - g^\pi = \frac{(C_2C_3 - C_1C_5)(\delta_1\gamma_{i,2} + \delta_4\gamma_{i,1}) + \delta_1 C_2 C_4 - \delta_4 C_1 C_4}{\Theta_2(\Theta_2 - \delta_4 C_3 + \delta_1 C_5)}, \tag{4.4}$$

and

$$g^{\pi''} - g^\pi = \frac{(C_2C_3 - C_1C_5)(-\delta_2\gamma_{i,2} - \delta_3\gamma_{i,1}) - \delta_2 C_2 C_4 + \delta_3 C_1 C_4}{\Theta_2(\Theta_2 + \delta_3 C_3 - \delta_2 C_5)}. \tag{4.5}$$

Note that the denominators of equations (A.1) and (4.5) are positive. Moreover, $\delta_1\delta_3 > 0$, so $\delta_1\gamma_{i,2} + \delta_4\gamma_{i,1} > 0$, and $\delta_2\gamma_{i,2} + \delta_3\gamma_{i,1} > 0$. Consider the following positive linear combination of $g^{\pi'} - g^{\pi}$ and $g^{\pi''} - g^{\pi}$:

$$(g^{\pi'} - g^{\pi})\lambda_1 + (g^{\pi''} - g^{\pi})\lambda_2 = \frac{C_4(C_1\gamma_{i,2} + C_2\gamma_{i,1})(\delta_1\delta_3 - \delta_2\delta_4)}{(\delta_1\gamma_{i,2} + \delta_4\gamma_{i,1})(\delta_2\gamma_{i,2} + \delta_3\gamma_{i,1})}, \tag{4.6}$$

where

$$\lambda_1 = \frac{\Theta_2(\Theta_2 - \delta_4 C_3 + \delta_1 C_5)}{(\delta_1\gamma_{i,2} + \delta_4\gamma_{i,1})} > 0,$$

$$\lambda_2 = \frac{\Theta_2(\Theta_2 + \delta_3 C_3 - \delta_2 C_5)}{(\delta_2\gamma_{i,2} + \delta_3\gamma_{i,1})} > 0.$$

Since $\delta_2\delta_4 < \delta_1\delta_3$,

$$(g^{\pi'} - g^{\pi})\lambda_1 + (g^{\pi''} - g^{\pi})\lambda_2 > 0.$$

Thus, at least one of $g^{\pi'} - g^{\pi}$ and $g^{\pi''} - g^{\pi}$ must be positive. That is, at least one of $g^{\pi'}$ and $g^{\pi''}$ should be greater than g^{π}, and policy π cannot be optimal. □

Theorem 4.2.1 implies that dominated team assignments are replaceable. The following proposition shows that, when the inequality in (iii) of Definition 4.2.1 becomes equality, even though the team assignment is no longer dominated, it is replaceable.

Proposition 4.2.1. *Team assignment i is replaceable if there exist two other assignments k, l such that*

1. $\gamma_{k,1} \geq \gamma_{i,1} \geq \gamma_{l,1}$,
2. $\gamma_{k,2} \leq \gamma_{i,2} \leq \gamma_{l,2}$,
3. $(\gamma_{i,1} - \gamma_{l,1})(\gamma_{i,2} - \gamma_{k,2}) = (\gamma_{k,1} - \gamma_{i,1})(\gamma_{l,2} - \gamma_{i,2})$

Proof. Suppose there such exist $i, k, l \in T$. Considering the same notations as in the proof of Theorem 4.2.1, but now we have $\delta_2\delta_4 = \delta_1\delta_3$. If $\delta_1 = \delta_4 = 0$ or $\delta_2 = \delta_3 = 0$, then team assignment i is equivalent to one or both of team assignments k, l, the result is trivial. Otherwise, $\delta_1\gamma_{i,2} + \delta_4\gamma_{i,1} > 0$, and $\delta_2\gamma_{i,2} + \delta_3\gamma_{i,1} > 0$. By equation (4.6),

$$(g^{\pi'} - g^{\pi})\lambda_1 + (g^{\pi''} - g^{\pi})\lambda_2 = 0.$$

Thus, at least one of $g^{\pi'} - g^{\pi}$ and $g^{\pi''} - g^{\pi}$ must be non-negative. That is, at least one of π' and π'' is as good as π and team assignment i is replaceable. □

4.2.2 Optimal policy

After our preliminary selection in the previous section, we remove the team assignments that are not on the Pareto boundary or dominated by some other team assignments. The next definition provides the set of potential optimal policies.

Definition 4.2.2. *(Optimal Assignment Set) A set of team assignments $T^* \subseteq T$ is called the optimal assignment set, if all the assignments in T^* are not replaceable.*

By Theorems 4.1.1 and 4.2.1, we can find an optimal policy among team assignments in the optimal assignment set, but so far it is not clear when and how we should use these assignments. To present our optimal policy in a concise way, we first renumber the assignments in T^*.

Proposition 4.2.2. *Let $|T^*| = N_t$. We can then number the assignments in T^* such that if $\gamma_{1,1} \geq \gamma_{2,1} \geq \ldots \geq \gamma_{N_t,1}$, then $\gamma_{1,2} \leq \gamma_{2,2} \leq \ldots \leq \gamma_{N_t,2}$, and $\frac{\gamma_{N_t-1,1}-\gamma_{N_t,1}}{\gamma_{N_t,2}-\gamma_{N_t-1,2}} > \ldots > \frac{\gamma_{2,1}-\gamma_{3,1}}{\gamma_{3,2}-\gamma_{2,2}} > \frac{\gamma_{1,1}-\gamma_{2,1}}{\gamma_{2,2}-\gamma_{1,2}}$.*

Proof. By definition, if two different assignments $i, k \in T^*$, then they are on the Pareto boundary, and thus if $\gamma_{i,1} \geq \gamma_{k,1}$, we must have $\gamma_{i,2} \leq \gamma_{k,2}$. Otherwise team i would have higher service rates than team k at both stations and thus k is not on the Pareto boundary. So, we can obtain $\gamma_{1,2} \leq \gamma_{2,2} \leq \ldots \leq \gamma_{N,2}$ in this manner. By definition, if team assignments i, k, l are irreplaceable, and $\gamma_{k,1} \geq \gamma_{i,1} \geq \gamma_{l,1}, \gamma_{k,2} \leq \gamma_{i,2} \leq \gamma_{l,2}$, then $\frac{\gamma_{i,1}-\gamma_{l,1}}{\gamma_{l,2}-\gamma_{i,2}} > \frac{\gamma_{k,1}-\gamma_{i,1}}{\gamma_{i,2}-\gamma_{k,2}}$. And thus we can obtain $\frac{\gamma_{N_t-1,1}-\gamma_{N_t,1}}{\gamma_{N_t,2}-\gamma_{N_t-1,2}} > \ldots > \frac{\gamma_{2,1}-\gamma_{3,1}}{\gamma_{3,2}-\gamma_{2,2}} > \frac{\gamma_{1,1}-\gamma_{2,1}}{\gamma_{2,2}-\gamma_{1,2}}$ in this manner. □

Now we are ready to present the optimal policy as follows.

Theorem 4.2.2. *(Optimal Policy) When $|T^*| = N_t$, and the team assignments in T^* have been reordered as in Proposition 4.2.2, then for any optimal policy $\pi^* \in \Pi^*$, there exist $i_0^*, \ldots, i_{B+2}^*$ with $1 = i_{B+2}^* \leq i_{B+1}^* \leq \ldots \leq i_0^* = N_t$ such that $\pi^*(s) = N_t + 1 - i_s^*$ for all $s \in S$.*

Proof. When $N_t = 1$, the problem is trivial. When $N_t > 1$, consider the following problem P:

Suppose in our current tandem queueing network, instead of M servers with N_t possible optimal team assignments, we now have $N_t + 1$ servers. Assume the service rate of server $i \in \{1, \ldots, N_t + 1\}$ at station $j \in \{1, 2\}$ is μ_{ij}, and when they work together, their service rates are additive. The goal is again dynamically assign the servers to the stations to maximize the long-run average throughput of the system.

Let $\mu_{11} = \gamma_{N_t,1}, \mu_{k,1} = \gamma_{N_t+1-k,1} - \gamma_{N_t+2-k,1}$ for $k = 2, \ldots, N_t$, $\mu_{N_t+1,1} = 0$, $\mu_{12} = 0, \mu_{k,2} = \gamma_{N_t+2-k,2} - \gamma_{N_t+1-k,2}$ for $k = 2, \ldots, N_t$, and $\mu_{N_t+1,2} = \gamma_{12}$. Then by the definition of T^*, we have $\mu_{ij} \geq 0$, for any $i \in \{1, \ldots, N_t + 1\}, j \in \{1, 2\}$, and $\frac{\mu_{12}}{\mu_{11}} < \frac{\mu_{22}}{\mu_{21}} < \ldots < \frac{\mu_{N_t+1,2}}{\mu_{N_t+1,1}}$.

Let Π_P be the set of all stationary deterministic policies of Problem P, and Π_P^* be the set of optimal policies of Problem P.

Assume that the optimal long-run average throughput of Problem P is g_P^*, and the optimal long-run average throughput of the original problem is g_O^*. Note that, for $k \in \{1, \ldots, N_t\}, \sum_{i=1}^{k} \mu_{i,1} = \gamma_{N_t+1-k,1}, \sum_{i=k+1}^{N_t+1} \mu_{i,2} = \gamma_{N_t+1-k,2}$, that is, all the N_t team assignments in T^* correspond to feasible assignments in Problem P. So $g_O^* \leq g_P^*$, and the optimal long-run average throughput of the original problem is bounded by g_P^*.

Next, we will show that we can reach this upper bound g_P^*. According to Theorem 4 of Hasenbein and Kim [28], for any optimal policy $\pi_P^* \in \Pi_P^*$ of Problem P, there exist thresholds $i_0, \ldots, i_{B+2}$ with $0 = i_{B+2} < i_{B+1} \leq \ldots \leq i_1 < i_0 = N_t + 1$ such that, in state $s \in S$, the optimal policy is to assign servers $1, \ldots, i_s$ to Station 1, and the rest of the servers to Station 2. We can observe that, in this optimal policy π_P^*:

- When the system is in state $s \in S \setminus \{0, B+2\}$, $1 \leq i_s \leq N_t$, the combined service rate of Station 1 is $\sum_{i=1}^{i_s} \mu_{i,1} = \gamma_{N_t+1-i_s,1}$, and the combined service rate of Station 2 is $\sum_{i=i_s+1}^{N_t+1} \mu_{i,2} = \gamma_{N_t+1-i_s,2}$, which is equivalent to using team assignment N_t+1-i_s in T^*;

- When the system is in state $s = 0$, the combined service rate of Station 1 is $\sum_{i=1}^{i_s} \mu_{i,1} = \sum_{i=1}^{N_t+1} \mu_{i,1} = \gamma_{1,1}$, the combined service rate of Station 2 is 0, which is equivalent to using team assignment 1 in T^* since there is no work to do at Station 2;

- When the system is in state $s = B+2$, the combined service rate at Station 1 is 0, the combined service rate at Station 2 is $\sum_{i=i_s+1}^{N_t+1} \mu_{i,2} = \sum_{i=1}^{N_t+1} \mu_{i,2} = \gamma_{N_t,2}$, which is equivalent to using team assignment N_t in T^* since Station 1 is blocked.

Thus, we can attain g_P^* in the original problem using the following policy π^*. Let $i_0^* = N_t, i_s^* = i_s$ for $s \in S \setminus \{0, B+2\}, i_{B+2}^* = 1$, and set $\pi^*(s) = N_t + 1 - i_s^*$ for all $s \in S$. Then, π^* is optimal. □

The following corollary follows immediately from Theorem 4.2.2.

Corollary 4.2.1. *For any optimal policy π^*, there exist $s_1^*, \ldots, s_{N_t-1}^*$ with $0 \leq s_1^* \leq \ldots \leq s_{N_t-1}^* \leq B+2$ such that*

$$\pi^*(s) = \begin{cases} 1 & \text{if } 0 \leq s \leq s_1^*, \\ 2 & \text{if } s_1^* < s \leq s_2^*, \\ \vdots & \quad\vdots \\ N_t & \text{if } s_{N_t-1}^* < s \leq B+2. \end{cases}$$

for all $s \in S$.

Note that, the theorem does not provide a methodology in how to determine the thresholds, but it significantly reduces the number of policies need to be evaluated from N_t^{B+2}

to $\binom{B+N_t}{N_t-1}$. This also explains why we provide conditions in reducing the size of N_t in the previous two sections. The following example illustrates how to use our method to find out the optimal policy.

Example 4.2.1. *Assume the buffer size* $B = 1$. *Then,* $S = \{0, 1, 2, 3\}$. *Suppose now we have 20 team assignments with service rates of* i*th team at station* $j \in \{1, 2\}$, $\gamma_{i,j}$, *for* $i \in \{1, \ldots, 20\}$ *and* $\gamma_{i,j}$ *is independently and randomly generated from* $\{1, 2, \ldots, 10\}$. *Specifically, the service rates of these team assignments are shown in the following scatter plot.*

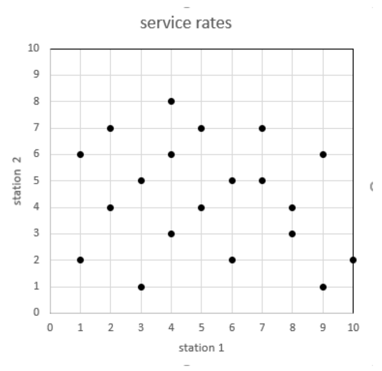


Figure 4.1: Service rates of 20 team assignments

First, we remove the team assignments that are not on Pareto boundary or dominated by others and obtain T^*. *There are 4 team assignments in* T^* *and the corresponding service rates are* $(\gamma_{1,1}, \gamma_{1,2}) = (10, 2)$, $(\gamma_{2,1}, \gamma_{2,2}) = (9, 6)$, $(\gamma_{3,1}, \gamma_{3,2}) = (7, 7)$, *and* $(\gamma_{4,1}, \gamma_{4,2}) = (4, 8)$. *Then we can find the optimal policy using Theorem 4.2.2. In Table 1, we evaluate the 10 possible policies and mark the optimal throughput with *. The optimal police is:*

$$\pi^*(s) = \begin{cases} 1 & \textit{if } s = 0, \\ 2 & \textit{if } s = 1, \\ 3 & \textit{if } s = 2, \\ 4 & \textit{if } s = 3. \end{cases}$$

Table 4.1: Throughputs of model in Example 4.2.1.

team assignment				throughput
$s = 0$	$s = 1$	$s = 2$	$s = 3$	
1	1	1	4	4.9799
1	1	2	4	5.6942
1	1	3	4	5.6722
1	1	4	4	5.5285
1	2	2	4	5.9530
1	2	3	4	5.9838*
1	2	4	4	5.9316
1	3	3	4	5.8741
1	3	4	4	5.8091
1	4	4	4	5.4902

From this example we can see that, using our method, we can first largely reduce the number of team assignments from 20 to 4, and then we reduce the number of total possible assignment policies of these 4 teams from $64 (= 4^3)$ to 10 by applying Theorem 4.2.2 and find out the optimal policy efficiently.

Note that, if $|T^*| = 1$, i.e., i^* is the only assignment left, then it is optimal to always choose this team assignment in all states. In the next section, we discuss the case when $|T^*| = 1$ is satisfied.

4.2.3 Permanent Team Assignment

In this section, we obtain the conditions when permanent team assignment is optimal, that is, when it is optimal for the servers to be static. If we use team assignment $t \in T$ at all times, then the corresponding throughput of this permanent assignment is:

$$g_s^t = \frac{\gamma_{t,1}\gamma_{t,2}\sum_{k=0}^{B+1}(\gamma_{t,1})^k(\gamma_{t,2})^{B+1-k}}{\sum_{k=0}^{B+2}(\gamma_{t,1})^k(\gamma_{t,2})^{B+2-k}}.$$

Theorem 4.2.3. *A permanent team assignment is optimal if and only if there exists team i such that $\gamma_{i,1} \geq \gamma_{j,1}, \gamma_{i,2} \geq \gamma_{j,2}$ for all $j \in T$, $j \neq i$, and the policy π^* such that $\pi^*(s) = i$ for all $s \in S$ is optimal.*

Proof. If such team assignment i exist, then $T^* = \{i\}$ since i is the only team assignment on the Pareto boundary. Then $|T^*| = 1$ and π^* is optimal.

If such team i does not exist, let $j = \underset{k\in T}{\text{argmin}}\gamma_{k,1}$ and $l = \underset{k\in T}{\text{argmin}}\gamma_{k,2}$, then $j \neq l$. Any optimal policy π^* has $\pi^*(0) = j, \pi^*(B+2) = l$. Thus, any permanent team assignment cannot be optimal.

□

Intuitively, when there exists a team assignment that has higher service rates at all stations than all the other team assignments, then it is optimal to use that team assignment at all times. And if such team assignment does not exist, it would be better to let the servers to be flexible to take full advantage of the specialty of different team assignments.

4.3 Systems with Proportional Rates

In this section, we want to explore the optimal policy for cases with some special structures. In Section 4.3.1, we consider a special case when the combined service rates of a team is proportional to the sum of their service rates with coefficient α, and provide the optimal policy when $\alpha = 1$. We also discuss the property of the optimal policy when $\alpha \neq 1$. In Section 5.3.1, we explore the optimal policy when there are constraints on team formation.

4.3.1 Proportional Team Service Rates

In this section, we consider the case when the combined service rate of a group of servers is proportional to the sum of their service rates (with coefficient α). Suppose the service rate of server $i \in I = \{1, 2, \ldots, M\}$ at station $j \in \{1, 2\}$ is μ_{ij}, and we label the servers such that $\frac{\mu_{12}}{\mu_{11}} \le \frac{\mu_{22}}{\mu_{21}} \le \ldots \le \frac{\mu_{M,2}}{\mu_{M,1}}$. Define the set of possible team assignments as $T = \{(t_1, \ldots, t_M) : t_i \in \{0, 1, 2\}, \forall i \in I\}$, where $t_i = 0$ if the ith server is idled, and $t_i = j$ if the ith server is assigned to station j, for $j = 1, 2$. We obtain the following theorem by applying our optimal policy.

Theorem 4.3.1. *When $\alpha = 1$ (i.e., the service rates are additive), there exist thresholds $i_0, \ldots, i_{B+2}$ with $0 = i_{B+2} < i_{B+1} \le \ldots \le i_1 < i_0 = M$ such that, in state $s \in S$, the optimal policy is to assign servers $1, \ldots, i_s$ to station 1, and the remaining servers to station 2.*

Proof. First, we prove that

$$T^* \subseteq \{t^0 = (1, \ldots, 1, 1), t^1 = (1, \ldots, 1, 2), \ldots, t^{M-1} = (1, 2, \ldots, 2), t^M = (2, 2, \ldots, 2)\}.$$

That is, for any $t = (t_1, \ldots, t_M) \in T^*$, if $t_i = 1$ for some $i \in I$, then $t_k = 1$ for $1 \le k \le i$; if $t_i = 2$ for some $i \in I$, then $t_k = 2$ for $i \le k \le M$. We prove by contradiction.

Suppose there exists a team assignment $t \in T$ such that $k < i$ and $t_k = 2, t_i = 1$. Then the team service rates of t at station j is $\gamma_{t,j} = \sum_{l=1}^{M} \mu_{l,j} 1\{t_l = j\}$ for $j = 1, 2$. If $\gamma_{t,j} > 0$ for $j = 1, 2$, we consider two other team assignments t' and t''. In particular, $t'_k = 1$, and $t'_l = t_l$ for $l \in I \setminus \{k\}$; $t''_i = 2$, and $t''_l = t_l$ for $l \in I \setminus \{i\}$. Then, we obtain that

$$\gamma_{t',1} = \gamma_{t,1} + \mu_{k,1}, \gamma_{t',2} = \gamma_{t,2} - \mu_{k,2};$$

$$\gamma_{t'',1} = \gamma_{t,1} - \mu_{i,1}, \gamma_{t'',2} = \gamma_{t,2} + \mu_{i,2}.$$

Thus, $\gamma_{t',1} \geq \gamma_{t,1} \geq \gamma_{t'',1}$, $\gamma_{t',2} \leq \gamma_{t,2} \leq \gamma_{t'',2}$, and $\frac{\mu_{k,2}}{\mu_{k,1}} \leq \frac{\mu_{i,2}}{\mu_{i,1}}$ implies that

$$(\gamma_{t,1} - \gamma_{t'',1})(\gamma_{t,2} - \gamma_{t',2}) = \mu_{i,1}\mu_{k,2} \leq \mu_{k,1}\mu_{i,2} = (\gamma_{t',1} - \gamma_{t,1})(\gamma_{t'',2} - \gamma_{t,2}).$$

By Theorem 4.2.1 and Proposition 4.2.1, t is replaceable, and thus $t \notin T^*$.

If $\gamma_{t,1} = 0$, since $t_i = 1$, we have $\mu_{i,1} = 0$, and $\mu_{i,2} > 0$ (otherwise server i should be removed from the set of servers). Then $\gamma_{t'',1} = \gamma_{t,1} - \mu_{i,1} = \gamma_{t,1}$, $\gamma_{t'',2} = \gamma_{t,2} + \mu_{i,2} > \gamma_{t,2}$. Thus, team assignment t is not on the Pareto boundary, and $t \notin T^*$. Similarly, we can obtain that $t \notin T^*$ if $\gamma_{t,2} = 0$.

Now we have proved that $T^* = \{t^0, \ldots, t^M\}$, and $\gamma_{t^m,1} = \sum_{l=1}^{M-m} \mu_{l,1}$, $\gamma_{t^m,2} = \sum_{l=M+1-m}^{M} \mu_{l,2}$, for $m = 0, 1, \ldots, M$. It is easy to check that with the current order, the team assignments in T^* satisfy the three conditions in Proposition 4.2.2 and thus we can obtain the desired result by applying Theorem 4.2.2. □

Note that, although the optimal policy when $\alpha = 1$ is given in Theorem 4 of Hasenbein and Kim [28], the result can be easily obtained using our method.

The following proposition provides the optimal assignment set T^* when $M = 2$. We introduce some notation to better illustrate the results. Let $\Sigma_1 = \mu_{11} + \mu_{21}, \Sigma_2 = \mu_{12} + \mu_{22}$; then the combined service rate of servers working together at subtask i is $\alpha\Sigma_i$ for $i = 1, 2$. Let $\mu_{11} = \beta_1\Sigma_1, \mu_{21} = (1 - \beta_1)\Sigma_1, \mu_{12} = \beta_2\Sigma_2, \mu_{22} = (1 - \beta_2)\Sigma_2$, then $\beta_j \in [0, 1]$ is the fraction of server 1 of the total service rate on subtask j, for $j = 1, 2$. Moreover, let

$$m_1 = \frac{\beta_1 + \beta_2 - 1}{2\beta_2 - 1}, \text{ when } \beta_2 \neq \frac{1}{2}.$$
$$m_2 = \frac{\beta_1 + \beta_2 - 1}{2\beta_1 - 1}, \text{ when } \beta_1 \neq \frac{1}{2}.$$

Moreover, our assumption of $\frac{\mu_{12}}{\mu_{11}} \leq \frac{\mu_{22}}{\mu_{21}}$ can be reorganized as $\beta_1 \geq \beta_2$.

Proposition 4.3.1. *When $\beta_1 \geq \beta_2$,*

1. *If $\alpha \geq 1 + \beta_1 - \beta_2$, then $T^* = \{(1, 1), (2, 2)\}$.*

2. *If* $\alpha < 1 + \beta_1 - \beta_2$, *then*

   (a) *if* $\beta_1 \geq \beta_2 > \frac{1}{2}$, $2\beta_1 - 2\beta_1^2 - 1 + \beta_2 \geq 0$,

$$T^* = \begin{cases} \{(1,1),(1,2),(2,2)\} & \textit{if } m_2 \leq \alpha < 1 + \beta_1 - \beta_2, \\ \{(1,1),(1,2),(2,1),(2,2)\} & \textit{if } \beta_1 \leq \alpha < m_2, \\ \{(1,2),(2,1),(2,2)\} & \textit{if } \beta_2 \leq \alpha < \beta_1, \\ \{(1,2),(2,1)\} & \textit{if } 0 < \alpha < \beta_2. \end{cases}$$

   (b) *if* $\beta_1 \geq \beta_2 > \frac{1}{2}$, $2\beta_1 - 2\beta_1^2 - 1 + \beta_2 < 0$,

$$T^* = \begin{cases} \{(1,1),(1,2),(2,2)\} & \textit{if } \beta_1 \leq \alpha < 1 + \beta_1 - \beta_2, \\ \{(1,2),(2,2)\} & \textit{if } m_2 \leq \alpha < \beta_1, \\ \{(1,2),(2,1),(2,2)\} & \textit{if } \beta_2 \leq \alpha < m_2, \\ \{(1,2),(2,1)\} & \textit{if } 0 < \alpha < \beta_2. \end{cases}$$

   (c) *if* $\beta_1 \geq \frac{1}{2} \geq \beta_2$, $\beta_1 + \beta_2 \geq 1$,

$$T^* = \begin{cases} \{(1,1),(1,2),(2,2)\} & \textit{if } \beta_1 \leq \alpha < 1 + \beta_1 - \beta_2, \\ \{(1,2),(2,2)\} & \textit{if } 1 - \beta_2 \leq \alpha < \beta_1, \\ \{(1,2)\} & \textit{if } 0 < \alpha < 1 - \beta_2. \end{cases}$$

   (d) *if* $\beta_1 \geq \frac{1}{2} \geq \beta_2$, $\beta_1 + \beta_2 < 1$,

$$T^* = \begin{cases} \{(1,1),(1,2),(2,2)\} & \textit{if } 1 - \beta_2 \leq \alpha < 1 + \beta_1 - \beta_2, \\ \{(1,1),(1,2)\} & \textit{if } \beta_1 \leq \alpha < 1 - \beta_2, \\ \{(1,2)\} & \textit{if } 0 < \alpha < \beta_1. \end{cases}$$

(e) if $\frac{1}{2} > \beta_1 \geq \beta_2$, $2\beta_2 - 2\beta_2^2 - \beta_1 \geq 0$,

$$T^* = \begin{cases} \{(1,1),(1,2),(2,2)\} & \text{if } m_1 \leq \alpha < 1+\beta_1-\beta_2, \\ \{(1,1),(1,2),(2,1),(2,2)\} & \text{if } 1-\beta_2 \leq \alpha < m_1, \\ \{(1,1),(1,2),(2,1)\} & \text{if } 1-\beta_1 \leq \alpha < 1-\beta_2, \\ \{(1,2),(2,1)\} & \text{if } 0 < \alpha < 1-\beta_1. \end{cases}$$

(f) if $\frac{1}{2} > \beta_1 \geq \beta_2$, $2\beta_2 - 2\beta_2^2 - \beta_1 < 0$,

$$T^* = \begin{cases} \{(1,1),(1,2),(2,2)\} & \text{if } 1-\beta_2 \leq \alpha < 1+\beta_1-\beta_2, \\ \{(1,1),(1,2)\} & \text{if } m_1 \leq \alpha < 1-\beta_2, \\ \{(1,1),(1,2),(2,1)\} & \text{if } 1-\beta_1 \leq \alpha < m_1, \\ \{(1,2),(2,1)\} & \text{if } 0 < \alpha < 1-\beta_1. \end{cases}$$

Proof. From Proposition 4.1.1 and Theorem 4.2.1, we can obtain that

1. $(1,1) \in T^*$ if and only if $\alpha > \max\{\beta_1, 1-\beta_1\}$.
2. $(2,2) \in T^*$ if and only if $\alpha > \max\{\beta_2, 1-\beta_2\}$.
3. $(1,2) \in T^*$ if at least one of the following cases holds:
   (a) $\beta_1 \geq \beta_2 > \frac{1}{2}, \alpha < \min\{m_1, 1+\beta_1-\beta_2\}$;
   (b) $\beta_1 \geq \frac{1}{2} \geq \beta_2, \alpha < 1+\beta_1-\beta_2$;
   (c) $\frac{1}{2} > \beta_1 \geq \beta_2, \alpha < \min\{m_2, 1+\beta_1-\beta_2\}$.
4. $(2,1) \in T^*$ if at least one of the following cases holds:
   (a) $\beta_1 \geq \beta_2 > \frac{1}{2}, \alpha < \min\{m_2, 1-\beta_1+\beta_2\}$;
   (b) $\beta_1 \geq \frac{1}{2} \geq \beta_2, \alpha < 1-\beta_1+\beta_2$;

(c) $\frac{1}{2} > \beta_1 \geq \beta_2, \alpha < \min\{m_1, 1-\beta_1+\beta_2\}$.

By comparing the values of these thresholds shown in the inequalities above, and reorganizing the above conditions, we can get the desired results. □

Note that, once we determine T^*, the optimal policy follows from Theorem 4.2.2.

Remark 4.3.1. *Proposition 4.3.1 coincides with Theorem 2.1 of Andradóttir, Ayhan, and Down [11] and Theorems 3.1-3.4, 4.1-4.2, and 4.4 of Andradóttir, Ayhan, and Down [13]. In particular, our results provide more specific information of team selection than Theorems 4.2 and 4.4 of Andradóttir, Ayhan, and Down [13] since we have two extra thresholds* m_1, m_2. *For instance, when* $\beta_1 \geq \beta_2 > \frac{1}{2}$ *and* $\beta_1 \leq \alpha \leq 1$, *Theorems 4.4 of [13] claims that* $T^* = \{(1,1),(1,2),(2,1),(2,2)\}$*; while we eliminate team assignment* $(2,1)$ *from* T^* *when either* $2\beta_1 - 2\beta_1^2 - 1 + \beta_2 \geq 0, m_2 \leq \alpha < 1+\beta_1-\beta_2$ *or* $2\beta_1 - 2\beta_1^2 - 1 + \beta_2 < 0$ *and further reduce the number of team assignments in* T^*.

Next, we consider the case when the servers are *generalists*. In particular, assume the service rate of server i at station $j \in \{1,2\}$ is $\mu_{ij} = \mu_i\gamma_j$, and the servers have proportional combined service rates with coefficient α. Then Proposition 4.3.1 can be simplified as follows.

Proposition 4.3.2. *When* $\mu_{ij} = \mu_i\gamma_j$, *then* $\beta_1 = \beta_2$, *and*

1. *If* $\beta_1 = \beta_2 \neq \frac{1}{2}$, *then*

$$T^* = \begin{cases} \{(1,1),(2,2)\} & \textit{if } 1 \leq \alpha, \\ \{(1,1),(1,2),(2,1),(2,2)\} & \textit{if } \max\{\beta_1, 1-\beta_1\} \leq \alpha < 1, \\ \{(1,2),(2,1)\} & \textit{if } 0 < \alpha < \max\{\beta_1, 1-\beta_1\}. \end{cases}$$

2. *If* $\beta_1 = \beta_2 = \frac{1}{2}$, *then*

$$T^* = \begin{cases} \{(1,1),(2,2)\} & \text{if } 1 \leq \alpha, \\ \{(1,1),(1,2),(2,2)\} & \text{if } \frac{1}{2} \leq \alpha < 1, \\ \{(1,2)\} & \text{if } 0 < \alpha < \frac{1}{2}. \end{cases}$$

Proof. When $\mu_{ij} = \mu_i \gamma_j$, we have $m_1 = m_2 = 1$, $\beta_1 = \beta_2 > 0$, and

$$2\beta_1 - 2\beta_1^2 - 1 + \beta_2 = (2\beta_1 - 1)(1 - \beta_1) > 0 \text{ when } \beta_1 = \beta_2 > \frac{1}{2},$$
$$2\beta_2 - 2\beta_2^2 - \beta_1 = \beta_2(1 - 2\beta_2) > 0 \text{ when } \frac{1}{2} > \beta_1 = \beta_2.$$

And we can obtain the desired results by plugging in the above equalities and inequalities.

□

Note that, Proposition 4.3.2 coincide with Theorem 2.1 of Andradóttir, Ayhan, and Down [11].

4.3.2 Teams of Specialized Servers

In this section, we consider the case when there are constraints on team formation. Suppose now we have K types of servers with different specialties, and we need exactly one server of each type to work as a team at each station.

Specifically, consider N stations, then we need N servers of each type. Let μ_{ij}^k be the service rate of ith server of type k working at station j, where $i, j \in \{1, \ldots, N\}, k \in \{1, \ldots, K\}$. Assume that $\mu_{ij}^k > 0$ for any i, k, j. A team's service rate is proportional to the sum of the service rates of its team members with coefficient α. Our objective is to find out the optimal team assignments that maximize the long-run throughput of this system.

In this section, we focus on the case $N = 2$. First, we can label servers such that $\mu_{11}^k \geq \mu_{21}^k$ for all k without loss of generality. We present the server allocation by a

$K-$dimensional vector $A = (a_1, \ldots, a_K)$, where a_k denotes the station that the first server of type k is assigned to. Let $S_i(A)$ be the team service rate at station $i \in \{1, 2\}$ of server allocation A, that is, $S_1(A) = \alpha \sum_{k=1}^{K}((2 - a_k)\mu_{11}^k + (a_k - 1)\mu_{21}^k)$ and $S_2(A) = \alpha \sum_{k=1}^{K}((a_k - 1)\mu_{12}^k + (2 - a_k)\mu_{22}^k)$.

Theorem 4.3.2. *If $\mu_{12}^{k'} < \mu_{22}^{k'}$ for some type k', we will always assign the first server of type k' to station 1, and second server of type k' to station 2 in any optimal policy.*

Proof. If on the contrary, we use team assignment $A_i = (a_{i,1}, \ldots, a_{i,K})$ in some policy π at state $s \in S$ with $a_{i,k'} = 2$, then we can find another assignment A_j such that $a_{j,k} = a_{i,k}$ for $\forall k \neq k'$, and $a_{j,k'} = 1$. Then $S_1(A_j) - S_1(A_i) = \alpha(\mu_{11}^{k'} - \mu_{21}^{k'}) \geq 0, S_2(A_j) - S_2(A_i) = \alpha(\mu_{22}^{k'} - \mu_{12}^{k'}) > 0$, and we can increase the service rates at both stations by using assignment A_j instead of A_i. Thus, this policy π can not be optimal, and the proof is complete. $\square$

Theorem 4.3.2 shows the optimal server assignment for server of type k such that $\mu_{11}^k \geq \mu_{21}^k$ and $\mu_{12}^k < \mu_{22}^k$. Indeed, if the two servers of type k are better at different stations, we will always assign the server to the station where they work faster to obtain larger team service rates at both stations.

Now, we need to find the optimal assignment for the servers of type k such that $\mu_{12}^k \geq \mu_{22}^k$. Without loss of generality, assume that $\mu_{12}^k \geq \mu_{22}^k$ for $k = 1, 2, \ldots, K_0$. That is, for the first K_0 types of servers, one server (by our assumption, i.e. the first server) dominates the other one at both stations. Let $d_k = \frac{\mu_{11}^k - \mu_{21}^k}{\mu_{12}^k - \mu_{22}^k}$, then we can reorder the types of servers such that $d_1 \leq d_2 \leq \ldots \leq d_{K_0}$. Denote $A_0 = (1, \ldots, 1), A_1 = (2, 1, \ldots, 1), A_2 = (2, 2, 1, \ldots, 1), \ldots, A_{K_0} = (2, \ldots, 2, 1, \ldots, 1)$, that is, for $i \in \{0, \ldots, K_0\}$, in assignment A_i, $a_{ik} = 2$, for $k = 1, \ldots, i$; $a_{ik} = 1$ for $k > i$. Then the optimal policy is given in the following theorem.

Theorem 4.3.3. *For any optimal policy π^*, there exist $s_1^*, \ldots, s_{K_0}^*$ with $1 \leq s_1^* \leq \ldots \leq$*

$s^*_{K_0} \leq B+1$ *such that*

$$\pi^*(s) = \begin{cases} A_0 & \textit{if } 0 \leq s \leq s^*_1, \\ A_1 & \textit{if } s^*_1 < s \leq s^*_2, \\ \vdots & \qquad \vdots \\ A_{K_0-1} & \textit{if } s^*_{K_0-1} < s \leq s^*_{K_0}, \\ A_{K_0} & \textit{if } s^*_{K_0} < s \leq B+2. \end{cases}$$

for all $s \in S$.

Proof. It is trivial when $K_0 = 0$. When $K_0 > 0$, first note that, by Theorem 4.3.2, for any assignment $A = (a_1, \ldots, a_K)$ used in any optimal policy, it must satisfy $a_k = 1$ for $k = K_0+1, \ldots, K$. Denote the set of all such team assignments as T^0. For any assignment $A \in T^0$, $A \neq A_0$, and $A \neq A_{K_0}$, there exists type $1 \leq k_1, k_2 \leq K_0$ such that $a_{k_1} = 2$, and $a_{k_2} = 1$. Then, we can find assignments $A^1, A^2 \in T^0$ such that $a^1_k = a^2_k = a_k$ for $k \neq k_1, k_2$, $a^1_{k_1} = a^1_{k_2} = 1$, $a^2_{k_1} = a^2_{k_2} = 2$. Note that, by Theorem 4.2.1, A is dominated by A^1 and A^2 if $d_{k_2} < d_{k_1}$, which leads to $k_2 < k_1$. Thus, $T^* \subseteq \{A_0, \ldots, A_{K_0}\}$. The rest of the proof follows from Theorem 4.2.2. $\square$

Intuitively, d_k can be regarded as the ratio of loss at station 1 and gain at station 2 if we switch the two servers of type k when the first server is at station 1 and the second server is at station 2, initially. Note that, the first server of any type k works faster than the second server. Furthermore, when the number of jobs in the buffer is small, we want to obtain higher throughput by pushing more jobs into the system, and so we assign the server with higher service rate at station 1 (i.e., the first server) for each type to station 1. As the number of jobs waiting in the buffer (to be served) at station 2 increases, we want to obtain higher throughput by pushing more jobs out of the system. Thus, we will switch the assignment of server 1 and server 2 of a selected type, gradually. Each time when we reach the threshold of switching the servers, we switch the type with smallest d_k so that we will relatively increase the service rate at station 2 as much as we can while decreasing the service rate at station 1 as little as possible. And when the number of jobs in the intermediate buffer is so large that station 1 is almost blocked, we will assign all first servers to station 2 to achieve the highest service rate at that station.

Remark 4.3.2.

1. *Since the servers always work as a team of $K > 1$, and their individual service rates are positive, the service rate at each station is always the combined service rate of a team, which equals to the sum of individual service rates of the team members times α. Thus, the value of α does not affect the choice of optimal policy.*
2. *When the servers are* generalists, *i.e., $\mu_{ij}^k = \mu_i^k \gamma_j$, then $\sigma_1 = \sigma_2 = \ldots = \sigma_K = \frac{\gamma_1}{\gamma_2}$, and $T^* = \{A_0, A_K\}$, where $A_0 = (1, 1, \ldots, 1)$ is to assign the first server of each type to station 1, and $A_K = (2, 2, \ldots, 2)$ is to assign the first server of each type to station 2. Then, for any optimal policy $\pi^* \in \Pi^*$, there exists $1 \leq s^* \leq B + 1$ such that $\pi^*(s) = A_0$ for $0 \leq s \leq s^*$, and $\pi^*(s) = A_K$ for $s^* < s \leq B + 2$.*

In the rest of this section, we illustrate our results in Theorems 4.3.2, 4.3.3 by considering two special cases, namely, when $K = 2$ and $K = 3$.

Suppose now we have four servers of two types: $\{a_1, a_2, b_1, b_2\}$, so that we need two different types of servers at each station. Then there are 4 possible team assignments: $T = \{A_0, A_1^1, A_1^2, A_2\}$, where $A_0 = (1, 1), A_1^1 = (1, 2), A_1^2 = (2, 1)$, and $A_2 = (2, 2)$. Specifically, in team A_0, $\{a_1, b_1\}$ work together at station 1; in team A_1^1, $\{a_1, b_2\}$ work together at station 1; in team A_1^2, $\{a_2, b_1\}$ work together at station 1; in team A_2, $\{a_2, b_2\}$ work together at station 1. Without loss of generality, assume that $\mu_{11}^a \geq \mu_{21}^a, \mu_{11}^b \geq \mu_{21}^b$, then Corollary 4.3.1 provides the optimal policy of this system.

Corollary 4.3.1.

(1) If $\mu_{12}^a \leq \mu_{22}^a, \mu_{12}^b \leq \mu_{22}^b$, then $T^* = \{A_0\}$.

(2) If $\mu_{12}^a \leq \mu_{22}^a, \mu_{12}^b > \mu_{22}^b$, then $T^* = \{A_0, A_1^1\}$.

(3) If $\mu_{12}^a > \mu_{22}^a, \mu_{12}^b \leq \mu_{22}^b$, then $T^* = \{A_0, A_1^2\}$.

(4) If $\mu_{12}^a > \mu_{22}^a, \mu_{12}^b > \mu_{22}^b$, denote $d_a = \frac{\mu_{11}^a - \mu_{21}^a}{\mu_{12}^a - \mu_{22}^a}, d_b = \frac{\mu_{11}^b - \mu_{21}^b}{\mu_{12}^b - \mu_{22}^b}$, then:

a. If $d_a < d_b$, then $T^* = \{A_0, A_1^2, A_2\}$;

b. If $d_a > d_b$, then $T^* = \{A_0, A_1^1, A_2\}$;

c. If $d_a = d_b$, then $T^* = \{A_0, A_2\}$.

In conclusion, when $T^* = \{A_0\}$, it is optimal to use team assignment A_0 in all states. When $T^* = \{A_0, A^k\}, A^k \in \{A_1^1, A_1^2, A_2\}$, there exists $1 \leq s^k \leq B+1$, such that if $\pi^*(s) = A_0$ for $0 \leq s \leq s^k$, and $\pi^*(s) = A^k$ for $s^k < s \leq B+2$, then π^* is an optimal policy. When $T^* = \{A_0, A_1^k, A_2\}, k = 1, 2$, there exists $1 \leq s_1^k \leq s_2^k \leq B+1$ such that, if $\pi^*(s) = A_0$ for $0 \leq s \leq s_1^k$, $\pi^*(s) = A_1^k$ for $s_1^k < s \leq s_2^k$, and $\pi^*(s) = A_2$ for $s_2^k < s \leq B+2$ then π^* is an optimal policy.

Now, consider the case $K = 3$. Then we have six servers of three types: $\{a_1, a_2, b_1, b_2, c_1, c_2\}$, and there are eight possible team assignments in T: $A_0 = (1,1,1)$, $A_1^1 = (2,1,1)$, $A_1^2 = (1,2,1)$, $A_1^3 = (1,1,2)$, $A_2^1 = (2,2,1)$, $A_2^2 = (2,1,2)$, $A_2^3 = (1,2,2)$, $A_3 = (2,2,2)$. Table 5.6 shows the detailed server assignments. Without loss of generality, assume that $\mu_{11}^a \leq \mu_{21}^a, \mu_{11}^b \leq \mu_{21}^b, \mu_{11}^c \leq \mu_{21}^c$. Also, we can re-index the type such that $d_a \leq d_b \leq d_c$. Then Corollary 4.3.2 provides the optimal policy for this system.

Table 4.2: Team Assignments for Three Types of Servers.

Team Number	Servers Work at Station 1	Servers Work at Station 2
A_0	(a_1, b_1, c_1)	(a_2, b_2, c_2)
A_1^1	(a_2, b_1, c_1)	(a_1, b_2, c_2)
A_1^2	(a_1, b_2, c_1)	(a_2, b_1, c_2)
A_1^3	(a_1, b_1, c_2)	(a_2, b_2, c_1)
A_2^1	(a_2, b_2, c_1)	(a_1, b_1, c_2)
A_2^2	(a_2, b_1, c_2)	(a_1, b_2, c_1)
A_2^3	(a_1, b_2, c_2)	(a_2, b_1, c_1)
A_3	(a_2, b_2, c_2)	(a_1, b_1, c_1)

Corollary 4.3.2.

(1) If $\mu_{12}^a \le \mu_{22}^a, \mu_{12}^b \le \mu_{22}^b, \mu_{12}^c \le \mu_{22}^c$*, then* $T^* = \{A_0\}$.

(2) If $\mu_{12}^a > \mu_{22}^a, \mu_{12}^b \le \mu_{22}^b, \mu_{12}^c \le \mu_{22}^c$*, then* $T^* = \{A_0, A_1^1\}$.

(3) If $\mu_{12}^a \le \mu_{22}^a, \mu_{12}^b > \mu_{22}^b, \mu_{12}^c \le \mu_{22}^c$*, then* $T^* = \{A_0, A_1^2\}$.

(4) If $\mu_{12}^a \le \mu_{22}^a, \mu_{12}^b \le \mu_{22}^b, \mu_{12}^c > \mu_{22}^c$*, then* $T^* = \{A_0, A_1^3\}$.

(5) If $\mu_{12}^a > \mu_{22}^a, \mu_{12}^b > \mu_{22}^b, \mu_{12}^c \le \mu_{22}^c$*, then if* $d_a < d_b$*,* $T^* = \{A_0, A_1^1, A_2^1\}$*; if* $d_a = d_b$*,* $T^* = \{A_0, A_2^1\}$.

(6) If $\mu_{12}^a > \mu_{22}^a, \mu_{12}^b \le \mu_{22}^b, \mu_{12}^c > \mu_{22}^c$*, then if* $d_a < d_c$*,* $T^* = \{A_0, A_1^1, A_2^2\}$*; if* $d_a = d_c$*,* $T^* = \{A_0, A_2^2\}$.

(7) If $\mu_{12}^a \le \mu_{22}^a, \mu_{12}^b > \mu_{22}^b, \mu_{12}^c > \mu_{22}^c$*, then if* $d_b < d_c$*,* $T^* = \{A_0, A_1^2, A_2^3\}$*; if* $d_b = d_c$*,* $T^* = \{A_0, A_2^3\}$.

(8) If $\mu_{12}^a > \mu_{22}^a, \mu_{12}^b > \mu_{22}^b, \mu_{12}^c > \mu_{22}^c$*, then:*

a. If $d_a = d_b = d_c$*, then* $T^* = \{A_0, A_3\}$*;*

b. If $d_a = d_b < d_c$*, then* $T^* = \{A_0, A_2^1, A_3\}$*;*

c. If $d_a < d_b = d_c$, then $T^* = \{A_0, A_1^1, A_3\}$.

d. If $d_a < d_b < d_c$, then $T^* = \{A_0, A_1^1, A_2^1, A_3\}$

When $T^* = \{A_0\}$, it is optimal to use team assignment A_0 in any recurrent state. When $T^* = \{A_0, A^k\}, A^k \in T \setminus \{A_0\}$, there exists $1 \leq s^k \leq B + 1$, such that if $\pi^*(s) = A_0$ for $0 \leq s \leq s^k$, and $\pi^*(s) = A^k$ for $s^k < s \leq B + 2$, then π^* is an optimal policy. When $T^* = \{A_0, A^i, A^j\}, A^i, A^j \in T \setminus \{A_0\}$, there exists $1 \leq s_{ij}^1 \leq s_{ij}^2 \leq B + 1$ such that, if $\pi^*(s) = A_0$ for $0 \leq s \leq s_{ij}^1$, $\pi^*(s) = A^i$ for $s_{ij}^1 < s \leq s_{ij}^2$, and $\pi^*(s) = A^j$ for $s_{ij}^2 < s \leq B + 2$, then π^* is an optimal policy. When $T^* = \{A_0, A_1^i, A_2^j, A_3\}, i, j \in \{1, 2, 3\}$, there exists $1 \leq \hat{s}_{ij}^1 \leq \hat{s}_{ij}^2 \leq \hat{s}_{ij}^3 \leq B + 1$ such that, if $\pi^*(s) = A_0$ for $0 \leq s \leq \hat{s}_{ij}^1$, $\pi^*(s) = A_1^i$ for $\hat{s}_{ij}^1 < s \leq \hat{s}_{ij}^2$, $\pi^*(s) = A_2^j$ for $\hat{s}_{ij}^2 < s \leq \hat{s}_{ij}^3$, and $\pi^*(s) = A_3$ for $\hat{s}_{ij}^3 < s \leq B + 2$ then π^* is an optimal policy.

4.4 Longer Lines

In this section, we investigate the systems with teams of specialized servers for longer lines. In particular, we focus on the case when the servers are generalists. Consider $N \geq 3$ tandem stations with finite intermediate buffers and K types of servers. For simplicity, let $\mu_i^k \gamma_j$ be the service rate of ith server of type k working at station j, where $i, j \in \{1, \ldots, N\}, k \in \{1, \ldots, K\}$. Indeed, μ_i^k could be regarded as the specialty of ith server of type k while γ_j might represent the difficulty of the task attached to station j.

First, we generalize the definition of dominated team assignment for longer lines.

Definition 4.4.1. *(Dominated Team Assignment) Team assignment $i \in T$ is a dominated team assignment, if there exists $p \in (0, 1)$ and assignments j, l such that $p\gamma_{j,k} + (1-p)\gamma_{l,k} > \gamma_{i,k}$ for $k = 1, \ldots, M$.*

Note that, if such p exists, then we can obtain higher throughput by using team assignment j, l instead of i. Thus i is dominated and any policy uses i is not optimal.

By our experience from previous research, the optimal policies for larger systems with general service rates would be much more complex than the two stations case. Therefore, for the rest of this section, we will propose heuristic policies for longer lines and evaluate their performance via simulation.

4.4.1 Heuristics for Longer Lines

In this section, we present the heuristic policy for longer lines with teams of specialized servers where the servers are generalists. To get more insights for our design of the heuristic policy, we first investigate the optimal policies for some special cases. Then, we will propose the heuristic policy for longer lines by learning from the structures of the optimal policies of these examples.

Consider the most basic case with $K = 2$, $N = 3$, and zero intermediate buffers between the stations. Denote the servers as $\{a_1, a_2, a_3, b_1, b_2, b_3\}$. Let $s = (s^1, s^2)$ be the state of the system, where s^j is the number of jobs that have been processed at stations preceding and including station j, but not at stations succeeding station j, for $j \in \{1, 2\}$. Then, the state space is

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

The following examples show the impact of different task difficulties among stations to the optimal policy of the system. In particular, Examples 4.4.1, 4.4.2, and 4.4.3 provide the optimal policies for generalists with task difficulties of stations that are balanced, biased towards upstream, and biased towards downstream, respectively.

Example 4.4.1. *If $\mu_1^a = 3, \mu_2^a = 2, \mu_3^a = 1, \mu_1^b = 3, \mu_2^b = 2, \mu_3^b = 1, \gamma_1 = \gamma_2 = \gamma_3 = 1$, then the optimal policy is as follows:*

1. *When $s = (0,0)$, assign (a_1, b_1) to station 1;*
2. *When $s = (2,0)$, assign (a_1, b_1) to station 2;*

3. *When* $s = (1, 2)$*, assign* (a_1, b_1) *to station 3;*

4. *When* $s = (1, 0)$*, assign* (a_2, b_2) *to station 1 and* (a_1, b_1) *to station 2;*

5. *When* $s = (0, 1)$*, assign* (a_1, b_1) *to station 1 and* (a_2, b_2) *to station 3;*

6. *When* $s = (2, 1)$*, the assignment is arbitrary as long as we assign* a_1, a_2, b_1, b_2 *to stations 2,3 under the type constraint;*

7. *When* $s = (0, 2)$*, assign* (a_2, b_2) *to station 1 and* (a_1, b_1) *to station 3;*

8. *When* $s = (1, 1)$*, assign* (a_3, b_3) *to station 1, and the rest of the assignments are arbitrary.*

Example 4.4.2. *If* $\mu_1^a = 3, \mu_2^a = 2, \mu_3^a = 1, \mu_1^b = 3, \mu_2^b = 2, \mu_3^b = 1, \gamma_1 = 3, \gamma_2 = 2, \gamma_3 = 1,$ *then the optimal policy is as follows:*

1. *When* $s = (0, 0)$*, assign* (a_1, b_1) *to station 1;*

2. *When* $s = (2, 0)$*, assign* (a_1, b_1) *to station 2;*

3. *When* $s = (1, 2)$*, assign* (a_1, b_1) *to station 3;*

4. *When* $s = (1, 0)$*, assign* (a_2, b_2) *to station 1 and* (a_1, b_1) *to station 2;*

5. *When* $s = (0, 1)$*, assign* (a_1, b_1) *to station 1 and* (a_2, b_2) *to station 3;*

6. *When* $s = (2, 1)$*, assign* (a_2, b_2) *to station 2 and* (a_1, b_1) *to station 3;*

7. *When* $s = (0, 2)$*, assign* (a_2, b_2) *to station 1 and* (a_1, b_1) *to station 3;*

8. *When* $s = (1, 1)$*, assign* (a_3, b_3) *to station 1,* (a_2, b_2) *to station 2, and* (a_1, b_1) *to station 3.*

Example 4.4.3. *If* $\mu_1^a = 3, \mu_2^a = 2, \mu_3^a = 1, \mu_1^b = 3, \mu_2^b = 2, \mu_3^b = 1, \gamma_1 = 1, \gamma_2 = 2, \gamma_3 = 3,$ *then the optimal policy is as follows:*

1. When $s = (0,0)$, assign (a_1, b_1) to station 1;

2. When $s = (2,0)$, assign (a_1, b_1) to station 2;

3. When $s = (1,2)$, assign (a_1, b_1) to station 3;

4. When $s = (1,0)$, assign (a_1, b_1) to station 1 and (a_2, b_2) to station 2;

5. When $s = (0,1)$, assign (a_1, b_1) to station 1 and (a_2, b_2) to station 3;

6. When $s = (2,1)$, assign (a_1, b_1) to station 2 and (a_2, b_2) to station 3;

7. When $s = (0,2)$, assign (a_1, b_1) to station 1 and (a_2, b_2) to station 3;

8. When $s = (1,1)$, assign (a_1, b_1) to station 1, (a_2, b_2) to station 2, and (a_3, b_3) to station 3.

From Examples 4.4.1, 4.4.2, and 4.4.3, we can observe that:

1. We always use the following three teams of servers: (a_1, b_1), (a_2, b_2), and (a_3, b_3) in all optimal policies of our numerical results. This observation is consistent with our result in Remark 4.3.2.2 for two-station systems.

2. When one of the three stations is starved or blocked, we will always let the slowest team (a_3, b_3) idle; when two of the three stations are starved or blocked, we will let the two slower teams, (a_2, b_2) and (a_3, b_3), be idle. That is, when some of the stations are starved or blocked, we will always let the slower teams be idle.

3. When $s = (0,1)$, we will always assign the fastest team (i.e., (a_1, b_1)) to station 1. Intuitively, we want the proportion of time that all three stations are working to be as large as possible to avoid starving or blocking, i.e., we want to push the system to $s = (1,1)$. Therefore, we assign the fastest team to station 1 to push more jobs into the system and avoid starving at station 2 when $s = (0,1)$.

4. For any state $s \in S \setminus \{(0,1)\}$, we always assign the fastest remaining team to the working and unassigned station with the highest difficulty. Intuitively, in order to maximize the long-run average throughput of the tandem system, we want to avoid having a bottleneck station or at least increase the service rate of the bottleneck station as much as possible. Therefore, we assign the faster team to the slower station to balance the service rate at each station. Moreover, if we assign (a_1, b_1) to station 3 and (a_2, b_2) to station 1 when $s = (0,1)$, and assign teams according to the optimal policy for the other states for Example 4.4.2, we will have a long-run average throughput of 4.6974, which is 0.2% less than the optimal throughput 4.7085. Thus, we can achieve a near-optimal throughput by always assigning the fastest team to the slowest station.

Combining our observations above with our previous results, we propose the following heuristic server assignment policies for the general case:

First, label the servers of type k such that $\mu_1^k \geq \mu_2^k \geq \ldots \geq \mu_N^k$ for $k = 1, \ldots, K$.

1. We will use N permanent teams that are formed based on their ability. We will use the following teams: the best server of each type, the second best server of each type,..., the worst server of each type. That is, the ith team consists of the ith server of each type for $i = 1, \ldots, N$.

2. When some of the stations are starved or blocked, we will always let the servers with lower service rates idle. That is, if n_0 stations are working at time t, we will let teams $1, \ldots, n_0$ work and $n_0 + 1, \ldots, N$ idle.

In order to assign these teams to the working stations, our basic idea is to balance the system by assigning the fastest remaining team to the slowest remaining station. Also, to fully utilize our teams, we want to avoid stations from blocking and starving. Thus, based on different assignments when there are blocked or starved stations in the system, we consider two different plans.

In the first plan, we prioritize eliminating blocking over starving. Specifically, we assign the teams according to the following rules in the order given below:

1. If there are blocked stations in the line, then assign the fastest teams to the stations immediately following the blocked stations (starting from the end of the line).

2. Next, starting from the end of the line, if the buffer size after some working station is greater than or equal to 2, and the buffer is already full, we regard this station as almost blocked station. If the station immediately following this almost blocked station is unassigned, we assign the fastest remaining team to this station.

3. Next, starting from the beginning of the line, if there is some starved station in the line, and the station immediately preceding the starved station is unassigned, we assign the fastest remaining team to this station.

4. Next, starting from the beginning of the line, if the buffer size before some working station is greater and equal to 2, and the buffer is empty, we regard this station as almost starved station. And if the station immediately preceding this almost starved station is unassigned, we assign the fastest remaining team to this station.

5. For all stations that do not have assigned servers, assign the fastest remaining team to the slowest remaining station. In case of ties, that is, if $\exists i, j$ s.t. $\gamma_i = \gamma_j$, assign the faster team closer to the end of line as in ”bucket brigades” (see Bartholdi and Eisenstein [18]).

Note that, if some station is both blocked and starved, we will mark the station as blocked station; if some station is both almost blocked and almost starved, we will mark the station as almost blocked station.

In the second plan, we prioritize eliminating starving over blocking. More specifically, we will switch the order of the rules in the first plan as $3 \to 4 \to 1 \to 2 \to 5$. That is, we assign the fastest teams to deal with the starved and the almost starved stations first, and then we assign the remaining fastest teams to deal with the blocked and the almost blocked stations.

4.4.2 Numerical Results

To investigate the validity of our heuristic policies and compare two different plans, in this section, we provide numerical results. We compute the optimality gap (in percentage) between our heuristics and the optimal results (i.e. the deviation of our heuristic from the optimal policy with respect to the long-run average throughput of the system). In addition, we compare our results to the optimality gap between the optimal dynamic policy and optimal static policy. The optimal static policy has been discussed in Section 4.2.3, which is the assignment that maximizes the long-run average throughput of the system when the servers are static and server assignments are permanent.

First, we present numerical results of the system with two types of servers $\{a, b\}$, different number of stations, different buffers, and randomly generated service rates to investigate the effectiveness of our heuristics. All tables display 95% confidence intervals, and the numbers are in percentage.

In Table 4.3, we provide the numerical results for three stations case. Specifically, to model systems with small buffers, medium buffers, and large buffers, we consider systems with common buffers for 1,5 or 10, and systems with buffers $B_1 = 1, B_2 = 10$, and $B_1 = 10, B_2 = 1$. We have $\mu_i^a, \mu_i^b, \gamma_j$ drawn independently from a uniform distribution with range [0.5,2.5] for all $i, j \in \{1, 2, 3\}$. We performed 5000 iterations for each pair of buffer sizes, and the results are shown in Table 4.3. Note that, for three stations systems, the two different plans are the same. More specifically, consider the following cases that include all the situations that might cause differences between these two plans in three stations line: (i) If some station is starved, and another station is blocked, then there is only one station working under the circumstance, and the assignment is obviously the same for both plans; (ii) if station 3 is starved or almost starved, and station 1 is blocked or almost blocked, we will assign the fastest team to station 2 in both plans. Thus, the two heuristic policies are the same in three stations case, and we only provide one heuristic results for both plans in Table 4.3.

Table 4.3: Numerical Results for Three Stations.

Buffer Sizes		% Optimality Gap	
B_1	B_2	Opt. Static	Heuristic
0	0	14.94 ± 0.18	0.47 ± 0.02
5	0	11.15 ± 0.18	1.88 ± 0.05
0	5	11.21 ± 0.18	1.49 ± 0.03
5	5	7.23 ± 0.16	2.09 ± 0.04
10	5	6.81 ± 0.15	2.36 ± 0.05
5	10	6.54 ± 0.15	2.31 ± 0.05
10	10	5.69 ± 0.14	2.48 ± 0.05
Average		9.14 ± 0.07	1.87 ± 0.02

Table 4.4 shows the numerical results for four stations systems. We consider systems with buffers 0,2, and 4. Again, $\mu_i^a, \mu_i^b, \gamma_j$ are drawn independently from a uniform distribution with range [0.5,2.5] for all $i, j \in \{1, 2, 3, 4\}$. We performed 1000 iterations for each pair of buffer sizes.

Table 4.4: Numerical Results for Four Stations.

Buffer Sizes			% Optimality Gap		
B_1	B_2	B_3	Opt. Static	Heuristic 1	Heuristic 2
0	0	0	17.15 ± 0.40	1.47 ± 0.05	1.52 ± 0.06
2	0	0	14.99 ± 0.41	2.51 ± 0.08	2.33 ± 0.07
0	2	0	15.67 ± 0.40	2.66 ± 0.07	2.85 ± 0.08
0	0	2	15.13 ± 0.41	1.42 ± 0.05	1.49 ± 0.07
2	2	2	11.57 ± 0.39	2.69 ± 0.09	2.70 ± 0.09
0	4	4	11.26 ± 0.40	2.72 ± 0.07	3.01 ± 0.10
4	0	4	11.18 ± 0.40	2.36 ± 0.12	2.35 ± 0.12
4	4	0	12.06 ± 0.41	3.86 ± 0.12	3.71 ± 0.10
4	4	4	8.60 ± 0.38	3.05 ± 0.11	3.12 ± 0.11
Average			13.11 ± 0.14	2.52 ± 0.03	2.56 ± 0.03

Table 4.5 shows the numerical results for five stations systems. We consider systems with buffers 0,1,and 2. Again, $\mu_i^a, \mu_i^b, \gamma_j$ are drawn independently from a uniform distribution with range [0.5,2.5] for all $i, j \in \{1, \ldots, 5\}$. We performed 200 iterations for each combination of buffer sizes.

Table 4.5: Numerical Results for Five Stations.

Buffer Sizes				% Optimality Gap		
B_1	B_2	B_3	B_4	Opt. Static	Heuristic 1	Heuristic 2
0	0	0	0	19.36 ± 0.85	2.40 ± 0.15	2.71 ± 0.18
1	0	0	0	18.30 ± 0.86	3.02 ± 0.21	2.91 ± 0.22
0	1	0	0	17.71 ± 0.86	3.11 ± 0.19	3.44 ± 0.19
0	0	1	0	18.37 ± 0.85	2.92 ± 0.20	3.38 ± 0.21
0	0	0	1	18.65 ± 0.86	2.06 ± 0.19	2.23 ± 0.20
1	1	1	1	14.30 ± 0.89	3.22 ± 0.18	3.28 ± 0.18
Average				17.66 ± 0.40	2.82 ± 0.08	3.05 ± 0.09

From the results of Table 4.3-4.5, we can derive the following conclusions:

1. In all situations, the performance of our heuristic policies are significantly better than the optimal static policy. Hence, our heuristic policies improve the throughput of the system by assigning teams dynamically.

2. The optimality gap between our heuristic policies and optimal dynamic policy is always less than 4%, which means that our heuristic policies yield near-optimal performance.

3. Heuristic plan 2 performs better than heuristic plan 1 when there are more buffers at the head of the line, but for all the other cases, for example, when the buffers are balanced between stations or there are more buffers distributed towards the end of the line, heuristic plan 1 works better.

4. We don't have a clear pattern of the relationship between the total buffers and the optimality gaps. Our heuristics perform better for more balanced systems than unbalanced ones even if the total buffer size of the balanced system is larger than that of the unbalanced system.

4.5 Conclusions

We have studied Markovian queueing systems with N tandem stations, finite intermediate buffers, and flexible servers with collaboration. For such queueing systems, we established sufficient criteria for eliminating inferior teams, and then identified the optimal team assignment policy among the remaining teams (i.e. optimal assignment set) for the two-station case. We showed that if we label the remaining teams according to their service rate at station 1 from high to low, then the optimal policy has monotone thresholds so that it uses the first team assignment (the team with highest service rate at station 1) when the buffer is empty, and transit to the second, third,. . . , last team assignment as the number of jobs in the buffer increases. Then, we applied our optimal policy to two special cases, namely proportional team service rates and teams of specialized servers. For teams of specialized servers, when the servers are generalists, the optimal policy set contains N permanent teams that are formed based on their ability. In particular, the N teams are the best server of each type, the second best server of each type,. . . , the worst server of each type. Based on this result for two stations systems and examples for three stations systems, we proposed heuristic policy with teams of specialized servers where the servers are generalists for longer lines, and obtained numerical results that suggested that our heuristics yield near-optimal performance for systems with more than two stations.

CHAPTER 5
OPTIMAL CONTROL OF QUEUEING SYSTEMS WITH DEFECTS

5.1 Introduction

In this chapter, we consider a Markovian system of N tandem stations, finite buffers between the stations, and N servers. Assume there is an infinite supply of raw materials in front of the first station and infinite storage space after the last station. Since in many systems, collaboration is infeasible or undesirable, we focus on non-collaborative servers that are not able to work on the same job and assume that there can be at most one job undergoing processing at each station. For example, limitations of workspace and tools can prevent multiple servers from working simultaneously at the same station. However, the servers are flexible, which means that they are cross-trained and allowed to switch between stations with negligible travel time. When a job is being processed at a station, it might incur a defect and be wasted. Let $0 \leq B_j < \infty, j \in \{1, 2, \ldots, N-1\}$, be the buffer size between station j and $j+1$, and $\mu_{i,j}, p_{i,j}, i, j \in \{1, 2, \ldots, N\}$, be the service rate and defect probability of server i working at station j, respectively. Assume that $\sum_{j=1}^{N} \mu_{i,j}(1 - p_{i,j}) > 0$ for $i \in \{1, 2, \ldots, N\}$ (otherwise the problem reduces to having $N-1$ servers) and $\sum_{i=1}^{N} \mu_{i,j}(1-p_{i,j}) > 0$ for $j \in \{1, 2, \ldots, N\}$ (otherwise the throughput is zero under any policy). Moreover, assume that if a job is processed by multiple servers in turn at the same station, its defect probability is determined by the (last) server who finishes processing this job at the station. For simplicity, let $\hat{\mu}_{i,j} = \mu_{i,j}(1 - p_{i,j})$ be the successful service rate for $i, j \in \{1, 2, \ldots, N\}$. Then $\hat{\mu}_{i,j}$ is the rate of obtaining a service completion with no defects by server i at station j. The service requirement for each job is independently and exponentially distributed at each station, and without loss of generality, we assume the mean service requirement of each job at each station is one. The system operates under manufacturing blocking. Our objective is to determine the dynamic server allocation policy that maximizes the long-run average throughput.

This system without defects (i.e., when $p_{i,j} = 0$ for $i, j \in \{1, 2, \dots, N\}$) has been analyzed by Işık, Andradóttir, and Ayhan [31]. For systems with two stations and two servers, they provided the server allocation policy that maximizes the long-run average throughput. For systems of arbitrary size, they presented optimal policy for the special case when each server is specialized at different station and proposed heuristic policies that appear to be near-optimal for the general case. However, in practice, it is common to have positive defect probabilities when servers process jobs. Thus, we include this factor into consideration. We will see that the introduction of defect probabilities changes the structure of the optimal policy. To the best of our knowledge, this is the first paper that considers the dynamic scheduling of servers when they are flexible and error-prone.

In the presence of positive defect probabilities, it is not only important to finish jobs quickly (by assigning a faster server), but to also have a successful service completion (by assigning a more reliable server). Thus, if the fastest server and the most reliable server at a station are not the same server, there is a trade-off between speed and more reliability. Hence, the optimal policy for systems with defects is more complex than the the optimal policy for systems without defects. In this chapter, we completely characterize the optimal policy for two stations and two servers. Specifically, if we label the servers such that $\hat{\mu}_{11} \geq \hat{\mu}_{21}$ (which is without loss of generality), then we show that the optimal policy takes two forms depending on whether (1) $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$ or (2) $\hat{\mu}_{11}\mu_{12} < \hat{\mu}_{21}\mu_{22}$. When $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$, the optimal policy has a single threshold and it is best to assign server i to station i for $i = 1, 2$ if the number of jobs in the intermediate buffer is less than that threshold; and otherwise, we will assign server i to station $3-i$ for $i = 1, 2$. However, when $\hat{\mu}_{11}\mu_{12} < \hat{\mu}_{21}\mu_{22}$, the optimal policy has two thresholds and server i should be assigned to station i for $i = 1, 2$ if the buffer is empty (the smaller threshold) or if the number of jobs in the buffer exceeds the larger threshold; otherwise, we will assign server i to station $3 - i$

for $i = 1, 2$. In addition to identifying the optimal policy for two stations and two servers, we propose several heuristic policies for larger systems and provide numerical results that suggest that our heuristics appear to be near-optimal.

The outline of this chapter is as follows. In Section 5.2, we provide a description of our problem and the notation we use throughout this chapter. In Section 5.3, we present preliminary results on the properties of the optimal policy. In Section 5.4, we identify the policy that maximizes the long-run average throughput for two stations and two servers. In Section 5.5, we provide heuristic policies for larger systems and provide numerical results that suggest that the performance of our heuristics is near-optimal. In Section 5.6, we summarize our findings and conclude the chapter. Finally, the proofs of some of our results are provided in Appendix B.

5.2 Formulation

In this section, we present a detailed description of our model. Let Π be the set of server assignment policies under consideration. For all $\pi \in \Pi$, let $D^\pi(t)$ be the number of successful departures (with no defects) from the last station under policy π by time t, and let

$$T^\pi = \limsup_{t \to \infty} \frac{\mathbb{E}[D^\pi(t)]}{t} \tag{5.1}$$

be the long-run average throughput corresponding to server allocation policy π. We want to solve the optimization problem

$$\max_{\pi \in \Pi} T^\pi. \tag{5.2}$$

For all $\pi \in \Pi$ and $t \geq 0$, let $X^\pi(t) = (X_1(t), \ldots, X_{N-1}(t)), X_j(t) \in \{0, 1, \ldots, B_j + 2\}$, where $X_j(t)$ is the number of jobs in the system that have completed service with no defects at station j but have not yet completed service at station $j+1$ at time t under policy π. Denote the set of all possible states as S. From now on, we assume that the class Π of server assignment policies under consideration consists of all Markovian stationary deterministic policies corresponding to the state space S of the stochastic processes $\{X^\pi(t)\}$. Let $a_{\sigma_1\sigma_2\ldots\sigma_N}$ be an action that indicates the allocation of the servers, where $\sigma_i = 0$ if server i is idled, and $\sigma_i = j$ if server i is assigned to station j, for $i, j \in \{1, 2, \ldots, N\}$. Let

$$A_x = A = \{a_{\sigma_1\sigma_2\ldots\sigma_N} : \sigma_i \in \{0, 1, \ldots, N\}, \sigma_i \neq \sigma_k \text{ for } i \neq k \text{ with } \sigma_i\sigma_k > 0\}$$

denote the set of available actions in state $x \in S$.

It is clear that $\{X^\pi(t) : t \geq 0\}$ is a continuous-time Markov chain. Let $\{q^\pi(x, x')\}$ be the transition rates of $\{X^\pi(t)\}$, and let $\mu_j^\pi(x), p_j^\pi(x)$ be the service rate and defect probability at station $j \in \{1, 2, \ldots, N\}$ in state $x \in S$ under policy $\pi \in \Pi$, respectively. Then, for $x \in S$,

$$q^\pi(x, x') = \begin{cases} \mu_1^\pi(x)(1 - p_1^\pi(x)) & \text{if } x' = x + e_1, x' \in S \\ \mu_j^\pi(x)(1 - p_j^\pi(x)) & \text{if } x' = x - e_{j-1} + e_j, x' \in S, 2 \leq j \leq N-1, \\ \mu_j^\pi(x)p_j^\pi(x) & \text{if } x' = x - e_{j-1}, x - e_{j-1} + e_j \in S, 2 \leq j \leq N-1, \\ \mu_N^\pi(x) & \text{if } x' = x - e_{N-1}, x' \in S \\ 0 & \text{otherwise,} \end{cases} \tag{5.3}$$

where $\{e_j : j = 1, \ldots, N-1\}$ is the standard basis for $(N-1)$-dimensional space. There exists a finite uniformization constant $q \leq \sum_i \max_j \mu_{i,j} < \infty$ such that $\{q^\pi(x, x')\}$ satisfy $\sum_{x' \in S, x' \neq x} q^\pi(x, x') \leq q$ for all $x \in S, \pi \in \Pi$. Thus, $\{X^\pi(t)\}$ is uniformizable. Let $\{Y^\pi(k)\}$ be the corresponding discrete-time Markov chain, so that $\{Y^\pi(k)\}$ has state space S and transition probabilities $p^\pi(x, x') = q^\pi(x, x')/q$ if $x' \neq x$ and $p^\pi(x, x) = 1 - \sum_{x' \in S, x' \neq x} q^\pi(x, x')/q$ for all $x \in S$. Using a similar argument as in [7], we can show that the original optimization problem in (5.2) can be translated into an equivalent discrete-time Markov decision problem. Specifically, for $x = (s_1, \dots, s_{N-1}) \in S$,

$$R^{\pi}(x) = \begin{cases} \mu_N^{\pi}(x)(1 - p_N^{\pi}(x)) & \text{if } s_{N-1} > 0, \\ 0 & \text{otherwise,} \end{cases} \tag{5.4}$$

is the departure rate (with no defects) from state x under policy π for all $x \in S$, and $\pi \in \Pi$. Then the optimization problem (5.2) has the same solution as the Markovian decision problem

$$\max_{\pi \in \Pi} \lim_{K \to \infty} E\Big\{\frac{1}{K} \sum_{k=1}^{K} R^{\pi}(Y^{\pi}(k-1))\Big\}. \tag{5.5}$$

Thus, maximizing the long-run average throughput of the original queueing system is equivalent to maximize the long-run average successful departure rate for the associated embedded discrete-time Markov chain.

5.3 Preliminary Results

In this section, we discuss the structure of the optimal policy for systems with N stations and N servers, where N is arbitrary.

We have discussed the trade-off between service rate and defect probability in Section 5.1. However, if there exists a server that is both more reliable and faster than all other servers at each station, and these servers are distinct, then we identify the optimal server allocation as follows:

Theorem 5.3.1. *Suppose there exists a permutation of the servers $i_1, \dots, i_N$ such that $\mu_{i_j j} = \max_{1 \le i \le N} \mu_{ij}, p_{i_j j} = \min_{1 \le i \le N} p_{ij}$ for all $j \in \{1, 2, \dots, N\}$. Then the policy that always assigns server i_j to station j for $j \in \{1, 2, \dots, N\}$ is optimal.*

Without loss of generality, assume that $i_j = j$ for $j \in \{1, 2, \dots, N\}$. Let π^* be the policy that always assigns server j to station j for $j \in \{1, 2, \dots, N\}$. For any policy $\pi \in \Pi$ and $\pi \neq \pi^*$, we consider an artificial system under policy π but with servers of modified service rates. Specifically, in this artificial system, at any state $s \in S$, the service rate at station $j \in \{1, \ldots, N\}$ is altered to μ_{jj} (the same as under π^* in the real system), while the defect probability of the server at station j remains the same as under π in the real system. Denote $\tilde{T}^{\pi}$ as the long-run average throughput of this artificial system under π. We will prove Theorem 5.3.1 in two steps: Lemma 5.3.1 will show that the throughput of the artificial system under π with modified service rates is no worse than the throughput of the real system under π, and Lemma 5.3.2 will show that the throughput of the real system under π^* is no worse than the artificial system under π. Then, π^* is no worse than π for any $\pi \in \Pi$ in the real system, and π^* is optimal. The proofs of Lemma 5.3.1 and Lemma 5.3.2 are provided in the Appendix.

Lemma 5.3.1. $\tilde{T}^{\pi} \geq T^{\pi}$.

Lemma 5.3.2. $T^{\pi^*} \geq \tilde{T}^{\pi}$.

Remark 5.3.1. *Lemma 5.3.1 holds as long as the service requirements at each station are independently and identically distributed. Thus, if the defect probability only depends on the station, i.e., $p_{i,j} = p_j$ for all $i, j \in \{1, \ldots, N\}$. Then, Theorem 5.3.1 holds for a more general system where the service requirements are independent and identically distributed random variables (but not necessarily exponentially distributed).*

For the result in Theorem 5.3.1 to hold in sample path sense, we need to prove that the ith successful departure (with no defects) from the artificial system under policy π with modified service rates should be no earlier than under policy π^* for all $i \geq 1$ and $\pi \in \Pi$. However, the following example demonstrates that this inequality may fail if the service requirements are not exponential.

Example 5.3.1. *Consider a system with $N = 2$ and $B_1 = 0$. Suppose the two servers are identical except that server 1 is more reliable than server 2 at station 1 (i.e., $p_{11} < p_{21}$). Let policy π_1 be the policy that always assigns server i to station i for $i = 1, 2$; and π_2 be the policy that assigns server i to station $3 - i$ for $i = 1, 2$. Then, if Theorem 5.3.1 holds* *in sample path sense, all successful departures from the system under policy π_1 should be earlier than under policy π_2.*

Consider two sample paths ω_1, ω_2, where we use policy π_i in sample path ω_i for $i = 1, 2$. Thus, ω_1 has lower defect probability at station 1 than ω_2. Let $S_j(i)$ be the service time of the ith job that arriving to station j, for $j = 1, 2, i \geq 1$. Suppose that both sample paths share the same service times as follows: $S_1(i) = 1$ for all $i \geq 1$, $S_2(1) = 3, S_2(i) = 1$ for all $i \geq 2$. Let $I_j^k(i) \in \{0, 1\}$ be the status of the ith job at station j upon its service completion under policy π_k for $i \geq 1, j, k = 1, 2$, where 0 and 1 refer to defective and successful service completion, respectively. Then, $p_{11} < p_{21}$ implies that $I_j^1(i) \geq I_j^2(i)$ for $j = 1, 2, i \geq 1$. Suppose $(I_1^1(i) : 1 \leq i \leq 5) = (1, 1, 0, 1, 1)$, $(I_1^2(i) : 1 \leq i \leq 5) = (1, 0, 0, 1, 1)$, $(I_2^1(i) : 1 \leq i \leq 3) = (I_2^2(i) : 1 \leq i \leq 3) = (0, 0, 1)$. Then, Table 5.1 shows the comparison of the time flow of the first five departures from station 1 and first three departures from station 2 in sample paths ω_1, ω_2. We number the jobs by their order of entering station 1 in both sample paths. From Table 5.1, we observe that the time of the first successful departure from the system under policy π_1 is at time 7, while the first successful departure from the system under policy π_2 is at 6. Thus, Theorem 5.3.1 does not hold in sample path sense.

Table 5.1: Departure times from station 1 and 2 in sample paths ω_1, ω_2 (successful departures are in bold).

	departure time in ω_1 from		departure time in ω_2 from	
job #	station 1	station 2	station 1	station 2
1	**1**	4	**1**	4
2	**4**	5	2	X
3	5	X	3	X
4	**6**	**7**	**4**	5
5	**7**	\	**5**	**6**

X-the job is defective and wasted at station 1.

\- not discussed in this example.

Note that, since there are infinitely many jobs in front of the first station, having a defective service completion at station 1 would not alter the state of the system. Therefore, if we assign server i to station 1 with successful service rate $\hat{\mu}_{i,1}$, it is equivalent to having a 'dummy' server with service rate $\hat{\mu}_{i,1}$ and zero defect probability at station 1. And using this ideal, we can obtain the following two propositions of the properties of server assignment at station 1. First, the following proposition shows that when the system is empty, the best server allocation is to assign the server with largest successful service rate to station 1.

Proposition 5.3.1. *When the system is empty, i.e.,* $s = (0, \ldots, 0)$*, it is optimal to assign the server such that* $\hat{\mu}_{i,1}$ *is maximized to station 1.*

Proof. For $\pi \in \Pi$, $\{X^\pi(t)\}$ is a semi-Markov process. And each time it enters a state $s \in S$, it remains there for a random amount of time with rate $q^\pi(s) = \sum_{s' \in S, s' \neq s} q^\pi(s, s')$. If we let X_n^π denote the state of the process after the nth transition, then $\{X_n^\pi, n \geq 0\}$ is a Markov chain with transition probability $v^\pi(s, s') = \frac{q^\pi(s,s')}{q^\pi(s)}$, for $s, s' \in S, s' \neq s$. Let $\eta^\pi(s)$ be the stationary probability of $s \in S$ for $\{X_n^\pi\}$. That is, $\eta^\pi(s)$ is the unique nonnegative solution of

$$\sum_{s \in S} \eta^\pi(s) = 1,$$
$$\eta^\pi(s) = \sum_{s' \in S} \eta^\pi(s') v^\pi(s', s), \forall s \in S.$$

Then, the long-run proportion of time that process $\{X^\pi(t)\}$ spends in state $s \in S$ is:

$$\tau^\pi(s) = \frac{\eta^\pi(s)/q^\pi(s)}{\sum_{s' \in S} \eta^\pi(s')/q^\pi(s')}.$$

When the system is empty, i.e., $s_0 = (0, \ldots, 0)$, $v^\pi(s_0, s') = 1$ if $s' = (1, 0, \ldots, 0)$; and $v^\pi(s_0, s') = 0$, otherwise, for $\forall \pi \in \Pi$. Thus, the change of assignment in state s_0 does not impact the value of $\tau^\pi(s)$ for any $s \in S$. Moreover,

$$\tau^\pi(s_0) = \frac{\eta(s_0)/q^\pi(s_0)}{\eta(s_0)/q^\pi(s_0) + \sum_{s' \in S, s' \neq s_0} \eta^\pi(s')/q^\pi(s')},$$
$$\tau^\pi(s) = \frac{\eta^\pi(s)/q^\pi(s)}{\eta(s_0)/q^\pi(s_0) + \sum_{s' \in S, s' \neq s_0} \eta^\pi(s')/q^\pi(s')}, s \in S,$$

where $q^\pi(s_0)$ is the successful service rate at station 1 in state s_0 under policy $\pi \in \Pi$.

Thus, by increasing $q^\pi(s_0)$, we can reduce the time that the process is in state s_0 (i.e., $\tau^\pi(s_0)$) and increase the time the process is in all the other states without changing anything else. Since in state $s_0 = (0, \ldots, 0)$, there is no departure from the system, we increase the long-run average throughput of the system by increasing the time that the process spends in the states where there are departures without changing the departure rates. Since $\max_{\pi \in \Pi} q^\pi(s_0) = \max_{1 \le i \le N} \hat{\mu}_{i,1}$, it is optimal to assign the server that maximizes $\hat{\mu}_{i,1}$ when in state $s_0 = (0, \ldots, 0)$. □

Next, the following proposition shows that, it suffices to consider policies that never idles station 1 when station 1 is not blocked.

Proposition 5.3.2. *There exists an optimal policy that never idles the server at station 1.*

Proof. We prove this proposition via sample path arguments. Note that, idling station 1 is equivalent to assigning a 'dummy' server i_0 with zero service rates and zero defect probability to station 1 (i.e., $\mu_{i_0,1} = 0, p_{i_0,1} = 0$). And by our previous analysis, assigning server i to station 1 with successful service rate $\hat{\mu}_{i,1}$ is equivalent to having a 'dummy' server i_1 with service rate $\hat{\mu}_{i,1}$ and zero defect probability at station 1 (i.e., $\mu_{i_1,1} = \hat{\mu}_{i,1}, p_{i_1,1} = 0$).

Consider the case an optimal policy π idles the server at station 1 in some state $s \in S$, without loss of generality, assume that server i is idled. Suppose policy π_0 is same as π except that it uses server i_0 at station 1 when π idles station 1, and π' is the same as π except that it uses the idling server i at station 1 when π idles station 1. Consider two processes on the same probability space, each starting from the same initial state s. Suppose that Process 1 uses policy π_0 that assigns server i_0 to station 1, and Process 2 uses policy π_1 that has the same server assignment as in π_0 except that it assigns server i_1 to station 1. Then, by Lemma 5.3.1, policy π_1 is at least as good as policy π_0. Since π_0 is equivalent to π, and π_1 is equivalent to π', we have policy π' at least as good as policy π. $\square$

By Proposition 5.3.2, we obtain that, when $N = 2$, the action set can be reduced to $A_s = \{a_{10}, a_{20}, a_{12}, a_{21}\}$ for $\forall s \in S$.

5.4 Optimal Policy for Two Stations

In this section, we consider the case with two stations and two servers, and fully characterize the optimal policy. Let $(\delta)^\infty$ denote the Markovian stationary deterministic policy corresponding to decision rule δ. For simplicity, when $N = 2$, let $B_1 = B$. Then, the state space is $S = \{0, \ldots, B+2\}$. Let

$$r = \frac{1 - p_{12}}{1 - p_{22}}$$

be the ratio of the success probability of servers 1 and 2 at station 2. This notation is helpful in describing the optimal policy. When there are two stations in tandem, our queueing system can be modeled as a birth-and-death process, with the birth rate in state s equal to the service completion rate with no defects (successful service rate) at station 1 in state s, and the death rate in state s equal to the service rate at station 2 in state s.

Without loss of generality, we can label the servers such that $\hat{\mu}_{11} \geq \hat{\mu}_{21}$, that is, server 1 has a higher successful service rate at station 1 than server 2. By Proposition 5.3.1, we should assign server 1 to station 1 when only station 1 is working (i.e., $s = 0$). However, when station 2 is working, the presence of defect probabilities make the structure of the optimal policy more complex than the one without defects since there is a trade-off between the speed and the reliability of the service. Specifically, if defect probabilities are zero (i.e., no defects), $\mu_{11} \geq \mu_{21}, \mu_{12} \leq \mu_{22}$, i.e., the two servers have their own specialty, then by Proposition 3 of [31], the optimal policy is always to assign server i to station i for $i = 1, 2$. Thus, when the defect probabilities are positive, one might conjecture that $\hat{\mu}_{11} \geq \hat{\mu}_{21}, \hat{\mu}_{12} \leq \hat{\mu}_{22}$ is sufficient for the same policy to be optimal. The next example shows that this is not always the case.

Example 5.4.1. *Consider the case when $B = 1$, then $S = \{0, 1, 2, 3\}$, $A_s = \{a_{10}, a_{20}, a_{12}, a_{21}\}$, $\forall s \in S$. Suppose $\hat{\mu}_{11} = 1.2, \hat{\mu}_{21} = 1, \mu_{12} = 1, \mu_{22} = 1.5, p_{22} = 0.7$. Then, the following three cases with different values of p_{12} satisfy $\hat{\mu}_{11} \geq \hat{\mu}_{21}, \hat{\mu}_{12} \leq \hat{\mu}_{22} = 1.05$, but the corresponding optimal policies are not the same as $(\delta)^\infty$, where $\delta(s) = a_{12}$, for $s \in S$.*

1. *If $p_{12} = 0.5$, then $(\delta_1^*)^\infty$ is optimal, where $\delta_1^*(0) = a_{12}, \delta_1^*(1) = \delta_1^*(2) = \delta_1^*(3) = a_{21}$;*

2. *If $p_{12} = 0.6$, then $(\delta_2^*)^\infty$ is optimal, where $\delta_2^*(0) = a_{12}, \delta_2^*(1) = \delta_2^*(2) = a_{21}, \delta_2^*(3) = a_{12}$;*

3. *If $p_{12} = 0.63$, then $(\delta_3^*)^\infty$ is optimal, where $\delta_3^*(0) = a_{12}, \delta_3^*(1) = a_{21}, \delta_3^*(2) = \delta_3^*(3) = a_{12}$.*

Moreover, $(\delta_2^*)^\infty, (\delta_3^*)^\infty$ are not included in any of the optimal policies characterized by [31].

We refer to $\hat{\mu}_{i1}\mu_{i2}$ as the *overall efficiency* of server i for $i = 1, 2$. Then, if $\hat{\mu}_{11} \geq \hat{\mu}_{21}$, we find that the optimal server allocation policy exhibits two different patterns depending on which server has a higher overall efficiency. That is, the optimal policy can be described by the following two cases: (1) $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$; (2) $\hat{\mu}_{11}\mu_{12} < \hat{\mu}_{21}\mu_{22}$. In Sections 5.4.1 and 5.4.2, we completely characterize the optimal policy for cases (1) and (2), respectively. In Section 5.4.3, we provide special cases when the optimal policy can be simplified.

5.4.1 Optimal Policy When $\hat{\mu}_{11} \geq \hat{\mu}_{21}$, $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$

In this section, we analyze the case when $\hat{\mu}_{11} \geq \hat{\mu}_{21}$, and $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$. Define δ_1^k for $k \in S$ such that

$$\delta_1^k(s) = \begin{cases} a_{12} & 0 \leq s \leq k, \\ a_{21} & k < s \leq B+2. \end{cases}$$

Note that $(\delta_1^k)^\infty$ is a non-idling threshold policy that assigns server i to station i in states $\{0, \dots, k\}$, and server i to station $3-i$ in states $\{k+1, \dots, B+2\}$ for $i = 1, 2$. Then, the corresponding long-run average throughput under policy $(\delta_1^k)^\infty$ is

$$T^{(\delta_1^k)^\infty} = \frac{\Theta_1(k)}{\Theta_2(k)}, \tag{5.6}$$

where

$$\Theta_1(k) = \hat{\mu}_{22}\mu_{12}^{B+2-k}\sum_{j=0}^{k-1}\hat{\mu}_{11}^{j+1}\mu_{22}^{k-1-j} + \hat{\mu}_{12}\hat{\mu}_{11}^{k+1}\sum_{j=0}^{B+1-k}\hat{\mu}_{21}^{j}\mu_{12}^{B+1-k-j},$$

$$\Theta_2(k) = \mu_{12}^{B+2-k}\sum_{j=0}^{k}\hat{\mu}_{11}^{j}\mu_{22}^{k-j} + \hat{\mu}_{11}^{k+1}\sum_{j=0}^{B+1-k}\hat{\mu}_{21}^{j}\mu_{12}^{B+1-k-j}.$$

For $k \in S \setminus \{0\}$, let

$$\begin{aligned}
c_1(k) &= \mu_{22}\Big[\mu_{22}^{k-1}\big(\mu_{12}^{B+3-k} + \hat{\mu}_{11}\sum_{j=0}^{B+1-k}\hat{\mu}_{21}^{j}\mu_{12}^{B+2-k-j}\big) + \hat{\mu}_{21}^{B+2-k}\sum_{j=0}^{k-1}\hat{\mu}_{11}^{j+1}\mu_{22}^{k-1-j}\Big],\\
c_2(k) &= \mu_{12}\Big(\mu_{22}^{k}\sum_{j=0}^{B+1-k}\hat{\mu}_{21}^{j}\mu_{12}^{B+2-k-j} + \hat{\mu}_{21}^{B+2-k}\sum_{j=0}^{k}\hat{\mu}_{11}^{j}\mu_{22}^{k-j}\Big),\\
C(k) &= \frac{c_1(k)}{c_2(k)},\\
f_1(k) &= (1-p_{22})c_1(k) - (1-p_{12})c_2(k).
\end{aligned}$$

Then $f_1(k)$ is positively proportional to $T^{(\delta_1^k)^\infty} - T^{(\delta_1^{k-1})^\infty}$, and $f_1(k) \geq 0$ if and only if $r \leq C(k)$.

We now present a lemma and a corollary that describe some useful properties of f_1.

Lemma 5.4.1. *When $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$, for $\forall k \in S \setminus \{0\}$, if $f_1(k) \geq 0$, then $f_1(i) \geq 0$ for $1 \leq i \leq k$; if $f_1(k) \leq 0$, then $f_1(j) \leq 0$ for $k \leq j \leq B+2$.*

Proof. Recall that $r = \frac{1-p_{12}}{1-p_{22}}$, and $f_1(k) \geq 0$ if and only if $r \leq C(k)$. Note that, for $k \in 1, \ldots B+1$,

$$\begin{aligned}
C(k+1) - C(k) &= \frac{\hat{\mu}_{21}^{B+1-k}\mu_{22}^{k-1}}{c_2(k)c_2(k+1)}(\hat{\mu}_{21}\mu_{22} - \hat{\mu}_{11}\mu_{12})\Theta_2(k)\\
&\leq 0.
\end{aligned}$$

That is, $C(k)$ is non-increasing in k when $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$. Then

$$f_1(k) \geq 0 \Rightarrow r \leq C(k) \Rightarrow r \leq C(k-1) \Rightarrow f_1(k-1) \geq 0.$$

Thus if $f_1(k) \geq 0$, $f_1(i) \geq 0$ for $0 \leq i \leq k$. We can prove the other half of the lemma by a similar argument. □

For ease of our analysis, define $f_1(0) = 0, f_1(B+3) = 0$. The following corollary follows from Lemma 5.4.1.

Corollary 5.4.1. *If $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$, then the set*

$$S_1^* = \{s \in S : f_1(s) \geq 0, f_1(s+1) \leq 0\}$$

is non-empty. Moreover, if there are multiple elements in S_1^*, then they are consecutive states.

Note that, if we denote $C(0) = \infty, C(B+3) = -\infty$, we can rewrite set S_1^* as follows:

$$S_1^* = \{s \in S : C(s+1) \leq r \leq C(s)\}. \tag{5.7}$$

Moreover, S_1^* is a singleton if $f_1(s) \neq 0$, for $\forall s \in S \setminus \{0\}$, or equivalently, $r \neq C(s)$, for $\forall s \in S \setminus \{0\}$. However, the converse is not true since if $\mu_{22} = 0$, $c_1(k) = 0$ for any $k \in S \setminus \{0\}$, which implies that $S_1^* = \{0\}$.

Lemma 5.4.1 shows that $T^{(\delta_1^k)^\infty}$ exhibits three possible behaviors with respect to k: if $f_1(B+2) \geq 0$, then $T^{(\delta_1^k)^\infty}$ is non-decreasing; if $f_1(1) \leq 0$, then $T^{(\delta_1^k)^\infty}$ is non-increasing; otherwise, $T^{(\delta_1^k)^\infty}$ is first non-decreasing and then non-increasing (however, this does not imply that $T^{(\delta_1^k)^\infty}$ is concave). Thus, the values of k such that $T^{(\delta_1^k)^\infty}$ is maximized would either be on the boundaries or be the turning point (or consecutive turning points). And if $|S_1^*| = 1$, i.e., S_1^* is a singleton, then $T^{(\delta_1^k)^\infty}$ has a unique optimum (maximum) point; if $|S_1^*| > 1$, then there are consecutive optimum points of $T^{(\delta_1^k)^\infty}$ with respect to k.

Theorem 5.4.1. *When $\hat{\mu}_{11} \geq \hat{\mu}_{21}$ and $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$, then $(\delta_1^{s^*})^\infty$ is optimal, where $s^* \in S_1^*$. Furthermore, it is the unique optimal policy in the class of Markovian stationary deterministic policies if $\hat{\mu}_{11} > \hat{\mu}_{21} > 0$, $f_1(s^*) > 0$, and $f_1(s^*+1) < 0$.*

Proof. It follows from our assumption on the service rates that $\mu_{11} > 0, \mu_{12} > 0$, and at least one of μ_{21}, μ_{22} is nonzero. Since the number of possible states and actions are both finite, by Theorem 9.1.8 of Puterman [41], there exists an optimal Markovian stationary deterministic policy.

Under our assumptions on the service rates and defect probabilities, the policy described in Theorem 5.4.1 implies that we have a communicating Markov decision process. Therefore, we use the Policy Iteration algorithm for communicating models to show that the policy we defined in this theorem is optimal. Choose the initial decision rule $\delta_0 = \delta_1^{s^*}$. Let r_{δ_0} and P_{δ_0} denote the corresponding reward vector and probability transition matrix, respectively. Without loss of generality, the uniformization constant can be taken as 1. Then

$$r(s, \delta_0(s)) = \begin{cases} 0 & \text{for } s = 0, \\ \hat{\mu}_{22} & \text{for } 1 \leq s \leq s^*, \\ \hat{\mu}_{12} & \text{for } s^* < s \leq B+2. \end{cases}$$

$$p(s'|s, \delta_0(s)) = \begin{cases} \hat{\mu}_{11} & \text{for } 0 \leq s \leq s^*, s' = s+1, \\ \hat{\mu}_{21} & \text{for } s^* < s \leq B+1, s' = s+1, \\ \mu_{22} & \text{for } 1 \leq s \leq s^*, s' = s-1, \\ \mu_{12} & \text{for } s^* < s \leq B+2, s' = s-1, \\ 1 - \hat{\mu}_{11} & \text{for } s = s' = 0, \\ 1 - (\hat{\mu}_{11} + \mu_{22}) & \text{for } 1 \leq s \leq \min\{s^*, B+1\}, s = s', \\ 1 - (\hat{\mu}_{21} + \mu_{12}) & \text{for } s^* < s \leq B+1, s = s', \\ 1 - \mu_{12} & \text{for } s = s' = B+2, s^* \leq B+1 \\ 1 - \mu_{22} & \text{for } s = s' = B+2, s^* = B+2 \\ 0 & \text{otherwise.} \end{cases}$$

Since the policy yields a unichain structure, we can solve the following equation to find a scalar g_0 and a vector h_0:

$$r_{\delta_0} - g_0 e + (P_{\delta_0} - I)h_0 = 0, \tag{5.8}$$

such that $h_0(0) = 0$, where e is the unit vector and I is the identity matrix. Then, $g_0 = T^{(\delta_1^{s^*})^\infty}$ as we defined in Eq.(5.6).

For $s \le s^* + 1$,

$$h_0(s) = \frac{g_0}{\hat{\mu}_{11}^s} \sum_{j=0}^{s-1} (j+1)\hat{\mu}_{11}^j \mu_{22}^{s-1-j} - \frac{\hat{\mu}_{22}}{\hat{\mu}_{11}^{s-1}} \sum_{j=0}^{s-2} (j+1)\hat{\mu}_{11}^j \mu_{22}^{s-2-j},$$

and for $s^* + 2 \le s \le B + 2$,

$$\begin{aligned} h_0(s) = h_0(s^*+1) &+ \frac{(g_0 - \hat{\mu}_{12})}{\hat{\mu}_{21}^{s-s^*-1}} \sum_{j=0}^{s-s^*-2} (j+1)\hat{\mu}_{21}^j \mu_{12}^{s-s^*-2-j} \\ &+ \frac{\mu_{12}}{\hat{\mu}_{21}^{s-s^*-1}} \sum_{j=0}^{s-s^*-2} \hat{\mu}_{21}^j \mu_{12}^{s-s^*-2-j} \big(\frac{g_0}{\hat{\mu}_{11}^{s^*+1}} \sum_{j=0}^{s^*} \hat{\mu}_{11}^j \mu_{22}^{s^*-j} - \frac{\hat{\mu}_{22}}{\hat{\mu}_{11}^{s^*}} \sum_{j=0}^{s^*-1} \hat{\mu}_{11}^j \mu_{22}^{s^*-1-j} \big). \end{aligned}$$

For the next step of the policy iteration algorithm, we choose

$$\delta_1(s) \in \arg\max_{a \in A_s} \left\{ r(s,a) + \sum_{j \in S} p(j|s,a) h_0(j) \right\}, \forall s \in S. \tag{5.9}$$

We now show that $\delta_0(s) = \delta_1(s)$ for all $s \in S$. In other words, the following inequality holds for all $s \in S, a \in A_s \setminus \{\delta_0(s)\}$:

$$\epsilon(s,a) = r(s,a) + \sum_{j \in S} p(j|s,a)h_0(j) - (r(s,\delta_0(s)) + \sum_{j \in S} p(j|s,\delta_0(s))h_0(j)) \le 0. \tag{5.10}$$

For $s \in \{0, 1, \ldots, s^*\}$, since $\delta_0(s) = a_{12}$, we will specify $\epsilon(s,a)$ for actions $\{a_{10}, a_{20}, a_{21}\}$. When $s = 0$, action a_{i0} is equivalent to $a_{i,3-i}$, for $i = 1, 2$ since station 2 is starved. Thus, we only need to specify $\epsilon(s,a)$ for a_{21}.

$$\epsilon(0, a_{21}) = \frac{1}{\hat{\mu}_{11}} (\hat{\mu}_{21} - \hat{\mu}_{11}) g_0 \le 0. \tag{5.11}$$

When $\hat{\mu}_{11} > \hat{\mu}_{21}$, inequality (5.11) is strict.

For $s = 1, \ldots, s^*$ and action a_{10},

$$\epsilon(s, a_{10}) = \frac{1}{\Theta_2(s^*)} \hat{\mu}_{11}^{s^*+1-s} \sum_{j=0}^{s-1} \hat{\mu}_{11}^{j} \mu_{22}^{s-1-j} (\hat{\mu}_{12}\kappa_1 - \hat{\mu}_{22}\kappa_2),$$

where

$$\kappa_1 = \mu_{22} \sum_{j=0}^{B+1-s^*} \hat{\mu}_{21}^{j} \mu_{12}^{B+1-s^*-j},$$

$$\kappa_2 = \hat{\mu}_{11} \sum_{j=0}^{B+1-s^*} \hat{\mu}_{21}^{j} \mu_{12}^{B+1-s^*-j} + \mu_{12}^{B+2-s^*}.$$

From equation (5.7), $r = \frac{1-p_{12}}{1-p_{22}} \leq C(s^*) = \frac{c_1(s^*)}{c_2(s^*)}$, then $\hat{\mu}_{22}\frac{c_1(s^*)}{\mu_{22}} \geq \hat{\mu}_{12}\frac{c_2(s^*)}{\mu_{12}}$, and

$$\begin{aligned}
\epsilon(s, a_{10}) &\leq \frac{\hat{\mu}_{22}}{\Theta_2(s^*) c_2(s^*)} \hat{\mu}_{11}^{s^*+1-s} \sum_{j=0}^{s-1} \hat{\mu}_{11}^{j} \mu_{22}^{s-1-j} \big(\frac{\mu_{12}}{\mu_{22}} c_1(s^*)\kappa_1 - c_2(s^*)\kappa_2\big) \\
&= -\frac{\hat{\mu}_{22}}{c_2(s^*)} \mu_{12} \hat{\mu}_{11}^{s^*+1-s} \hat{\mu}_{21}^{B+2-s^*} \sum_{j=0}^{s-1} \hat{\mu}_{11}^{j} \mu_{22}^{s-1-j} \\
&\leq 0.
\end{aligned}$$

Next, we specify $\epsilon(s, a)$ for $s = 1, \ldots, s^*$ and action a_{20},

$$\epsilon(s, a_{20}) = \frac{1}{\Theta_2(s^*)} (\hat{\mu}_{12}\kappa_3 - \hat{\mu}_{22}\kappa_4),$$

where

$$\begin{aligned}
\kappa_3 &= \hat{\mu}_{11}^{s^*-s} \sum_{j=0}^{B+1-s^*} \hat{\mu}_{21}^{j} \mu_{12}^{B+1-s^*-j} (\hat{\mu}_{21} \sum_{j=0}^{s} \hat{\mu}_{11}^{j} \mu_{22}^{s-j} - \hat{\mu}_{11}^{s+1}), \\
\kappa_4 &= \hat{\mu}_{11}^{s^*-s+1} \sum_{j=0}^{B+2-s^*} \hat{\mu}_{21}^{j} \mu_{12}^{B+2-s^*-j} \sum_{j=0}^{s-1} \hat{\mu}_{11}^{j} \mu_{22}^{s-1-j} \\
&\quad + (\hat{\mu}_{11} - \hat{\mu}_{21}) \mu_{12}^{B+2-s^*} \sum_{j=0}^{s^*-s-1} \hat{\mu}_{21}^{j} \mu_{12}^{s^*-1-j}.
\end{aligned}$$

Similarly, from equation (5.7), $\hat{\mu}_{22}\frac{c_1(s^*)}{\mu_{22}} \geq \hat{\mu}_{12}\frac{c_2(s^*)}{\mu_{12}}$, and

$$\begin{aligned}
\epsilon(s, a_{20}) &\leq \frac{\hat{\mu}_{12}}{\Theta_2(s^*)c_1(s^*)}\big[c_1(s^*)\kappa_3 - \frac{\mu_{22}}{\mu_{12}}c_2(s^*)\kappa_4\big] \\
&= -\frac{\hat{\mu}_{12}}{c_1(s^*)}\mu_{22}\Big\{\hat{\mu}_{11}^{s^*-s+1}\hat{\mu}_{21}^{B+2-s^*}\sum_{j=0}^{s-1}\hat{\mu}_{11}^{j}\mu_{22}^{s-1-j} \\
&+ (\hat{\mu}_{11} - \hat{\mu}_{21})\big[\mu_{12}\mu_{22}^{s^*-1}\sum_{j=0}^{B+1-s^*}\hat{\mu}_{21}^{j}\mu_{12}^{B+1-s^*-j} + \hat{\mu}_{21}^{B+2-s^*}\sum_{j=0}^{s^*-s-1}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-1-j}\big]\Big\} \\
&\leq 0.
\end{aligned}$$

Next, for action a_{21} and $s = 1, \dots, s^*$,

$$\epsilon(s, a_{21}) = -\frac{1}{\Theta_2(s^*)}\Big[f_1(s^*) + (\hat{\mu}_{11}\mu_{12} - \hat{\mu}_{21}\mu_{22})\sum_{j=0}^{s^*-s-1}\hat{\mu}_{11}^{j}\mu_{22}^{s^*-1-j} \times \Upsilon_1\Big], \qquad (5.12)$$

where

$$\Upsilon_1 = \big[(\hat{\mu}_{11} - \hat{\mu}_{21})(1 - p_{22}) + \mu_{12}(p_{12} - p_{22})\big]\sum_{j=0}^{B+1-s^*}\hat{\mu}_{21}^{j}\mu_{12}^{B+1-s^*-j} + \hat{\mu}_{21}^{B+2-s^*}(1 - p_{22}).$$

When $s^* = B + 2$, $\Upsilon_1 = 1 - p_{22} \geq 0$, $\epsilon(s, a_{21}) \leq 0$ for all $s \in S$; when $s^* < B + 2$, by the analysis in Lemma 5.4.1, we have $r \leq C(1)$, and

$$r \leq C(1) \Rightarrow \Upsilon_1 \geq 0.$$

Thus, $\epsilon(s, a_{21}) \leq 0$ for $s = 1, \dots, s^*$, and the inequality is strict when $f_1(s^*) > 0$.

For $s \in \{s^* + 1, \dots, B + 2\}$, since $\delta_0(s) = a_{21}$, we will specify $\epsilon(s, a)$ for actions $\{a_{10}, a_{20}, a_{12}\}$.

For action a_{10} and $s \in \{s^* + 1, \dots, B + 2\}$,

$$\epsilon(s, a_{10}) = \frac{\hat{\mu}_{11}}{\Theta_2(s^*)}(\hat{\mu}_{12}\kappa_5 - \hat{\mu}_{22}\kappa_6),$$

where

$$\kappa_5 = \mu_{12}^{s-s^*} \sum_{j=0}^{s^*} \hat{\mu}_{11}^j \mu_{22}^{s^*-j} \sum_{j=0}^{B+1-s} \hat{\mu}_{21}^j \mu_{12}^{B+1-s-j} - \hat{\mu}_{11}^{s^*} \sum_{j=0}^{B+1-s^*} \hat{\mu}_{21}^j \mu_{12}^{B+1-s^*-j},$$

$$\kappa_6 = \mu_{12}^{s-s^*} \sum_{j=0}^{s^*-1} \hat{\mu}_{11}^j \mu_{22}^{s^*-1-j} \big(\hat{\mu}_{11} \sum_{j=0}^{B+1-s} \hat{\mu}_{21}^j \mu_{12}^{B+1-s-j} + \mu_{12}^{B+2-s}\big).$$

From equation (5.7), $\hat{\mu}_{22}\frac{c_1(s^*)}{\mu_{22}} \geq \hat{\mu}_{12}\frac{c_2(s^*)}{\mu_{12}}$, and

$$\begin{aligned}
\epsilon(s, a_{10}) &\leq \frac{\hat{\mu}_{11}\hat{\mu}_{12}}{\Theta_2(s^*)c_1(s^*)}\big[c_1(s^*)\kappa_5 - \frac{\mu_{22}}{\mu_{12}}c_2(s^*)\kappa_6\big] \\
&= -\frac{\hat{\mu}_{12}}{c_1(s^*)}\hat{\mu}_{11}\mu_{22}\hat{\mu}_{21}^{B+2-s}\big(\hat{\mu}_{21}^{s-s^*} \sum_{j=0}^{s^*-1} \hat{\mu}_{11}^j \mu_{22}^{s^*-1-j} + \mu_{12}\mu_{22}^{s^*-1} \sum_{j=0}^{s-s^*-1} \hat{\mu}_{21}^j \mu_{12}^{s-s^*-1-j}\big) \\
&\leq 0.
\end{aligned}$$

Next, for action a_{20} and $s \in \{s^*+1, \ldots, B+2\}$,

$$\epsilon(s, a_{20}) = \frac{1}{\Theta_2(s^*)}(\hat{\mu}_{12}\kappa_7 - \hat{\mu}_{22}\kappa_8),$$

where

$$\kappa_7 = \hat{\mu}_{21}\mu_{12}^{s-s^*} \sum_{j=0}^{s^*} \hat{\mu}_{11}^j \mu_{22}^{s^*-j} \sum_{j=0}^{B+1-s} \hat{\mu}_{21}^j \mu_{12}^{B+1-s-j} - \hat{\mu}_{11}^{s^*+1} \sum_{j=0}^{B+1-s^*} \hat{\mu}_{21}^j \mu_{12}^{B+1-s^*-j},$$

$$\kappa_8 = \hat{\mu}_{11}\mu_{12}^{s-s^*} \sum_{j=0}^{s^*-1} \hat{\mu}_{11}^j \mu_{22}^{s^*-1-j} \sum_{j=0}^{B+2-s} \hat{\mu}_{21}^j \mu_{12}^{B+2-s-j}.$$

From equation (5.7), $\hat{\mu}_{22}\frac{c_1(s^*)}{\mu_{22}} \geq \hat{\mu}_{12}\frac{c_2(s^*)}{\mu_{12}}$, and

$$\epsilon(s, a_{20}) \leq \frac{\hat{\mu}_{12}}{\Theta_2(s^*)c_1(s^*)}\big[c_1(s^*)\kappa_7 - \frac{\mu_{22}}{\mu_{12}}c_2(s^*)\kappa_8\big]$$

$$
\begin{aligned}
&= -\frac{\hat{\mu}_{12}}{c_1(s^*)}\mu_{22}\Big[(\hat{\mu}_{11}-\hat{\mu}_{21})\mu_{12}\mu_{22}^{s^*-1}\sum_{j=0}^{B+1-s^*}\hat{\mu}_{21}^{j}\mu_{12}^{B+1-s^*-j} \\
&+\hat{\mu}_{21}^{B+2-s^*}\hat{\mu}_{11}\sum_{j=0}^{s^*-1}\hat{\mu}_{11}^{j}\mu_{22}^{s^*-1-j}+\mu_{12}\hat{\mu}_{21}^{B+3-s}\mu_{22}^{s^*-1}\sum_{j=0}^{s-s^*-1}\hat{\mu}_{21}^{j}\mu_{12}^{s-s^*-1-j}\Big] \\
&\le 0.
\end{aligned}
$$

Finally, for action a_{12} and $s \in \{s^*+1,\dots,B+2\}$,

$$
\epsilon(s,a_{12}) = \frac{1}{\Theta_2(s^*)}\Big[f_1(s^*+1)-(\hat{\mu}_{11}\mu_{12}-\hat{\mu}_{21}\mu_{22})\sum_{j=0}^{s^*-s-2}\mu_{12}^{j}\hat{\mu}_{21}^{B-s^*-j}\times\Upsilon_2\Big], \quad (5.13)
$$

where

$$
\Upsilon_2 = (\hat{\mu}_{12}-\hat{\mu}_{22})\sum_{j=0}^{s^*}\hat{\mu}_{11}^{j}\mu_{22}^{s^*-j}+\mu_{22}^{s^*+1}(1-p_{22}).
$$

When $s^*=0$, $\Upsilon_2=\hat{\mu}_{12}\ge 0$, $\epsilon(s,a_{12})\le 0$ for $s\in\{1,\dots,B+2\}$; when $s^*>0$, by the analysis in Lemma 5.4.1, we have $r\ge C(B+2)$, and

$$
r \ge C(B+2) \Rightarrow \Upsilon_2 \ge 0.
$$

Thus, $\epsilon(s,a_{12})\le 0$ for $s=s^*+1,\dots,B+2$, and the inequality is strict when $f_1(s^*+1)<0$.

This proves that $\delta_0(s)=\delta_1(s)$ for all $s\in S$. Thus, by Theorem 9.5.1 of Puterman [41], the policy described in this theorem is optimal.

To prove uniqueness among Markovian stationary deterministic policies, we use a similar approach to Andradóttir and Ayhan [6]. Consider a decision rule δ' that differs from δ_0 in at least one state $s\in S$. Define

$$
\begin{aligned}
&u = P_{\delta'}g_0e - g_0e = 0, \\
&v = r_{\delta'}+(P_{\delta'}-I)h_0-g_0e = r_{\delta'}+P_{\delta'}h_0-(r_{\delta_0}+P_{\delta_0}h_0),
\end{aligned}
$$

where we have used equation (5.8). Note that inequality (5.10) holds for all $s \in S, a \in A_s \setminus \{\delta_0(s)\}$, and is strict when $\hat{\mu}_{11} > \hat{\mu}_{21} > 0$, $f_1(s^*) > 0$, and $f_1(s^* + 1) < 0$. Thus, $v(s) \leq 0$ for all $s \in S$, and if $\hat{\mu}_{11} > \hat{\mu}_{21} > 0$, $f_1(s^*) > 0$, $f_1(s^* + 1) < 0$, we must have

$$v(s) < 0, \text{ for } \forall s \in S \text{ with } \delta'(s) \neq \delta_0(s). \tag{5.14}$$

Let g' denote the (possibly state dependent) throughput of the stationary policy $(\delta')^\infty$. Suppose that $P_{\delta'}$ has n recurrent classes and partition $P_{\delta'}$ such that $P_1, \ldots, P_n$ correspond to transitions within closed recurrent classes, $Q_1, \ldots, Q_n$ to transitions from transient to recurrent states, and Q_{n+1} to transitions between transient states. Define $\Delta g = g' - g_0 e$, and let $P^*_{\delta'}$ be the limiting matrix under decision rule δ'. Partition g', Δg, v, and $P^*_{\delta'}$ in a manner that is consistent with this partition of $P_{\delta'}$. Then, from Lemma 9.2.5 of Puterman [41], we can obtain that

$$\Delta g_i = P^*_i v_i, \text{ for } i = 1, \ldots, n. \tag{5.15}$$

If both μ_{21} and μ_{22} are positive, P_{δ_0} is irreducible; and if $\mu_{22} = 0, \mu_{21} > 0$, then $s^* = 0$, and $\delta_0 = \delta_1^0$ also result in irreducible transition matrices. Hence, when $\mu_{21} > 0$, P_{δ_0} is irreducible. Since $\delta'(s) \neq \delta_0(s)$ for some state $s \in S$, then $\delta'(s) \neq \delta_0(s)$ in at least one state $s_0 \in S$ that is recurrent under δ'. However, equations (5.14) and (5.15) imply that $g'(s_0) < g_0$, so that the decision rule δ' can not be optimal. Therefore, when $\hat{\mu}_{11} > \hat{\mu}_{21}$, $f_1(s^*) > 0$, $f_1(s^* + 1) < 0$, the optimal policy is unique. $\square$

Intuitively, when $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$, server 1 is overall more effective than server 2. To increase the long-run average throughput of the system, when the number of jobs in the system is small (i.e., $s \leq s^*$), our priority is to push more jobs into the system and avoid starving at station 2, thus we assign the overall more efficient server (i.e., server 1) to station 1; as the number of jobs in the system gets larger and the system becomes more crowded, our priority changes to push more jobs out of the system and avoid blocking at station 1, therefore we switch the assignment and assign the overall more efficient server to station 2.

In Theorem 5.4.1, we provide conditions showing how the the service rates and defect probabilities at station 1 impact the structure of the optimal policy. However, the effect of the defect probabilities at station 2 is implicit in the value of s^*. To better illustrate the influence of defect probabilities at station 2 on the optimal policy, we obtain the following corollary that follows from Theorem 5.4.1 and equation (5.7).

Corollary 5.4.2. *When $\hat{\mu}_{11} \geq \hat{\mu}_{21}$ and $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$, Table 5.2 shows the optimal policy as a function of $r = \frac{1-p_{12}}{1-p_{22}}$.*

Table 5.2: Optimal policy in case (1) as a function of r.

Range of r	Optimal Policy	a_{12} Optimal in States	a_{21} Optimal in States
$r < C(B+2)$	$(\delta_1^{B+2})^\infty$	$0, 1, \ldots, B+2$	$\emptyset$
$C(B+2) \leq r < C(B+1)$	$(\delta_1^{B+1})^\infty$	$0, 1, \ldots, B+1$	$B+2$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$C(2) \leq r < C(1)$	$(\delta_1^{1})^\infty$	$0, 1$	$2, \ldots, B+2$
$C(1) \leq r$	$(\delta_1^{0})^\infty$	0	$1, \ldots, B+2$

Note that, a_{12} is always the optimal action in state 0 regardless of r as shown in Table 5.2, which coincides with Proposition 5.3.1 since server 1 has a higher successful service rate at station 1 than server 2 (i.e., $\hat{\mu}_{11} \geq \hat{\mu}_{21}$).

From Table 5.2 we observe that, the value of $s^* \in S_1^*$ lies in on the boundary of S when r is either small or high, and s^* is non-increasing with respect to r. Intuitively, when station 2 is working (i.e., $s > 0$), there is a trade-off between the faster server and the more reliable server. We attach more importance to the defect probability than the service rate for station 2 since we do not want to waste our efforts at the first station. Therefore, when one server is much more reliable than the other, that is, when r is either very high (i.e., $r \geq C(1)$) or very small (i.e., $r < C(B+2)$), we would always assign the more reliable server at station 2 to station 2. Otherwise, when r is moderate, as r increases, server 2 becomes relatively less reliable at station 2, thus a smaller value of s^* increases the relative time of server 2 working at station 2 and yields a higher throughput (recall that s^* is the threshold point at which we switch our server assignment).

5.4.2 Optimal Policy When $\hat{\mu}_{11} \geq \hat{\mu}_{21}$, $\hat{\mu}_{11}\mu_{12} \leq \hat{\mu}_{21}\mu_{22}$

In this section, we discuss the case when $\hat{\mu}_{11} \geq \hat{\mu}_{21}$, and $\hat{\mu}_{11}\mu_{12} \leq \hat{\mu}_{21}\mu_{22}$. The steps are similar to Case (1). Define δ_2^k for $k \in S$ such that

$$\delta_2^k(s) = \begin{cases} a_{12} & s = 0, \\ a_{21} & 1 \leq s \leq k, \\ a_{12} & k < s \leq B+2. \end{cases}$$

Observe that the policy $(\delta_2^k)^\infty$ is a non-idling policy with two thresholds. First, by Proposition 5.3.1, we will always assign server 1 to station 1 when in state s under the assumption $\hat{\mu}_{11} \geq \hat{\mu}_{21}$. Policy $(\delta_2^k)^\infty$ assigns server i to station $3-i$ in states $\{1, \dots, k\}$ and assigns server i to station i in states $\{k+1, \dots, B+2\}$ for $i = 1, 2$. Then, the corresponding long-run average throughput under policy $(\delta_2^k)^\infty$ is

$$T^{(\delta_2^k)^\infty} = \frac{\Theta_3(k)}{\Theta_4(k)}, \tag{5.16}$$

where

$$\Theta_3(k) = \hat{\mu}_{11}\Big(\hat{\mu}_{22}\hat{\mu}_{21}^k \sum_{j=0}^{B+1-k} \hat{\mu}_{11}^j \mu_{22}^{B+1-k-j} + \hat{\mu}_{12}\mu_{22}^{B+2-k} \sum_{j=0}^{k-1} \hat{\mu}_{21}^j \mu_{12}^{k-1-j}\Big),$$
$$\Theta_4(k) = \hat{\mu}_{11}\big(\hat{\mu}_{21}^k \sum_{j=0}^{B+1-k} \hat{\mu}_{11}^j \mu_{22}^{B+1-k-j} + \mu_{22}^{B+2-k} \sum_{j=0}^{k-1} \hat{\mu}_{21}^j \mu_{12}^{k-1-j}\big) + \mu_{12}^k \mu_{22}^{B+2-k}.$$

For $k \in S \setminus \{0\}$, let

$$\begin{aligned} d_1(k) &= \mu_{22}\Big\{\mu_{12}^{k-1}\big[(\hat{\mu}_{11} - \hat{\mu}_{21} + \mu_{12}) \sum_{j=0}^{B+2-k} \hat{\mu}_{11}^{j}\mu_{22}^{B+2-k-j} + \hat{\mu}_{21}\hat{\mu}_{11}^{B+2-k}\big] \\ &\quad + \hat{\mu}_{11}^{B+3-k}\sum_{j=0}^{k-2}\hat{\mu}_{21}^{j+1}\mu_{12}^{k-2-j}\Big\}, \\ d_2(k) &= \mu_{12}\Big(\mu_{12}^{k-1}\sum_{j=0}^{B+3-k}\hat{\mu}_{11}^{j}\mu_{22}^{B+3-k-j} + \hat{\mu}_{11}^{B+3-k}\sum_{j=0}^{k-2}\hat{\mu}_{21}^{j+1}\mu_{12}^{k-2-j}\Big), \\ D(k) &= \frac{d_1(k)}{d_2(k)}, \\ f_2(k) &= (1-p_{12})d_2(k) - (1-p_{22})d_1(k). \end{aligned}$$

Then $f_2(k)$ is positively proportional to $T^{(\delta_2^k)^\infty} - T^{(\delta_2^{k-1})^\infty}$, and $\phi(k) \geq 0$ if and only if $r \geq D(k)$. Moreover, $D(k) \geq 1$ for $\forall k \in S \setminus \{0\}$.

Next, we provide a lemma and a corollary to describe the properties of $f_2(k)$ that would be useful to interpret our results.

Lemma 5.4.2. *When $\hat{\mu}_{11}\mu_{12} \leq \hat{\mu}_{21}\mu_{22}$, for $\forall k \in S \setminus \{0\}$, if $f_2(k) \geq 0$, then $f_2(i) \geq 0$ for $1 \leq i \leq k$; if $f_2(k) \leq 0$, then $f_2(j) \leq 0$ for $k \leq j \leq B+2$.*

Proof. For $k \in S \setminus \{0\}$,

$$\begin{aligned} D(k+1) - D(k) &= \frac{\hat{\mu}_{11}^{B+1-k}\mu_{12}^{k-1}}{d_2(k)d_2(k+1)}(\hat{\mu}_{21}\mu_{22} - \hat{\mu}_{11}\mu_{12})\Theta_4(k) \\ &\geq 0. \end{aligned}$$

That is, $D(k)$ is non-decreasing in k when $\hat{\mu}_{11}\mu_{12} \leq \hat{\mu}_{21}\mu_{22}$. Then

$$f_2(k) \geq 0 \Rightarrow r \geq D(k) \Rightarrow r \geq D(k-1) \Rightarrow f_2(k-1) \geq 0.$$

Thus, if $f_2(k) \geq 0$, $f_2(i) \geq 0$ for $0 \leq i \leq k$. We can prove the other half of the lemma by a similar argument. □

For ease of our analysis, define $f_2(0) = 0, f_2(B+3) = 0$.

Corollary 5.4.3. *If $\hat{\mu}_{11}\mu_{12} \leq \hat{\mu}_{21}\mu_{22}$, then the set*

$$S_2^* = \{s \in S : f_2(s) \geq 0, f_2(s+1) \leq 0\}$$

is non-empty. Moreover, if there are multiple elements in S_2^*, then they are consecutive states.

Note that, if we denote $D(0) = -\infty, D(B+3) = \infty$, we can rewrite set S_2^* as follows:

$$S_2^* = \{s \in S : D(s) \leq r \leq D(s+1)\}. \tag{5.17}$$

Moreover, S_2^* is a singleton if $f_2(s) \neq 0$, for $\forall s \in S \setminus \{0\}$, or equivalently, $r \neq D(s)$, for $\forall s \in S \setminus \{0\}$.

Observe that the structure of f_2 is similar to that of f_1. Similarly, by Lemma 5.4.2, the values of k such that $T^{(\delta_2^k)^\infty}$ is maximized would either be on the boundaries or be the turning point (or consecutive turning points). And if $|S_2^*| = 1$, i.e., S_2^* has a single element, then $T^{(\delta_2^k)^\infty}$ has a unique optimum (maximum) point; if $|S_2^*| > 1$, then there are consecutive optimum points of $T^{(\delta_2^k)^\infty}$ with respect to k.

Theorem 5.4.2. *When $\hat{\mu}_{11} \geq \hat{\mu}_{21}$ and $\hat{\mu}_{11}\mu_{12} \leq \hat{\mu}_{21}\mu_{22}$, then $(\delta_2^{s^*})^\infty$ is optimal, where $s^* \in S_2^*$. Furthermore, it is the unique optimal policy in the class of Markovian stationary deterministic policies if $\hat{\mu}_{11} > \hat{\mu}_{21}$, $f_2(s^*) > 0$, and $f_2(s^*+1) < 0$.*

Proof. The proof of Theorem 5.4.2 is similar to the proof of Theorem 5.4.1. First, $\hat{\mu}_{11} \geq \hat{\mu}_{21}$ implies that $\mu_{11} > 0$ and $\mu_{12} > 0$. Moreover, $\hat{\mu}_{11}\mu_{12} \leq \hat{\mu}_{21}\mu_{22}$ implies that μ_{21}, μ_{22} are also positive. Since the number of possible states and actions are both finite, by Theorem 9.1.8 of Puterman [41], there exists an optimal Markovian stationary deterministic policy.

Under our assumptions on the service rates, the policy described in Theorem 5.4.2 implies a irreducible Markov chain. Therefore, we have a communicating Markov decision process. We again use the Policy Iteration algorithm for communicating models to show that the policy we defined in this theorem is optimal. Choose the initial decision $\delta'_0 = \delta_2^{s^*}$, let $r_{\delta'_0}$ and $P_{\delta'_0}$ denote the corresponding reward vector and probability transition matrix, respectively. Then

$$r(s,\delta'_0(s)) = \begin{cases} 0 & \text{for } s = 0, \\ \hat{\mu}_{12} & \text{for } 1 \le s \le s^*, \\ \hat{\mu}_{22} & \text{for } s^* < s \le B+2; \end{cases}$$

$$p(s'|s,\delta'_0(s)) = \begin{cases} \hat{\mu}_{11} & \text{for } s = 0, s' = 1, \\ \hat{\mu}_{21} & \text{for } 1 \le s \le s^*, s' = s+1, \\ \hat{\mu}_{11} & \text{for } s^* < s \le B+1, s' = s+1, \\ \mu_{12} & \text{for } 1 \le s \le s^*, s' = s-1, \\ \mu_{22} & \text{for } s^* < s \le B+2, s' = s-1, \\ 1-\hat{\mu}_{11} & \text{for } s = s' = 0, \\ 1-(\hat{\mu}_{21}+\mu_{12}) & \text{for } 1 \le s \le s^*, s = s', \\ 1-(\hat{\mu}_{11}+\mu_{22}) & \text{for } s^* < s \le B+1, s = s', \\ 1-\mu_{22} & \text{for } s = s' = B+2, s^* \le B+1, \\ 1-\mu_{12} & \text{for } s = s' = B+2, s^* = B+2, \\ 0 & \text{otherwise.} \end{cases}$$

Since the policy yields unichain structure, we can solve the following equation to find a scalar g'_0 and a vector h'_0:

$$r_{\delta'_0} - g'_0 e + (P_{\delta'_0} - I)h'_0 = 0, \tag{5.18}$$

such that $h'_0(0) = 0$, where e is the unit vector and I is the identity matrix. Then, $g'_0 =$

$T^{(\delta_2^{s^*})^\infty}$ as we defined in equation (5.16).

For $s \le s^* + 1$,

$$h_0'(s) = \frac{g_0'}{\hat{\mu}_{11}\hat{\mu}_{21}^{s-1}}\Big[\hat{\mu}_{11}\sum_{j=0}^{s-2}(j+1)\hat{\mu}_{21}^{j}\mu_{12}^{s-2-j} + \sum_{j=0}^{s-1}\hat{\mu}_{21}^{j}\mu_{12}^{s-1-j}\Big] - \frac{\hat{\mu}_{12}}{\hat{\mu}_{21}^{s-1}}\sum_{j=0}^{s-2}(j+1)\hat{\mu}_{21}^{j}\mu_{12}^{s-2-j},$$

and for $s^* + 2 \le s \le B + 2$,

$$\begin{aligned} h_0'(s) = {} & h_0'(s^*+1) + \frac{(g_0' - \hat{\mu}_{22})}{\hat{\mu}_{11}^{s-s^*-1}}\sum_{j=0}^{s-s^*-2}(j+1)\hat{\mu}_{11}^{j}\mu_{22}^{s-s^*-2-j} \\ & + \frac{\mu_{22}}{\hat{\mu}_{11}^{s-s^*-1}}\sum_{j=0}^{s-s^*-2}\hat{\mu}_{11}^{j}\mu_{22}^{s-s^*-2-j}\Big[\frac{g_0'}{\hat{\mu}_{11}\hat{\mu}_{21}^{s^*}}\big(\hat{\mu}_{11}\sum_{j=0}^{s^*-1}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-1-j} + \mu_{12}^{s^*}\big) \\ & - \frac{\hat{\mu}_{12}}{\hat{\mu}_{21}^{s^*}}\sum_{j=0}^{s^*-1}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-1-j}\Big]. \end{aligned}$$

For the next step of the policy iteration algorithm, we choose

$$\delta_1'(s) \in \arg\max_{a \in A_s}\left\{r(s,a) + \sum_{j \in S} p(j|s,a)h_0'(j)\right\}, \forall s \in S. \tag{5.19}$$

We now show that $\delta_0'(s) = \delta_2(s)$ for all $s \in S$. In other words, we will prove that the following inequality holds for all $s \in S, a \in A_s \setminus \{\delta_0(s)\}$:

$$\epsilon'(s,a) = r(s,a) + \sum_{j \in S} p(j|s,a)h_0'(j) - (r(s,\delta_0'(s)) + \sum_{j \in S} p(j|s,\delta_0'(s))h_0'(j)) \le 0. \tag{5.20}$$

When $s = 0$, a_{i0} and $a_{i,3-i}$ are equivalent, for $i = 1, 2$ since station 2 is starved. Thus, we only need to specify $\epsilon(s,a)$ for a_{21}.

$$\epsilon'(0,a_{21}) = \epsilon'(0,a_{20}) = \frac{1}{\hat{\mu}_{11}}(\hat{\mu}_{21} - \hat{\mu}_{11})g_0' \le 0, \text{ and}$$

Note that the above inequality is strict when $\hat{\mu}_{21} > \hat{\mu}_{11}$.

When $s \in \{1, \ldots, s^*\}$, $\delta_0'(s) = a_{21}$, and we will specify $\epsilon'(s,a)$ for actions $\{a_{10}, a_{20}, a_{12}\}$.

For $s = 1, \ldots, s^*$ and action a_{10},

$$\epsilon'(s, a_{10}) = \frac{1}{\Theta_4(s^*)} \hat{\mu}_{11} \hat{\mu}_{21}^{s^*-s} \sum_{j=0}^{s-1} \hat{\mu}_{21}^{j} \mu_{12}^{s-1-j} (\hat{\mu}_{22} \kappa'_1 - \hat{\mu}_{12} \kappa'_2),$$

where

$$\kappa'_1 = (\hat{\mu}_{11} - \hat{\mu}_{21} + \mu_{12}) \sum_{j=0}^{B+1-s^*} \hat{\mu}_{11}^{j} \mu_{22}^{B+1-s^*-j},$$

$$\kappa'_2 = \sum_{j=0}^{B+2-s^*} \hat{\mu}_{11}^{j} \mu_{22}^{B+2-s^*-j}.$$

From equation (5.17), $r = \frac{1-p_{12}}{1-p_{22}} \geq D(s^*) = \frac{d_1(s^*)}{d_2(s^*)}$, then $\hat{\mu}_{22} \frac{d_1(s^*)}{\mu_{22}} \leq \hat{\mu}_{12} \frac{d_2(s^*)}{\mu_{12}}$, and

$$\begin{aligned}
\epsilon'(s, a_{10}) &\leq \frac{\hat{\mu}_{22}}{\Theta_4(s^*) d_2(s^*)} \hat{\mu}_{11} \hat{\mu}_{21}^{s^*-s} \sum_{j=0}^{s-1} \hat{\mu}_{21}^{j} \mu_{12}^{s-1-j} \big(d_2(s^*) \kappa'_1 - \frac{\mu_{12}}{\mu_{22}} d_1(s^*) \kappa'_2\big) \\
&= -\frac{\hat{\mu}_{22}}{d_2(s^*)} \mu_{12} \hat{\mu}_{11}^{B+3-s^*} \hat{\mu}_{21}^{s^*-s} \sum_{j=0}^{s-1} \hat{\mu}_{21}^{j} \mu_{12}^{s-1-j} \\
&\leq 0.
\end{aligned}$$

Next, for $s = 1, \ldots, s^*$ and action a_{20},

$$\epsilon'(s, a_{20}) = \frac{1}{\Theta_4(s^*)} (\hat{\mu}_{22} \kappa'_3 - \hat{\mu}_{12} \kappa'_4),$$

where

$$\begin{aligned}
\kappa'_3 &= \hat{\mu}_{21}^{s^*+1-s} \sum_{j=0}^{B+1-s^*} \hat{\mu}_{11}^{j} \mu_{22}^{B+1-s^*-j} (\hat{\mu}_{11} \mu_{12} \sum_{j=0}^{s-2} \hat{\mu}_{21}^{j} \mu_{12}^{s-2-j} + \mu_{12}^{s}), \\
\kappa'_4 &= \hat{\mu}_{11} \hat{\mu}_{21}^{s^*+1-s} \sum_{j=0}^{B+1-s^*} \hat{\mu}_{11}^{j} \mu_{22}^{B+1-s^*-j} \sum_{j=0}^{s-1} \hat{\mu}_{21}^{j} \mu_{12}^{s-1-j} \\
&\quad + \mu_{22}^{B+2-s^*} \big(\hat{\mu}_{11} \sum_{j=0}^{s^*-1} \hat{\mu}_{21}^{j} \mu_{12}^{s^*-1-j} - \hat{\mu}_{21} \sum_{j=0}^{s^*-s-1} \hat{\mu}_{21}^{j} \mu_{12}^{s^*-1-j}\big).
\end{aligned}$$

Similarly, from equation (5.17), $\hat{\mu}_{22}\frac{d_1(s^*)}{\mu_{22}} \leq \hat{\mu}_{12}\frac{d_2(s^*)}{\mu_{12}}$, and

$$\begin{aligned}
\epsilon'(s, a_{20}) &\leq \frac{\hat{\mu}_{12}}{\Theta_4(s^*)d_1(s^*)}\big[\frac{\mu_{22}}{\mu_{12}}d_2(s^*)\kappa'_3 - d_1(s^*)\kappa'_4\big] \\
&= -\frac{\hat{\mu}_{12}}{d_1(s^*)}\mu_{22}\Big\{\hat{\mu}_{11}^{B+3-s^*}\hat{\mu}_{21}^{s^*-s}\sum_{j=0}^{s-1}\hat{\mu}_{21}^{j}\mu_{12}^{s-1-j} \\
&+ (\hat{\mu}_{11} - \hat{\mu}_{21})\big[\mu_{12}^{s^*-1}\mu_{22}\sum_{j=0}^{B+1-s^*}\hat{\mu}_{11}^{j}\mu_{22}^{B+1-s^*-j} + \hat{\mu}_{11}^{B+2-s^*}\sum_{j=0}^{s^*-s-1}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-1-j}\big]\Big\} \\
&\leq 0.
\end{aligned}$$

Next, for $s = 1, \ldots, s^*$ and action a_{12},

$$\epsilon'(s, a_{12}) = -\frac{1}{\Theta_4(s^*)}\Big[f_2(s^*) + (\hat{\mu}_{21}\mu_{22} - \hat{\mu}_{11}\mu_{12})\sum_{j=0}^{s^*-s-1}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-2-j} \times \Upsilon'_1\Big], \qquad (5.21)$$

where

$$\Upsilon'_1 = \hat{\mu}_{12}\sum_{j=0}^{B+2-s^*}\hat{\mu}_{11}^{j}\mu_{22}^{B+2-s^*-j} - \hat{\mu}_{22}(\hat{\mu}_{11} - \hat{\mu}_{21} + \mu_{12})\sum_{j=0}^{B+1-s^*}\hat{\mu}_{11}^{j}\mu_{22}^{B+1-s^*-j}.$$

When $s^* = B + 2$, $\Upsilon'_1 = \hat{\mu}_{12} \geq 0$, $\epsilon'(s, a_{12}) \leq 0$ for all $s \in S$; when $s^* < B + 2$, by the analysis in Lemma 5.4.2, we have $r \geq D(1)$, and

$$r \geq D(1) \Rightarrow \Upsilon'_1 \geq 0.$$

Thus, $\epsilon'(s, a_{12}) \leq 0$ for $s = 1, \ldots, s^*$.

For $s \in \{s^*{+}1, \ldots, B{+}2\}$, $\delta'_0(s) = a_{12}$, we will specify $\epsilon'(s, a)$ for actions $\{a_{10}, a_{20}, a_{21}\}$.

For $s \in \{s^* + 1, \ldots, B + 2\}$ and action a_{10},

$$\epsilon'(s, a_{10}) = \frac{1}{\Theta_4(s^*)}(\hat{\mu}_{22}\kappa'_5 - \hat{\mu}_{12}\kappa'_6),$$

where

$$\kappa_5' = \hat{\mu}_{11}\mu_{22}^{s-s^*}\sum_{j=0}^{B+1-s}\hat{\mu}_{11}^{j}\mu_{22}^{B+1-s-j}\big(\mu_{12}^{s^*}+\hat{\mu}_{11}\sum_{j=0}^{s^*-1}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-1-j}\big)-\hat{\mu}_{11}\hat{\mu}_{21}^{s^*}\sum_{j=0}^{B+1-s^*}\hat{\mu}_{11}^{j}\mu_{22}^{B+1-s^*-j},$$

$$\kappa_6' = \hat{\mu}_{11}\mu_{22}^{s-s^*}\sum_{j=0}^{s^*-1}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-1-j}\sum_{j=0}^{B+2-s}\hat{\mu}_{11}^{j}\mu_{22}^{B+2-s-j}.$$

From equation (5.17), $\hat{\mu}_{22}\frac{d_1(s^*)}{\mu_{22}} \le \hat{\mu}_{12}\frac{d_2(s^*)}{\mu_{12}}$, and

$$\begin{aligned}
\epsilon(s,a_{10})' &\le \frac{\hat{\mu}_{22}}{\Theta_4(s^*)d_2(s^*)}\big(d_2(s^*)\kappa_5' - \frac{\mu_{12}}{\mu_{22}}d_1(s^*)\kappa_6'\big)\\
&= -\frac{\hat{\mu}_{22}}{d_2(s^*)}\mu_{12}\hat{\mu}_{11}^{B+3-s}\big(\hat{\mu}_{11}^{s^*-s}\hat{\mu}_{21}\sum_{j=0}^{s^*-2}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-2-j} + \mu_{12}^{s^*-1}\sum_{j=0}^{s-s^*}\hat{\mu}_{11}^{j}\mu_{22}^{s-s^*-j}\big)\\
&\le 0.
\end{aligned}$$

Next, for $s \in \{s^*+1,\dots,B+2\}$ and action a_{20},

$$\epsilon(s,a_{20})' = \frac{1}{\Theta_4(s^*)}(\hat{\mu}_{22}\kappa_7' - \hat{\mu}_{12}\kappa_8'),$$

where

$$\begin{aligned}
\kappa_7' &= \hat{\mu}_{21}\mu_{22}^{s-s^*}\sum_{j=0}^{B+1-s}\hat{\mu}_{11}^{j}\mu_{22}^{B+1-s-j}\big(\mu_{12}^{s^*}+\hat{\mu}_{11}\sum_{j=0}^{s^*-1}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-1-j}\big)\\
&\quad - \hat{\mu}_{11}\hat{\mu}_{21}^{s^*}\sum_{j=0}^{B+1-s^*}\hat{\mu}_{11}^{j}\mu_{22}^{B+1-s^*-j},\\
\kappa_8' &= \hat{\mu}_{11}\mu_{22}^{s-s^*}\sum_{j=0}^{s^*-1}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-1-j}\big(\mu_{22}^{B+2-s}+\hat{\mu}_{21}\sum_{j=0}^{B+1-s}\hat{\mu}_{11}^{j}\mu_{22}^{B+1-s-j}\big).
\end{aligned}$$

From equation (5.17), $\hat{\mu}_{22}\frac{d_1(s^*)}{\mu_{22}} \leq \hat{\mu}_{12}\frac{d_2(s^*)}{\mu_{12}}$, and

$$\begin{aligned}
\epsilon'(s, a_{20}) &\leq \frac{\hat{\mu}_{22}}{\Theta_4(s^*)d_2(s^*)}\big(d_2(s^*)\kappa'_7 - \frac{\mu_{12}}{\mu_{22}}d_1(s^*)\kappa'_8\big) \\
&= -\frac{\hat{\mu}_{22}}{d_2(s^*)}\mu_{12}\Big\{\hat{\mu}_{11}^{B+3-s}\big(\mu_{12}^{s^*-1}\sum_{j=0}^{s-s^*}\hat{\mu}_{11}^{j}\mu_{22}^{s-s^*-j} + \hat{\mu}_{11}^{s^*-s}\hat{\mu}_{21}\sum_{j=0}^{s^*-2}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-2-j}\big) \\
&+ (\hat{\mu}_{11} - \hat{\mu}_{21})\mu_{12}^{s^*-1}\mu_{22}^{s+1-s^*}\sum_{j=0}^{B+1-s}\hat{\mu}_{11}^{j}\mu_{22}^{B+1-s-j}\Big\} \\
&\leq 0.
\end{aligned}$$

Finally, for $s \in \{s^* + 1, \ldots, B + 2\}$ and action a_{21},

$$\epsilon'(s, a_{21}) = \frac{1}{\Theta_4(s^*)}\Big[f_2(s^* + 1) + (\hat{\mu}_{11}\mu_{12} - \hat{\mu}_{21}\mu_{22})\sum_{j=0}^{s^*-s-2}\mu_{22}^{j}\hat{\mu}_{11}^{B-s^*-j} \times \Upsilon'_2\Big], \quad (5.22)$$

where

$$\Upsilon'_2 = \hat{\mu}_{22}\big(\hat{\mu}_{11}\sum_{j=0}^{s^*-1}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-1-j} + \mu_{12}^{s^*}\big) - \hat{\mu}_{12}\hat{\mu}_{11}\sum_{j=0}^{s^*-1}\hat{\mu}_{21}^{j}\mu_{12}^{s^*-1-j}.$$

When $s^* = 0$, $\Upsilon'_2 = \hat{\mu}_{22} \geq 0$, $\epsilon'(s, a_{21}) \leq 0$ for $s \in \{1, \ldots, B + 2\}$; when $s^* > 0$, by the analysis in Lemma 5.4.2, we have $r \leq D(B + 2)$, and

$$r \leq D(B + 2) \Rightarrow \Upsilon'_2 \geq 0.$$

Thus, $\epsilon'(s, a_{21}) \leq 0$ for $s = s^* + 1, \ldots, B + 2$.

This proves that $\delta'_0(s) = \delta'_1(s)$ for all $s \in S$. Thus, by Theorem 9.5.1 of Puterman [41], the policy described in this theorem is optimal.

Note that, when $\hat{\mu}_{11} > \hat{\mu}_{21}$, $f_2(s^*) > 0$, $f_2(s^* + 1) < 0$, inequality (5.20) is strict for all $s \in S, a \in A_s \setminus \{\delta_0(s)\}$. The proof of uniqueness is similar to the proof of uniqueness for case (1), thus we omit for brevity. □

When $\hat{\mu}_{11} \geq \hat{\mu}_{21}$, and $\hat{\mu}_{11}\mu_{12} \leq \hat{\mu}_{21}\mu_{22}$, we have $\mu_{12} \leq \mu_{22}$. Thus, server 1 is better at station 1, server 2 is faster at station 2, and server 2 is more effective overall than server

1. The intuition of Theorem 5.4.2 is similar to our analysis of Theorem 5.4.1. Briefly speaking, when $s = 0$, only station 1 is working, we assign the server with higher successful service rate at station 1 to station 1. When $s > 0$, we assign the server with higher overall efficiency (i.e., server 2) to station 1 when the number of jobs in the system is small to push more jobs into the system, and we switch the assignment when the number of jobs in the system exceeds the threshold s^* to push more jobs out of the system. Again, to better understand the impact of the defect probabilities at station 2 on the value of s^*, we obtain the following corollary from Theorem 5.4.2 and equation (5.17).

Corollary 5.4.4. *When $\hat{\mu}_{11} \geq \hat{\mu}_{21}$ and $\hat{\mu}_{11}\mu_{12} \leq \hat{\mu}_{21}\mu_{22}$, Table 5.3 shows the optimal policy as a function of the value of $r = \frac{1-p_{12}}{1-p_{22}}$.*

Table 5.3: Optimal policy in case (2) as a function of r.

Range of r	Optimal Policy	a_{12} Optimal in States	a_{21} Optimal in States
$r \leq D(1)$	$(\delta_2^0)^\infty$	$0, 1, \ldots, B+2$	$\emptyset$
$D(1) < r \leq D(2)$	$(\delta_2^1)^\infty$	$0, 2, \ldots, B+2$	1
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$D(B+1) < r \leq D(B+2)$	$(\delta_2^{B+1})^\infty$	$0, B+2$	$1, \ldots, B+1$
$D(B+2) < r$	$(\delta_2^{B+2})^\infty$	0	$1, \ldots, B+2$

From Table 5.3 we observe that, the value of $s^* \in S_2^*$ lies in on the boundary of S when r is either small or high, and s^* is non-decreasing with respect to r. The intuition of Corollary 5.4.4 is similar to Corollary 5.4.2 except that now server 2 has higher overall efficiency than server 1, so we omit the interpretation for concision.

5.4.3 Special Cases

In this section, we provide conditions under which the optimal policy has a simple form. The following corollary shows that when servers 1 and 2 have higher successful service rates at stations 1 and 2, respectively, and server 2 has lower defect probability at station 2, then we will always assign the server to the station where they have higher successful service rates. This result is slightly more general than our conclusions in Theorem 5.3.1 when $N = 2$.

Corollary 5.4.5. *When $\hat{\mu}_{11} \geq \hat{\mu}_{21}, p_{12} \geq p_{22}$, and $\hat{\mu}_{22} \geq \hat{\mu}_{12}$, then it is optimal to always assign server 1 to station 1 and server 2 to station 2.*

Proof. We will show that the optimal policy is $(\delta_1^{B+2})^\infty$ or $(\delta_2^0)^\infty$. Therefore, when $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$, by Corollary 5.4.1 and Theorem 5.4.1, we need to prove that $S_1^* = \{B+2\}$; and when $\hat{\mu}_{11}\mu_{12} < \hat{\mu}_{21}\mu_{22}$, by Corollary 5.4.3 and Theorem 5.4.2, we need to prove that $S_2^* = \{0\}$.

When $\hat{\mu}_{11}\mu_{12} \geq \hat{\mu}_{21}\mu_{22}$, we can rewrite $f_1(B+2)$ as

$$f_1(B+2) = (p_{12} - p_{22})\mu_{12}\mu_{22}^{B+2} + (\hat{\mu}_{22} - \hat{\mu}_{12})\sum_{j=0}^{B+1} \hat{\mu}_{11}^j \mu_{22}^{B+1-j} \geq 0.$$

Thus, by Lemma 5.4.1 and Corollary 5.4.1, we have $S_1^* = \{B+2\}$ as desired.

When $\hat{\mu}_{11}\mu_{12} < \hat{\mu}_{21}\mu_{22}$, since $p_{12} \geq p_{22}$, we can obtain that

$$\hat{\mu}_{21}\hat{\mu}_{22} = \hat{\mu}_{21}\mu_{22}(1-p_{22}) < \hat{\mu}_{11}\mu_{12}(1-p_{12}) = \hat{\mu}_{11}\hat{\mu}_{12},$$

and we can reorganize $f_2(1)$ as

$$f_2(1) = -(\hat{\mu}_{21}\hat{\mu}_{22} - \hat{\mu}_{11}\hat{\mu}_{12})\hat{\mu}_{11}^{B+1} - \Big[(\hat{\mu}_{11} - \hat{\mu}_{21})\hat{\mu}_{22} + (p_{12} - p_{22})\mu_{12}\mu_{22}\Big]\sum_{j=0}^{B+1} \hat{\mu}_{11}^j \mu_{22}^{B+1-j} < 0.$$

Thus, by Lemma 5.4.2 and Corollary 5.4.3, we have $S_2^* = \{0\}$ as desired. □

Remark 5.4.1. *By Corollary 5.4.5, when $\hat{\mu}_{11} \geq \hat{\mu}_{21}$, $\mu_{12} \leq \mu_{22}$, and $p_{i,2} = p_2$, it is optimal to always assign server i to station i for $i = 1, 2$.*

The following remark shows that when the overall efficiencies among two servers are equal, the optimal policy is static.

Remark 5.4.2. *When $\hat{\mu}_{11} \geq \hat{\mu}_{21}$, $\hat{\mu}_{11}\mu_{12} = \hat{\mu}_{21}\mu_{22}$, we have $f_1(1) = f_1(B+2) = -f_2(1) = -f_2(B+2)$. Thus,*

(a) If $f_1(1) \leq 0$, then $0 \in S_1^*, B+2 \in S_2^*$, and Theorems 5.4.1, 5.4.2 imply that $(\delta_1^0)^\infty = (\delta_2^{B+2})^\infty$ is optimal, where $\delta_1^0 = (a_{12}, a_{21}, \ldots, a_{21})$;

(b) If $f_1(1) \geq 0$, then $B+2 \in S_1^*, 0 \in S_2^*$, and Theorems 5.4.1, 5.4.2 imply that $(\delta_2^0)^\infty = (\delta_1^{B+2})^\infty$ is optimal, where $\delta_2^0 = (a_{12}, a_{12}, \ldots, a_{12})$.

Note that

$$f_1(1) \geq 0 \Leftrightarrow r \leq C(1),$$

where

$$C(1) = 1 + \frac{(\hat{\mu}_{11} - \hat{\mu}_{21}) \sum_{j=0}^{B+1} \hat{\mu}_{21}^j \mu_{12}^{B+1-j}}{\sum_{j=0}^{B+2} \hat{\mu}_{21}^j \mu_{12}^{B+2-j}} \geq 1.$$

Moreover, when $\hat{\mu}_{11} \geq \hat{\mu}_{21}$, $\hat{\mu}_{11}\mu_{12} = \hat{\mu}_{21}\mu_{22}$, we have $\mu_{12} \leq \mu_{22}$. Then server 1 is better at station 1 while server 2 is faster at station 2. Thus, we would always assign server 1 to station 1 and server 2 to station 2 unless server 2 is significantly less reliable at station 2 relative to server 1 (in which case, the ratio of successful service probability of server 1 and server 2 is high). And if server 2 is not reliable at station 2 compared to server 1 (i.e., when $r > C(1) \geq 1$), we would assign server 1 to station 2 when station 2 is not starved.

Next, we consider the case when servers are reliable, i.e., when $p_{ij} = 0$ for $i, j = 1, 2$. Our results coincide with [31] in this case.

[31] considered the case when $p_{ij} = 0$ for $i, j = 1, 2$, and presented the optimal policy in two cases. First, they also labeled the servers such that $\mu_{11} \geq \mu_{21}$, which is equivalent to our assumption of $\hat{\mu}_{11} \geq \hat{\mu}_{21}$ when $p_{11} = p_{21} = 0$. When $\mu_{22} \geq \mu_{12}$, they proved that it is optimal to assign server 1 to station 1 and server 2 to station 2, which is equivalent to our conclusions in Remark 5.4.1. When $\mu_{22} < \mu_{12}$, they showed that the optimal policy is in the

form of Theorem 5.4.1, which also coincides with our conclusion since $\hat{\mu}_{11}\mu_{12} > \hat{\mu}_{21}\mu_{22}$ in this case, and our set of threshold S_1^* in Corollary 5.4.1 coincides with their set of thresholds as in Corollary 1 of [31].

5.5 Heuristic Policies for Longer Lines

In Section 5.4, we determined the optimal server assignment policy for tandem systems with two stations and two servers. The form of the optimal policy is already complex for the two stations system, and in our experience, the optimal policy for longer lines is likely to be even more complicated. Thus, in this section, we investigate the properties of the optimal policy and explore heuristic policies for systems with $N \geq 3$.

We refer a policy as non-idling if we always assign a server to each of the working stations. Theorems 5.4.1 and 5.4.2 indicate that for systems with two stations, there always exists a non-idling optimal policy. Moreover, by Proposition 1 of Işık, Andradóttir, and Ayhan [31], there exists a non-idling optimal policy for systems with arbitrary size when the defect probabilities are zero. It is a natural guess that there exists a non-idling optimal policy for systems with arbitrary size and general defect probabilities. However, we will show that this conjecture is wrong in the next section.

In Section 5.5.1, we discuss this non-idling property for larger systems, and identify that the optimal policy may not be non-idling when $N \geq 3$. In Section 5.5.2, we introduce the heuristic policies for longer lines, and evaluate their performance based on numerical results for three stations. Finally, in Section 5.5.3, we provide numerical results for our selected heuristics for systems with $N = 4$ and $N = 5$.

5.5.1 Non-idling vs. Idling

In this section, we show that the optimal policy is not necessary non-idling for systems with $N \geq 3$. This is in contrast to systems with $N = 2$ stations. Recall that Theorems 5.4.1 and 5.4.2 show that the optimal policy is non-idling when $N = 2$. Similarly, Proposition

5.3.2 states that there exist optimal policy that never idles the first station. However, the following example shows that when $N = 3$, there may not exist an optimal policy that is non-idling.

Example 5.5.1. *Consider the system with three stations, three servers, and $B_1 = B_2 = 0$. Recall that $\mu_{i,j}, p_{i,j}$ are the service rate and defect probability of server i at station j for $i, j = 1, 2, 3$, respectively. Suppose the service rates and defect probabilities are as in the following matrices, where the rows represent the server and the columns represent the station. For instance, $\mu_{1,2} = 1.2$ is listed on row 1 and column 2 of μ.*

$$\mu = \begin{bmatrix} 1.2 & 1.2 & 1.2 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad p = \begin{bmatrix} 0.1 & 0.1 & 0.1 \\ 0.7 & 0.7 & 0.7 \\ 0.9 & 0.9 & 0.9 \end{bmatrix}$$

Then, the optimal policy would idle server 3 at station 3 in state $s = (1, 1)$ and obtain a long-run average throughput of 0.3777, while the best non-idling policy has the same server assignment as in the optimal policy except that it assigns server i to $4 - i$ for $i = 1, 2, 3$ in state $s = (1, 1)$ and achieves a throughput of 0.3656. In conclusion, the best non-idling policy has a throughput about 3.2% lower than the throughput of the optimal policy.

Although from Example 5.5.1, we find that the best non-idling policy no longer guarantees a maximal throughput of larger systems, the difference between the throughputs of the optimal policy and the best non-idling policy is not significant. Table 5.4 provides the numerical results of the comparison of the best non-idling policy and the optimal policy. More specifically, we compute the average throughputs of the best non-idling policy and the optimal policy, and calculate the percentage of the deviation of the best non-idling policy from the optimal policy (optimality gap) for 10,000 iterations with $N = 3$, buffers randomly picked from integers $0, 1, \ldots, 10$, μ_{ij} drawn independently from a uniform distribution with range $(0, 1)$, and p_{ij} drawn independently from a uniform distribution with ranges $(0,1)$, $(0,0.5)$, $(0,0.1)$, and $(0,0.01)$. We also display 95% confidence intervals for the optimality gaps.

Table 5.4: Comparison of the Best Non-idling Policy and the Optimal Policy for Three Stations.

μ	p	Best Non-idle	Optimal	% Optimality Gap
$U(0,1)$	$U(0,1)$	0.1514	0.1533	1.22 ± 0.05
$U(0,1)$	$U(0,0.5)$	0.2851	0.2862	0.41 ± 0.02
$U(0,1)$	$U(0,0.1)$	0.4458	0.4459	0.03 ± 0.00
$U(0,1)$	$U(0,0.01)$	0.4882	0.4882	0.00 ± 0.00
Average		0.3426	0.3434	0.42

From Table 5.4, we observe that the average deviations of the best non-idling policy from the optimal policy for all ranges of defect probability are very small (less than 2%). Thus, we can conclude that the best non-idling policy is near-optimal (if not already optimal) for larger systems with $N \geq 3$. Recall that, by Proposition 1 of Işık, Andradóttir, and Ayhan [31], there exists a non-idling optimal policy when the servers are reliable (i.e., with zero defect probabilities). And from Table 5.4 we notice that the average deviations of the best non-idling policy from the optimal policy is decreasing (to zero) as the range and the value of defect probability becomes smaller, which confirms Işık, Andradóttir, and Ayhan's result in [31] numerically.

5.5.2 Heuristic Policies

In this section, we describe various heuristic server assignment policies and provide numerical results that suggest some of our heuristics are near-optimal for $N = 3$. For brevity, denote $q_{i,j} = 1 - p_{i,j}$ as the success probability for server i at station j, $i, j \in \{1, \ldots, N\}$ for the rest of this chapter.

First, we investigate the optimal server allocation policy of systems with infinite buffers between the stations (i.e., $B_1 = \cdots = B_{N-1} = \infty$). In this case, there is no blocking in the system, and the long-run average throughput of the system is determined by the bottleneck station. Moreover, the throughput of each station is determined by the minimum of the arrival rate and the departure rate of the station, and the arrival rate of the next station is the departure rate of jobs with no defects at the current station. Specifically, suppose μ_j, q_j are the service rate and success probability at station j. Then, the successful departure rate of station 1 is $\mu_1(1-p_1) = \mu_1 q_1$, which is also the arrival rate to station 2. Therefore, the departure rate of station 2 is $\min\{\mu_1 q_1, \mu_2\}$, and the successful departure rate of station 2 is $\min\{\mu_1 q_1, \mu_2\}q_2$. Proceeding in a similar manner, we can obtain the successful departure rate of station N, i.e., the long-run average throughput of the system, as

$$\min\{\mu_1 q_1 \cdots q_N, \mu_2 q_2 \cdots q_N, \ldots, \mu_N q_N\} = \min_{1\le j\le N}\{\mu_j \prod_{k=j}^{N} q_k\}.$$

Thus, the optimal stationary policy for the infinite buffers case is to assign server $i_j \in \{1,\ldots,N\}$ to station $j \in \{1,\ldots,N\}$ such that $\{i_1,\ldots,i_N\} = \{1,\ldots,N\}$ and

$$\min_{1\le j\le N}\{\mu_{i_j,j} \prod_{k=j}^{N} q_{i_k,k}\} \tag{5.23}$$

is maximized.

When the buffers are finite, we propose the following heuristic policy that is inspired by the optimal stationary policy for the infinite buffers case, and we refer to this heuristic as ‘Flow’.

- Flow: At any time t, let $J \subseteq \{1,\ldots,N\}$ be the set of working stations. Then assign server $i_j \in \{1,\ldots,N\}$ to station $j \in J$ such that $\cup_{j\in J}\{i_j\} = J$ and

$$\min_{j\in J}\{\mu_{i_j,j} \prod_{k\ge j, k\in J} q_{i_k,k}\}$$

is maximized.

Andradóttir, Ayhan, and Down [7] proved that for a Markovian queueing system with two tandem stations and two collaborative servers with no defects and additive combined service rates, it is optimal to assign the servers that maximize the product of the service rate of each station unless the system is blocked or starved. Combining this idea with the existence of the defect probability, and the fact that we would use the server with higher successful service rate at station 1 when the system is empty (see Proposition 5.3.1), we suggest the following heuristic policy (which we refer to as $\Pi\mu q$) that always maximizes the product of successful service rates of all the working stations.

- $\Pi\mu q$: At any time t, let $J \subseteq \{1, \dots, N\}$ be the set of working stations. Then assign server $i_j \in \{1, \dots, N\}$ to station $j \in J$ such that $\cup_{j \in J}\{i_j\} = J$ and

$$\prod_{j \in J} \mu_{i_j,j} q_{i_j,j}$$

  is maximized.

Note that, both *Flow* and $\Pi\mu q$ agree with Proposition 5.3.1 and Theorem 5.3.1. However, *Flow* and $\Pi\mu q$ are different from the optimal policy we characterized in Section 5.4 for two-station systems. Thus, we first check their performance for systems with $N = 2$. We compute the percentage of the deviation of the heuristic policies *Flow* and $\Pi\mu q$ from the optimal policy (optimality gap) for 10,000 iterations with buffers randomly picked from integers $0, 1, \dots, 10$, μ_{ij} drawn independently from a uniform distribution with range $(0, 1)$, and p_{ij} drawn independently from a uniform distribution with range $(0, 0.1)$. The average optimality gaps of *Flow* and $\Pi\mu q$ are 1.00% and 0.72%, respectively. Thus, *Flow* and $\Pi\mu q$ are near-optimal for two-station systems.

Before providing numerical results for systems with $N = 3$, we describe two auxiliary heuristic policies of $\Pi\mu q$, namely $\Pi\mu$ and Πq, to help us compare the effects of the service rate and the defect probability. Specifically, $\Pi\mu$ and Πq maximize the product of only service rates and only success probabilities of all the working stations, respectively.

- $\Pi\mu$: At any time t, let $J \subseteq \{1, \ldots, N\}$ be the set of working stations. Then assign server $i_j \in \{1, \ldots, N\}$ to station $j \in J$ such that $\cup_{j \in J}\{i_j\} = J$ and

  $$\prod_{j \in J} \mu_{i_j, j}$$

  is maximized.

- Πq: At any time t, let $J \subseteq \{1, \ldots, N\}$ be the set of working stations. Then assign server $i_j \in \{1, \ldots, N\}$ to station $j \in J$ such that $\cup_{j \in J}\{i_j\} = J$ and

  $$\prod_{j \in J} q_{i_j, j}$$

  is maximized.

Now, we are ready to obtain the numerical results for systems with $N = 3$ stations. We will compare heuristic policies *Flow*, $\Pi\mu q$, $\Pi\mu$, and Πq with two benchmark policies, namely the best stationary policy and the arbitrary stationary policy. We present numerical results for systems with three stations and randomly generated buffer sizes, service rates, and defect probabilities to investigate the performance of our heuristics. Specifically, we consider buffers randomly picked from integers $0, 1, \ldots, 10$, while μ_{ij} is drawn independently from a uniform distribution with ranges $(0, 1)$ and $(0, 10)$, and p_{ij} is drawn independently from a uniform distribution with ranges $(0, 1)$, $(0, 0.5)$, $(0, 0.1)$, and $(0, 0.01)$. We did 10,000 iterations for each of several pairs of different ranges of the service rates and defect probabilities, and compute the percentage of the deviation of the heuristic policy from the optimal policy. The results are shown in Table 5.5. Note that, in this section, the numbers in all tables are in percentages, and all tables display 95% confidence intervals.

Table 5.5: % Optimality Gap of *Flow*, $\Pi\mu q$, $\Pi\mu$, Πq and Static Policies for Three Stations (the minimum optimality gap in each row is shown in bold).

μ	p	Flow	$\Pi\mu q$	$\Pi\mu$	Πq	Arb. Static	Opt. Static
$U(0,1)$	$U(0,1)$	$\mathbf{8.12 \pm 0.23}$	13.69 ± 0.34	53.02 ± 0.64	30.27 ± 0.58	73.05 ± 0.58	25.53 ± 0.48
$U(0,10)$	$U(0,1)$	$\mathbf{8.12 \pm 0.23}$	13.69 ± 0.34	53.02 ± 0.64	30.27 ± 0.58	73.05 ± 0.58	25.53 ± 0.48
$U(0,1)$	$U(0,0.5)$	$\mathbf{4.26 \pm 0.11}$	6.95 ± 0.16	13.44 ± 0.25	41.00 ± 0.60	56.77 ± 0.58	13.99 ± 0.30
$U(0,10)$	$U(0,0.5)$	$\mathbf{4.26 \pm 0.11}$	6.95 ± 0.16	13.44 ± 0.25	41.00 ± 0.60	56.77 ± 0.58	13.99 ± 0.30
$U(0,1)$	$U(0,0.1)$	$\mathbf{2.11 \pm 0.07}$	2.33 ± 0.06	2.49 ± 0.06	48.55 ± 0.58	52.77 ± 0.61	10.23 ± 0.29
$U(0,10)$	$U(0,0.1)$	$\mathbf{2.11 \pm 0.07}$	2.33 ± 0.06	2.49 ± 0.06	48.55 ± 0.58	52.77 ± 0.61	10.23 ± 0.29
$U(0,1)$	$U(0,0.01)$	$\mathbf{1.99 \pm 0.07}$	$\mathbf{1.99 \pm 0.06}$	$\mathbf{1.99 \pm 0.06}$	50.09 ± 0.57	52.54 ± 0.61	9.95 ± 0.29
$U(0,10)$	$U(0,0.01)$	$\mathbf{1.99 \pm 0.07}$	$\mathbf{1.99 \pm 0.06}$	$\mathbf{1.99 \pm 0.06}$	50.09 ± 0.57	52.54 ± 0.61	9.95 ± 0.29
Average		**4.12**	6.24	17.74	42.48	58.78	14.93

From Table 5.5, we derive the following conclusions:

1. *Flow* performs strictly better than other policies for all cases except that when the defect probabilities are very small, $\Pi\mu q$ and $\Pi\mu$ are as good as *Flow* with respect to the mean of optimality gaps. Moreover, *Flow* is near-optimal since its deviation from the optimal policy is always under 10% even when the defect probabilities are unrealistically large (i.e., when $p \sim U(0,1)$).

2. $\Pi\mu q$ is always better than both Πq and $\Pi\mu$, since $\Pi\mu q$ contains the information of both the service rates and the defect probabilities, while Πq and $\Pi\mu$ only consider one of them.

3. When defect probabilities are large, Πq performs better than $\Pi\mu$, and when defect probabilities are small, $\Pi\mu$ performs better than Πq. Thus, we may focus on the defect probabilities when they are large, and focus on the service rates when the defect probabilities are small. When both μ and p are drawn from uniform distributions with the same magnitude and range (i.e., $U(0,1)$), Πq performs much better than $\Pi\mu$. Thus, the defect probability is more influential than the service rate.

4. The optimality gap for all the policies except for Πq decreases as the magnitude and range of possible defect probabilities decrease.

5. The magnitude change of μ does not impact the numerical results for all the policies we considered here. This is because we generate the scenarios (buffer size, service rates, and success probabilities) using common random numbers and hence choosing μ from $U(0, 10)$ is equivalent to choosing μ from $U(0, 1)$ and increasing the time unit by a factor of 10. From now on, we will only present the results for μ chosen from $U(0, 1)$.

6. The performance of dynamic policies *Flow* and $\Pi\mu q$ are significantly better than the best static policy in all the cases, the performance of $\Pi\mu$ is better than the best static policy when the defect probabilities are not large, and the performance of Πq is always worse than the best static policy.

7. The average performance of all dynamic policies are always strictly better than an arbitrary static policy. However, ignoring either the service rates or the defect probabilities (as in Πq and $\Pi\mu$) may result in an optimality gap as large as one corresponding to an arbitrary static policy.

8. The optimality gap of the best static policy is around 3.5 times that of *Flow* when the defect probability follows $U(0, 1)$, around 4.9 times that of *Flow* when the defect probability follows $U(0, 0.1)$, and around 5 times that of *Flow* when the defect probability follows $U(0, 0.01)$. And we observe the same pattern for $\Pi\mu q$. Thus, the relative performance of dynamic policies *Flow* and $\Pi\mu q$ over static policies are better when defect probability is small.

Note that, $\Pi\mu q$ performs better than *Flow* when the defect probability follows $U(0, 0.01)$ in terms of the variance of the optimality gaps, and *Flow* is the best when the defect probabilities are higher with larger range. Moreover, it is clear from equation (5.23) that the defect probability at later stations in the system has higher effect on our decision in *Flow*. For example, the success probability at station 3, i.e., $q_{i_3,3}$ appears in every term of the set we choose from, while the success probability at station 1, i.e., $q_{i_1,1}$ appears only in the first term. However, in $\Pi\mu q$, we put equal weight on the defect probability at all stations. Intuitively, the further downstream a job is in the system, the more efforts we have put on it, and the closer the job is to be completed. Thus, we want to have more reliable servers at stations closer to the end of the system to reduce the loss of our efforts and increase the probability of having a successful completed job. We now propose new heuristic policies based on this idea.

To further emphasize the defect probabilities in latter stations, we combine the two best heuristic policies, *Flow* and $\Pi\mu q$, as follows:

- $\Pi Flow$: At any time t, let $J \subseteq \{1, \dots, N\}$ be the set of working stations. Then assign server $i_j \in \{1, \dots, N\}$ to station $j \in J$ such that $\cup_{j \in J}\{i_j\} = J$ and

  $$\prod_{j \in J} \mu_{i_j,j} q_{i_j,j}^{j}$$

  is maximized.

In addition to $\Pi Flow$, we also propose another new heuristic policy $\Pi Flow^*$ that combines $\Pi\mu q$ and $\Pi Flow$ so that it put more weight on the defect probabilities in latter stations than $\Pi\mu q$, and less weight on the defect probabilities in latter stations than $\Pi Flow$.

- $\Pi Flow^*$: At any time t, let $J \subseteq \{1, \dots, N\}$ be the set of working stations, let J_1 be a subset of J such that, $j \in J_1$ if station $j \geq 2$ is working and station $j - 1$ is blocked. Denote $J_2 = J \setminus J_1$. Then assign server $i_j \in \{1, \dots, N\}$ to station $j \in J$ such that $\cup_{j \in J}\{i_j\} = J$ and

  $$\prod_{j \in J_1} \mu_{i_j,j} q_{i_j,j} \prod_{j \in J_2} \mu_{i_j,j} q_{i_j,j}^{j}$$

  is maximized.

Table 5.6 shows the numerical results of *Flow*, $\Pi\mu q$, $\Pi Flow$, and $\Pi Flow^*$. The choice of parameters are the same as in Table 5.5 except that we no longer present the results for μ chosen from $U(0, 10)$.

Table 5.6: $\%$ Optimality Gap of *Flow*, $\Pi\mu q$, $\Pi Flow$, and $\Pi Flow^*$ for Three Stations (the minimum optimality gap in each row is shown in bold).

μ	p	*Flow*	$\Pi\mu q$	$\Pi Flow$	$\Pi Flow^*$
$U(0,1)$	$U(0,1)$	8.12 ± 0.23	13.69 ± 0.34	6.51 ± 0.19	$\mathbf{5.61 \pm 0.15}$
$U(0,1)$	$U(0,0.5)$	4.26 ± 0.11	6.95 ± 0.16	4.34 ± 0.11	$\mathbf{3.83 \pm 0.09}$
$U(0,1)$	$U(0,0.1)$	$\mathbf{2.11 \pm 0.07}$	2.33 ± 0.06	2.25 ± 0.06	2.22 ± 0.06
$U(0,1)$	$U(0,0.01)$	$\mathbf{1.99 \pm 0.07}$	$\mathbf{1.99 \pm 0.06}$	$\mathbf{1.99 \pm 0.06}$	$\mathbf{1.99 \pm 0.06}$
Average		4.12	6.24	3.77	$\mathbf{3.41}$

From Table 5.6, we can derive the following conclusions:

1. $\Pi Flow^*$ performs best when the magnitude and range of defect probability are larger, i.e., when the defect probabilities follow $U(0, 1)$ and $U(0, 0.5)$. Its average deviation from the optimal policy is always under 6%. It is also the best heuristic policy on average.

2. *Flow* performs strictly better than others when the magnitude of defect probability is moderate, i.e., when the defect probabilities follow $U(0, 0.1)$.

3. When the magnitude of defect probability is small, i.e., when the defect probabilities follow $U(0, 0.01)$, all heuristics shown in Table 5.6 perform equally good with their average deviations from the optimal policy all under 2%.

4. $\Pi Flow^*$ always performs no worse than $\Pi Flow$ and $\Pi\mu q$.

In conclusion, *Flow* and $\Pi Flow^*$ are the two better heuristic policies. From this point on, we will focus on *Flow* and $\Pi Flow^*$.

Remark 5.5.1. *We have also tried other revised versions of* $\Pi Flow$*, but they are not as good as* $\Pi Flow^*$*, so we omit the numerical results for them. For example, we have tried a revised version of* $\Pi Flow^*$ *such that we put station* j *in set* J_1 *if either its preceding station* $j-1$ *is blocked or its subsequent station* $j+1$ *is starved. However, the optimality gap for this heuristic is worse than that of* $\Pi Flow^*$*.*

Since the idea of *Flow* is from infinite buffer systems, intuitively, the performance of *Flow* is related to the buffer size. Specifically, one would expect that the larger the buffer sizes are, the better the performance of *Flow* is. Hence, we obtain the numerical results for *Flow* and $\Pi Flow^*$ with different choices of buffer sizes. We choose buffers from $\{0, 5, 10\}$ with both balanced and unbalanced buffer allocation. We did 10,000 runs for each pair of buffer sizes with the service rates randomly generated from a uniform distribution on $(0, 1)$, and defect probabilities randomly generated from a uniform distribution on $(0, 0.5)$, $(0, 0.1)$, and $(0, 0.01)$. We did not consider defect probabilities in the range of $(0, 1)$ as defect probabilities above 50% seem impractical. The results are shown in Tables 5.7, 5.8, and 5.9, respectively.

Table 5.7: % Optimality Gaps of *Flow* and $\Pi Flow^*$ with Different Buffers for $N = 3, p \sim U(0, 0.5)$ (the minimum optimality gap in each row is shown in bold).

B_1	B_2	*Flow*	$\Pi Flow^*$	Arb. Static	Opt. Static
0	0	2.99 ± 0.08	$\mathbf{1.99 \pm 0.07}$	54.84 ± 0.56	14.67 ± 0.29
5	0	3.69 ± 0.10	$\mathbf{3.32 \pm 0.09}$	55.87 ± 0.57	14.32 ± 0.31
0	5	5.27 ± 0.11	$\mathbf{4.13 \pm 0.09}$	55.97 ± 0.56	15.31 ± 0.30
10	0	4.03 ± 0.11	$\mathbf{3.81 \pm 0.10}$	56.14 ± 0.58	14.44 ± 0.31
0	10	5.93 ± 0.12	$\mathbf{4.77 \pm 0.10}$	56.19 ± 0.56	15.57 ± 0.30
5	5	4.31 ± 0.11	$\mathbf{4.09 \pm 0.10}$	56.55 ± 0.58	13.90 ± 0.30
10	5	$\mathbf{4.36 \pm 0.11}$	$\mathbf{4.36 \pm 0.11}$	56.74 ± 0.58	13.81 ± 0.30
5	10	4.62 ± 0.11	$\mathbf{4.49 \pm 0.11}$	56.73 ± 0.58	14.02 ± 0.30
10	10	$\mathbf{4.50 \pm 0.12}$	4.58 ± 0.11	56.85 ± 0.58	13.80 ± 0.31
Average		4.41	**3.95**	56.21	14.43

Table 5.8: % Optimality Gaps of *Flow* and $\Pi Flow^*$ with Different Buffers for $N = 3, p \sim U(0, 0.1)$ (the minimum optimality gap in each row is shown in bold).

B_1	B_2	*Flow*	$\Pi Flow^*$	Arb. Static	Opt. Static
0	0	0.81 ± 0.03	$\mathbf{0.39 \pm 0.02}$	50.46 ± 0.59	10.74 ± 0.27
5	0	1.94 ± 0.06	$\mathbf{1.61 \pm 0.05}$	51.76 ± 0.60	10.72 ± 0.29
0	5	2.12 ± 0.06	$\mathbf{1.90 \pm 0.05}$	51.57 ± 0.60	10.72 ± 0.28
10	0	2.32 ± 0.07	$\mathbf{2.00 \pm 0.06}$	51.98 ± 0.60	10.80 ± 0.29
0	10	2.61 ± 0.07	$\mathbf{2.41 \pm 0.06}$	51.77 ± 0.60	10.83 ± 0.29
5	5	$\mathbf{2.14 \pm 0.07}$	2.42 ± 0.06	52.98 ± 0.61	10.34 ± 0.30
10	5	$\mathbf{2.31 \pm 0.07}$	2.74 ± 0.07	53.27 ± 0.61	10.42 ± 0.30
5	10	$\mathbf{2.34 \pm 0.07}$	2.78 ± 0.07	53.22 ± 0.61	10.42 ± 0.30
10	10	$\mathbf{2.39 \pm 0.07}$	2.98 ± 0.07	53.49 ± 0.61	10.41 ± 0.30
Average		**2.11**	2.14	52.28	10.60

Table 5.9: % Optimality Gaps of *Flow* and $\Pi Flow^*$ with Different Buffers for $N = 3, p \sim U(0, 0.01)$ (the minimum optimality gap in each row is shown in bold).

B_1	B_2	*Flow*	$\Pi Flow^*$	Arb. Static	Opt. Static
0	0	0.79 ± 0.03	$\mathbf{0.27 \pm 0.02}$	50.33 ± 0.59	10.54 ± 0.27
5	0	1.92 ± 0.06	$\mathbf{1.50 \pm 0.05}$	51.48 ± 0.60	10.44 ± 0.29
0	5	1.86 ± 0.06	$\mathbf{1.48 \pm 0.05}$	51.47 ± 0.60	10.38 ± 0.29
10	0	2.28 ± 0.07	$\mathbf{1.86 \pm 0.06}$	51.65 ± 0.60	10.47 ± 0.29
0	10	2.22 ± 0.07	$\mathbf{1.86 \pm 0.06}$	51.62 ± 0.60	10.39 ± 0.29
5	5	$\mathbf{2.06 \pm 0.07}$	2.17 ± 0.06	52.83 ± 0.61	10.10 ± 0.30
10	5	$\mathbf{2.22 \pm 0.07}$	2.46 ± 0.07	53.09 ± 0.62	10.17 ± 0.30
5	10	$\mathbf{2.21 \pm 0.07}$	2.47 ± 0.07	53.09 ± 0.62	10.16 ± 0.30
10	10	$\mathbf{2.27 \pm 0.07}$	2.66 ± 0.07	53.32 ± 0.62	10.16 ± 0.31
Average		1.98	$\mathbf{1.86}$	52.10	10.31

From Tables 5.7, 5.8, and 5.9 we can observe that:

1. The optimality gaps of both heuristic policies *Flow*, $\Pi Flow^*$ and both benchmark policies are increasing as the range of defect probability increases.

2. $\Pi Flow^*$ performs better when the buffer sizes are not large, *Flow* performs better when the buffer sizes are large, and this threshold of the buffer sizes becomes smaller when the range of defect probability drops from $U(0, 0.5)$ to $U(0, 0.1)$, and when the buffer allocation is more skewed.

3. When the sum of the buffers are the same, the performances of both *Flow* and $\Pi Flow^*$ get better as the buffer allocation is skewed to the left if the defect probabilities are of range $U(0, 0.5)$; if the defect probabilities are of ranges $U(0, 0.1)$ and $U(0, 0.01)$, as the buffer allocation becomes more balanced, the performance of *Flow* gets better while the performance of $\Pi Flow^*$ gets worse.

4. The deviation of $\Pi Flow^*$ from the optimal policy becomes larger as the buffer size increases.

5. *Flow* performs the best on average when the range of defect probability is moderate (i.e., when $p \sim U(0, 0.1)$); otherwise, $\Pi Flow^*$ performs the best on average. The performances of both *Flow* and $\Pi Flow^*$ are near-optimal with optimality gap less than 5% in all the cases.

Based on the previous numerical results, the conditions for either of $\Pi Flow^*$ or *Flow* be the best policy is determined by multiple factors including the buffer size, buffer allocation, service rates and defect probabilities. And the underneath pattern of how these factors impact the performance of our heuristics is hard to quantify. To further decrease the optimality gap of our heuristic from the optimal policy, we simply choose the better policy among $\Pi Flow^*$ and *Flow*, and refer to this new heuristic policy as *BoT*. By definition, *BoT* is always no worse than both of $\Pi Flow^*$ and *Flow*, and is plausible to be the best heuristic policy among all the policies we have discussed in this section.

In order to check the performance of *BoT*, we calculate the optimality gaps of *BoT* using the same parameters chosen and generated as in Table 5.6. For direct comparison, the results of *Flow*, $\Pi Flow^*$, and *BoT* are shown in Table 5.10.

Table 5.10: % Optimality Gap of *Flow*, $\Pi Flow^*$, and *BoT* for Three Stations (the minimum optimality gap in each row is shown in bold).

μ	p	*Flow*	$\Pi Flow^*$	*BoT*
$U(0,1)$	$U(0,1)$	8.12 ± 0.23	5.61 ± 0.15	$\mathbf{4.29 \pm 0.13}$
$U(0,1)$	$U(0,0.5)$	4.26 ± 0.11	3.83 ± 0.09	$\mathbf{2.76 \pm 0.07}$
$U(0,1)$	$U(0,0.1)$	2.11 ± 0.07	2.22 ± 0.06	$\mathbf{1.59 \pm 0.04}$
$U(0,1)$	$U(0,0.01)$	1.99 ± 0.07	1.99 ± 0.06	$\mathbf{1.45 \pm 0.04}$
Average		4.12	3.41	2.52

From Table 5.10, we observe that *BoT* significantly improves the performance of *Flow* and $\Pi Flow^*$, and maintains optimality gaps less than 5% for all the cases. Combining the numerical results of Table 5.6 and Table 5.10, ***BoT*** is the best heuristic policy for all cases we discussed for systems with three stations. Thus, we can conclude that heuristic policy ***BoT*** performs near-optimal when $N = 3$. Note that, although ***BoT*** performs the best among all the heuristics, it is harder to apply than the other two near-optimal heuristics *Flow* and $\Pi Flow^*$.

In the next section, we will further validate the performance of *Flow*, $\Pi Flow^*$, and ***BoT*** for systems with $N \geq 4$.

5.5.3 Numerical Results for Systems with More Than Three Stations

In this section, we focus on the two best "pure" heuristic policies *Flow*, and $\Pi Flow^*$, and one "mixed" heuristic policy ***BoT***. We will provide the numerical results of these heuristics for systems with four and five stations.

Tables 5.11 and 5.12 show numerical results for the heuristics *Flow*, $\Pi Flow^*$, and ***BoT*** with two benchmarks, i.e., the best stationary and the arbitrary stationary policies for four stations and five stations, respectively. Similar to the three stations cases, we consider μ_{ij}, p_{ij} drawn independently from a uniform distribution as indicated in the tables. Due to the computational difficulties of finding the optimal policy as the size of the system increases, buffers are now randomly picked from integers 0 to 5 for systems with $N = 4$, and 0 to 1 for systems with $N = 5$. We did 10,000 iterations with common random number generator for systems with $N = 4$ and $N = 5$, and compute the percentage of the deviation of the heuristic policies and the benchmark policies from the optimal policy.

Table 5.11: % Optimality Gaps of *Flow*, $\Pi Flow^*$, and *BoT* for Four Stations (the minimum optimality gap in each column is shown in bold).

μ	p	*Flow*	$\Pi Flow^*$	*BoT*	Arb. Static	Opt. Static
$U(0,1)$	$U(0,1)$	10.67 ± 0.20	7.79 ± 0.14	$\mathbf{6.22 \pm 0.12}$	86.86 ± 0.37	31.71 ± 0.44
$U(0,1)$	$U(0,0.5)$	6.69 ± 0.11	5.22 ± 0.09	$\mathbf{4.12 \pm 0.07}$	69.78 ± 0.45	17.51 ± 0.26
$U(0,1)$	$U(0,0.1)$	2.62 ± 0.05	2.26 ± 0.04	$\mathbf{1.73 \pm 0.03}$	63.24 ± 0.52	10.52 ± 0.23
$U(0,1)$	$U(0,0.01)$	2.38 ± 0.05	1.71 ± 0.04	$\mathbf{1.36 \pm 0.03}$	62.86 ± 0.53	9.91 ± 0.24
Average		5.59	4.25	$\mathbf{3.36}$	70.69	17.41

Table 5.12: % Optimality Gaps of *Flow*, $\Pi Flow^*$, and *BoT* for Five Stations (the minimum optimality gap in each column is shown in bold).

μ	p	*Flow*	$\Pi Flow^*$	*BoT*	Arb. Static	Opt. Static
$U(0,1)$	$U(0,1)$	12.52 ± 0.19	8.85 ± 0.13	$\mathbf{7.48 \pm 0.11}$	93.41 ± 0.22	35.27 ± 0.39
$U(0,1)$	$U(0,0.5)$	8.42 ± 0.11	5.76 ± 0.08	$\mathbf{4.96 \pm 0.07}$	77.04 ± 0.35	19.85 ± 0.23
$U(0,1)$	$U(0,0.1)$	2.57 ± 0.03	1.58 ± 0.03	$\mathbf{1.40 \pm 0.02}$	67.85 ± 0.45	9.99 ± 0.16
$U(0,1)$	$U(0,0.01)$	2.02 ± 0.03	0.80 ± 0.02	$\mathbf{0.74 \pm 0.02}$	67.33 ± 0.46	8.91 ± 0.17
Average		6.38	4.25	$\mathbf{3.65}$	76.41	18.51

Comparing the numerical results in Tables 5.11 and 5.12 with the numerical results for three stations system (Tables 5.5 and 5.6), we observe that:

1. The optimality gaps for all heuristic policies increase as the number of stations increases when the range of defect probability are wide (i.e., $U(0,1)$ and $U(0,0.5)$). Intuitively, all our heuristic policies are non-idling. However, the optimal policy may idle the server with high defect probabilities (as seen in Example 5.5.1), and we suspect that the gaps between the best non-idling policy and the optimal policy increase as the range of defect probability becomes wider and the size of the system becomes larger. Thus, the increasing optimality gaps for our heuristic policies may be a result of an increasing gap between the best non-idling policy and the optimal policy as the size of the system increases.

2. *BoT* performs the best in all the cases, and its average deviation from the optimal policy is under 7.5%. Moreover, *BoT* maintains the smallest variance on the average optimality gap among all the policies as the number of stations increases, which indicates the stability of its performance.

3. For four and five stations systems, the average performance of $\Pi Flow^*$ is better than *Flow* in all the cases, while for three stations system, $\Pi Flow^*$ is worse than *Flow* when the magnitude of the defect probability is moderate (i.e., when it follows $U(0, 0.1)$). Comparing to three stations systems, we have more stations in tandem and consider smaller range of the buffer sizes for the numerical results of four and five stations, and both of these two changes would lead to a higher probability of having blocked stations. Recall that *Flow* is optimal when there is no blocking, so this worse performance of *Flow* when $N = 4$ than $N = 3$ may be caused by more stations and smaller range of buffer size choices.

In conclusion, for longer lines, *BoT* is the best "mixed" heuristic policy, and $\Pi Flow^*$ is the best "pure" heuristic policy on average.

5.6 Conclusions

In practice, it is common to have defective jobs when they are being processed by the servers, and the defect probabilities depend on the proficiency of the server. However, most of the existing papers that study queueing systems with flexible servers assumed that the defect probabilities are zero. Other existing papers address defects but assume that part of the defective jobs can be fixed, and focus on the planning and control of rework in a production system. We investigated the optimal server allocation problem with flexible and error-prone servers. In particular, for a queueing system with N tandem stations, infinite supply in front of the first station, finite intermediate buffers, and N flexible but non-collaborative servers, we considered the server allocation policy that maximizes the long-run average throughput of the system in the presence of defects.

For Markovian systems with two stations and two servers, we characterized the optimal policy with respect to which server has the higher effectiveness overall. Specifically, we proved that when the system is empty, we should assign the server with higher successful service rate at station 1 to station 1; but when station 2 is working, we would assign the server that is more effective overall to station 1 when the number of jobs in the buffer is small, and assign this server to station 2 when the buffers are crowded.

For larger Markovian systems, we provided a partial characterization of the optimal policy through sample path analysis, and proved that when a distinct server is the fastest and most reliable at each station, the optimal policy always assign the server to the station where they are fastest and most reliable. Furthermore, we showed that the server at the first station should never be idled. Next, we analyzed the best static server assignment when the buffers are infinite, and developed heuristic policies based on the previous work. Using the insights gained from the numerical results of the heuristic policies under variate scenarios for three stations systems, we revised our heuristic policies and proposed new heuristic policies that integrated the advantages of the original heuristic policies. As a result, we finalized two "pure" and one "mixed" heuristic policies that are easy to implement and performed to be near-optimal for three stations systems. And the numerical results for systems with four and five stations further validate the near-optimal performances of these three heuristic policies.

CHAPTER 6
SUMMARY AND FUTURE RESEARCH

6.1 Summary

This dissertation focused on the optimal control of manufacturing and service systems through dynamic allocation of cross-trained servers. For a multi-server tandem queueing system with finite intermediate buffers and infinite supply in front of the first station, we explored the server assignment policy that maximizes the long-run average throughput of the system. The specific systems that we discussed are commonly seen in practice but not widely discussed in the literature.

In Chapter 3, we analyzed the server allocation problem when each job can be decomposed into multiple subtasks and there are no precedence relationships among the subtasks within each station. We first characterized the optimal static, flexible, and collaborative task assignment approaches, and further inspected the optimal policies for two special cases, namely when buffers are zero and when the sum of the buffers goes to infinity. Comparing the task assignment approaches with three other server coordination methods, namely teamwork with or without task partitioning and non-collaboration, we concluded that task assignment is preferable when the servers are highly specialized; otherwise, teamwork or non-collaboration are preferable depending on whether the synergy level among the servers is high or not. To better capture the properties of these server coordination methods, we further investigated two cases when the servers are generalists or specialists. The numerical results showed that, when the servers are generalists, we prefer non-collaboration, and then teamwork without task partitioning as the synergy level among servers goes from low to high; and when the servers are specialists, we prefer collaborative task assignment if the servers are highly specialized, otherwise, we prefer teamwork with task partitioning if the synergy level is high, non-collaboration if the synergy level is moderate or low.

In Chapter 4, we studied server allocation problem in terms of teams when the servers are flexible and collaborative. Unlike most of the existing papers that assumed a fixed synergy level when the servers collaborate, we focused on the service rate of the teams without providing a specific formation of the team service rates with respect to any other factors. We exhibited sufficient criteria for eliminating teams that are not on the Pareto boundary or be dominated by other teams, then we present the optimal policy among the remaining teams, which we referred to as the optimal assignment set, for systems with two stations. We verified that the optimal policy has monotone thresholds on the number of jobs in the buffer for teams in the optimal assignment set. Then we validated our optimal policy by applying it to two special cases: proportional team service rates and teams of specialized servers. Motivated by the optimal policy of systems with two stations, we proposed heuristic policy for larger systems with teams of specialized servers when they servers are generalists. The numerical results suggested that our heuristic policy performed near-optimal.

In Chapter 5, we address the fact that jobs may incur damage and be wasted when being processed by the servers. However, most existing papers ignore the possibility of defects when studying queueing systems with flexible servers. As far as we are aware, this is the first work to consider the server allocation problem in the presence of defects. For Markovian systems with two stations and two servers, we demonstrated that the optimal policy is either a single or a double threshold policy on the number of jobs in the buffer. Specifically, we proved that when the system is empty, we should assign the server with higher successful service rate at station 1 to station 1; but when station 2 is working, we would assign the server that is more effective overall to station 1 when the number of jobs in the buffer is small, and assign this server to station 2 when the buffers are crowded. A partial characterization of the optimal policy is given for longer lines. We proved that when a distinct server is the fastest and most reliable at each station, the optimal policy always assigns the server to the station where they are fastest and most reliable. Furthermore, we would never idle the server assigned to station 1, but we might idle the servers at other working stations for systems with more than two stations. We also presented and compared several heuristic policies that are easy to implement and performed well for larger systems.

6.2 Future Research Directions

In this section, we present potential extensions of the problems we studied in this dissertation.

For the problem in Chapter 3, we mainly focus on task assignment approaches when a job can be decomposed into two subtasks at each station. However, as is given in the comments prior to Proposition 3.1.1, some of our results may no longer hold when the number of subtasks exceeds two. Therefore, we expect to explore more about task assignment approaches when a job can be decomposed into more than two subtasks with no precedence relationships.

Another future research direction we propose here is based on our findings in Chapter 4. For teams of specialized servers where the servers are generalists, we suspect that the optimal policy always uses permanent teams that are formed based on their ability. More specifically, for systems with N tandem stations and N servers of each type, we will use the following N teams: the best server of each type, the second best server of each type,..., the worst server of each type. We have proved this result for systems with two stations (as is given in Remark 4.3.2.2), and the numerical results for three stations systems (as in Examples 4.4.1, 4.4.2, and 4.4.3) also support this conjecture. Thus, we plan to prove this result for systems with arbitrary number of stations.

For the systems in the presence of defects in Chapter 5, we focus on maximizing the long-run average throughput of the system. However, in production systems, when the cost of raw materials is not trivial, it is natural to include a penalty cost of wasted raw materials (jobs). Therefore, our first research direction is to consider the cost minimization problem of these systems with variable costs, such as a penalty cost for defects.

Another promising extension related to the problem in Chapter 5 is motivated by our results in Section 5.5.1. For systems with more than two stations, the optimal policy may involve server idling. More specifically, as is shown in Example 5.5.1, when a server is very unreliable at every station, the optimal policy may always let the server idle. Thus, it is meaningless to hire this server in the first place. For this reason, we suspect that there exists a threshold on the defect probability over which we will choose not to hire a server, and we are interested in finding or characterizing this threshold in our future research.

The future research directions we proposed so far are closely related to the problems we discussed in this thesis. For a broader research agenda, we strive to continue our research on variant scenarios in the field of optimal allocation of servers to optimize system performance. Towards this end, we consider a problem motivated by issues in healthcare settings. In the emergency departments, the health conditions of the patients may deteriorate while waiting to be treated or when being treated. We start by considering a single server queueing system with multiple classes of customers that can change type (e.g., due to deterioration in health condition) or abandon the system (e.g., due to death or recovery) before his treatment is completed. Optimal control of queueing systems with customer abandonments have been discussed in the literature. For instance, Down, Koole, and Lewis [22] studied the dynamic server control of a single-server system with abandonments. Optimal control of queueing systems with customers that can change status is scarcely discussed (see Cao and Xie [20] and Zayas-Cabán and Ahn [52]). Moreover, neither of [20] or [52] completely characterized the optimal server assignment policy even for systems with two types of jobs. Both of these papers provided partial description of the optimal policy, and [52] proposed conjecture of the optimal policy for cases that are not solved in their paper. To the best of our knowledge, no existing paper has fully solved the optimal server allocation policy for a single-server system with customers that can change type and abandon the system. Therefore, this is one potential research direction that we intend to explore.

Appendices

APPENDIX A
APPENDICES FOR CHAPTER 3

In the appendices, we provide supplementary materials for Chapter 3. In Appendix A.1, we provide the proof of Proposition 3.2.1. In Appendix A.2, we provide proofs for results in Section 3.3. In Appendix A.3, we provide proofs of results in Section 3.5. In Appendix A.4, we present the comparisons of server coordination methods that are not included in Section 3.5. In Appendix A.5, we provide model descriptions for the two teamwork approaches.

A.1 Proof of Proposition 3.2.1

Let

$$\begin{aligned} f_1 &= \mu_{22} - \mu_{12}; \\ f_2 &= \sum_{k=0}^{B_1+B_2+1} \mu_{11}^k \mu_{22}^{B_1+B_2+2-k} \Big(\mu_{11} \sum_{j=0}^{B_1+B_2+1} \mu_{21}^j \mu_{12}^{B_1+B_2+1-j} + \mu_{12}^{B_1+B_2+2} \Big) \\ &\quad - \sum_{k=0}^{B_1+B_2+1} \mu_{21}^k \mu_{12}^{B_1+B_2+2-k} \sum_{j=0}^{B_1+B_2+2} \mu_{11}^j \mu_{22}^{B_1+B_2+2-j}; \\ f_3 &= \sum_{k=0}^{B_1+B_2+1} \mu_{21}^k \mu_{12}^{B_1+B_2+1-k} \sum_{j=0}^{B_1+B_2} \mu_{11}^{j+1} \mu_{22}^{B_1+B_2+1-j} \\ &\quad - \sum_{k=0}^{B_1+B_2} \mu_{21}^{k+1} \mu_{12}^{B_1+B_2+1-k} \sum_{j=0}^{B_1+B_2+1} \mu_{11}^j \mu_{22}^{B_1+B_2+1-j}. \end{aligned} \tag{A.1}$$

Then, $f_1 \propto T_{12}^f - T_{12}^{1f}$, $f_2 \propto T_{12}^f - T_{21}^{1f}$, and $f_3 \propto T_{12}^{1f} - T_{21}^{1f}$.

If we treat $B_1 + B_2$ as variable and the service rates as given, then we prove by induction that $f_2(B_1+B_2) = L(B_1+B_2)$, where

$$\begin{aligned} L(B_1+B_2) = (\mu_{11}-\mu_{21}) &\sum_{k=0}^{B_1+B_2} \mu_{12}^{k+1}\mu_{21}^{B_1+B_2-k} \sum_{j=0}^{k} \mu_{11}^{j}\mu_{22}^{B_1+B_2+2-j} \\ + (\mu_{22}-\mu_{12}) &\sum_{k=0}^{B_1+B_2+1} \mu_{11}^{k+1}\mu_{22}^{B_1+B_2+1-k} \sum_{j=0}^{k} \mu_{12}^{j}\mu_{21}^{B_1+B_2+1-j}. \end{aligned} \tag{A.2}$$

Note that,

$$\begin{aligned} L(n) &= (\mu_{11}-\mu_{21})\Big(\sum_{k=0}^{n-1} \mu_{12}^{k+1}\mu_{21}^{n-k} \sum_{j=0}^{k} \mu_{11}^{j}\mu_{22}^{n+2-j} + \mu_{12}^{n+1}\sum_{j=0}^{n} \mu_{11}^{j}\mu_{22}^{n+2-j}\Big) \\ &+ (\mu_{22}-\mu_{12})\Big(\sum_{k=0}^{n} \mu_{11}^{k+1}\mu_{22}^{n+1-k} \sum_{j=0}^{k} \mu_{12}^{j}\mu_{21}^{n+1-j} + \mu_{11}^{n+2}\sum_{j=0}^{n+1} \mu_{12}^{j}\mu_{21}^{n+1-j}\Big) \\ &= \mu_{21}\mu_{22}L(n-1) + (\mu_{11}-\mu_{21})\mu_{12}^{n+1}\sum_{j=0}^{n} \mu_{11}^{j}\mu_{22}^{n+2-j} + (\mu_{22}-\mu_{12})\mu_{11}^{n+2}\sum_{j=0}^{n+1} \mu_{12}^{j}\mu_{21}^{n+1-j}. \end{aligned} \tag{A.3}$$

When $B_1+B_2=0$,

$$\begin{aligned} f_2(0) &= (\mu_{22}^2+\mu_{11}\mu_{22})\big[\mu_{11}(\mu_{21}+\mu_{12})+\mu_{12}^2\big] - (\mu_{12}^2+\mu_{21}\mu_{12})\big(\mu_{22}^2+\mu_{11}\mu_{22}+\mu_{11}^2\big) \\ &= (\mu_{11}-\mu_{21})\mu_{12}\mu_{22}^2 + (\mu_{22}-\mu_{12})\mu_{11}\big[\mu_{11}(\mu_{21}+\mu_{12})+\mu_{22}\mu_{21}\big] \\ &= L(0). \end{aligned}$$

Suppose now that $f_2(B_1+B_2) = L(B_1+B_2)$ for $B_1+B_2 = 0, 1, \ldots, n-1$. Then, for

$B_1 + B_2 = n$, (A.1) yields

$$\begin{aligned}
f_2(n) &= \sum_{k=0}^{n+1} \mu_{11}^k \mu_{22}^{n+2-k} \Big(\mu_{11} \sum_{j=0}^{n+1} \mu_{12}^j \mu_{21}^{n+1-j} + \mu_{12}^{n+2} \Big) - \sum_{k=0}^{n+1} \mu_{12}^{k+1} \mu_{21}^{n+1-k} \sum_{j=0}^{n+2} \mu_{11}^j \mu_{22}^{n+2-j} \\
&= \Big(\mu_{11}^{n+1}\mu_{22} + \sum_{k=0}^{n} \mu_{11}^k \mu_{22}^{n+2-k} \Big) \Big[\mu_{21}\big(\mu_{11} \sum_{j=0}^{n} \mu_{12}^j \mu_{21}^{n-j} + \mu_{12}^{n+1}\big) + (\mu_{11} + \mu_{12} - \mu_{21})\mu_{12}^{n+1} \Big] \\
&- \Big(\mu_{12}^{n+2} + \sum_{k=0}^{n} \mu_{12}^{k+1} \mu_{21}^{n+1-k} \Big) \Big(\mu_{11}^{n+2} + \sum_{j=0}^{n+1} \mu_{11}^j \mu_{22}^{n+2-j} \Big) \\
&= \mu_{21}\mu_{22} f_2(n-1) + \mu_{11}^{n+1}\mu_{22} \Big(\mu_{11} \sum_{j=0}^{n+1} \mu_{12}^j \mu_{21}^{n+1-j} + \mu_{12}^{n+2} \Big) \\
&+ (\mu_{11} + \mu_{12} - \mu_{21}) \mu_{12}^{n+1} \sum_{k=0}^{n} \mu_{11}^k \mu_{22}^{n+2-k} - \mu_{12}^{n+2} \sum_{j=0}^{n+2} \mu_{11}^j \mu_{22}^{n+2-j} - \mu_{11}^{n+2} \sum_{k=0}^{n} \mu_{12}^{k+1} \mu_{21}^{n+1-k} \\
&= \mu_{21}\mu_{22} L(n-1) + (\mu_{11} - \mu_{21}) \mu_{12}^{n+1} \sum_{j=0}^{n} \mu_{11}^j \mu_{22}^{n+2-j} + (\mu_{22} - \mu_{12}) \mu_{11}^{n+2} \sum_{j=0}^{n+1} \mu_{12}^j \mu_{21}^{n+1-j} \\
&= L(n),
\end{aligned}$$

where we have used (A.3). Thus, $f_2 = L(B_1 + B_2)$, and from now on, we use equation (A.2) as the expression for f_2.

When $\mu_{22} \geq \mu_{12}$, we have $f_1 \geq 0, f_2 \geq 0$, which implies that $T_{12}^f \geq T_{12}^{1f}, T_{12}^f \geq T_{21}^{1f}$, and hence assignment A_{12}^f is optimal.

When $\mu_{22} < \mu_{12}, \mu_{11} = \mu_{21} > 0$, we have $f_2 \leq 0$, and f_3 can be simplified as follows:

$$f_3 = \mu_{21}^{B_1+B_2+2} \sum_{k=0}^{B_1+B_2} \mu_{21}^k (\mu_{22}^{B_1+B_2+1-k} - \mu_{12}^{B_1+B_2+1-k}) \leq 0.$$

Thus, $T_{12}^f \leq T_{21}^{1f}, T_{12}^{1f} \leq T_{21}^{1f}$, and hence assignment A_{21}^{1f} is optimal, which corresponds to $\mu_{22}^* = \mu_{12}$.

When $\mu_{22} < \mu_{12}, \mu_{11} > \mu_{21} = 0$, then $f_1 < 0$, which implies that $T_{12}^f < T_{12}^{1f}$. Moreover, when $\mu_{21} = 0$, f_3 can be simplified as follows:

$$f_3 = \mu_{12}^{B_1+B_2+1} \sum_{j=0}^{B_1+B_2} \mu_{11}^{j+1} \mu_{22}^{B_1+B_2+1-j} > 0.$$

Therefore, $T_{12}^{1f} > T_{21}^{1f}$, and assignment A_{12}^{1f} is optimal (i.e., $\mu_{22}^{*} = 0$).

Finally, when $\mu_{22} < \mu_{12}, \mu_{11} > \mu_{21} > 0$, we have $f_1 < 0$ and hence $T_{12}^{f} < T_{12}^{1f}$. Therefore, we only need to compare T_{12}^{1f} and T_{21}^{1f} (i.e., determine the sign of f_3). If we treat μ_{22} as variable and $\mu_{11}, \mu_{12}, \mu_{21}$ as given, then

$$\begin{aligned} \frac{\partial f_3(\mu_{22})}{\partial \mu_{22}} = & \sum_{j=0}^{B_1+B_2} (B_1 + B_2 + 1 - j) \mu_{11}^{j} \mu_{22}^{B_1+B_2-j} \times \\ & \Big(\sum_{k=0}^{B_1+B_2+1} \mu_{21}^{k} \mu_{12}^{B_1+B_2+1-k} (\mu_{11} - \mu_{21}) + \mu_{21}^{B_1+B_2+2} \Big) > 0. \end{aligned} \tag{A.4}$$

That is, $f_3(\mu_{22})$ is increasing with respect to μ_{22}. Moreover,

$$\begin{aligned} f_3(0) &= -\mu_{11}^{B_1+B_2+1} \sum_{k=0}^{B_1+B_2} \mu_{21}^{k+1} \mu_{12}^{B_1+B_2+1-k} < 0, \\ f_3(\mu_{12}) &= \sum_{k=0}^{B_1+B_2+1} \mu_{21}^{k} \mu_{12}^{B_1+B_2+1-k} \sum_{j=0}^{B_1+B_2} \mu_{11}^{j+1} \mu_{12}^{B_1+B_2+1-j} \\ &\quad - \sum_{k=0}^{B_1+B_2} \mu_{21}^{k+1} \mu_{12}^{B_1+B_2+1-k} \sum_{j=0}^{B_1+B_2+1} \mu_{11}^{j} \mu_{12}^{B_1+B_2+1-j} \\ &= \sum_{k=0}^{B_1+B_2+1} \mu_{21}^{k} \mu_{12}^{B_1+B_2+1-k} \Big(\sum_{j=0}^{B_1+B_2+1} \mu_{11}^{j} \mu_{12}^{B_1+B_2+2-j} - \mu_{12}^{B_1+B_2+2} \Big) \\ &\quad - \Big(\sum_{k=0}^{B_1+B_2+1} \mu_{21}^{k} \mu_{12}^{B_1+B_2+2-k} - \mu_{12}^{B_1+B_2+2} \Big) \sum_{j=0}^{B_1+B_2+1} \mu_{11}^{j} \mu_{12}^{B_1+B_2+1-j} \\ &= \mu_{12}^{B_1+B_2+2} \sum_{k=1}^{B_1+B_2+1} (\mu_{11}^{k} - \mu_{21}^{k}) \mu_{12}^{B_1+B_2+1-k} > 0. \end{aligned}$$

Thus there exist a unique $\mu_{22}^{*} \in (0, \mu_{12})$ such that $f_3(\mu_{22}^{*}) = 0$. Since $f_3(\cdot)$ is increasing with respect to μ_{22}, we have $T_{12}^{1f} \geq T_{21}^{1f}$ if and only if $\mu_{22} \geq \mu_{22}^{*}$. Replacing μ_{22} with

x, multiplying both sides of the equation $f_3(x) = 0$ with $(x - \mu_{11})$, and using (4.3), we can obtain equation (3.4). And the fact that we multiply the equation with $(x - \mu_{11})$ adds one more root (i.e., μ_{11}) to equation (3.4). Thus, μ_{11} and μ_{22}^* are the only positive roots of equation (3.4). □

A.2 Proofs for Section 3.3

We provide the proofs of Propositions 3.3.3 and 3.3.5 in Appendices A.2.1 and A.2.2, respectively.

A.2.1 Proof of Proposition 3.3.3

When $\beta_1 \geq \beta_2 > \frac{1}{2}$, $x_1 = \max\{\beta_1, \alpha\}\Sigma_1, x_2 = \max\{\beta_2, \alpha\}\Sigma_2$, and $\mu_{11}\mu_{12} > \mu_{21}\mu_{22}$. If $\gamma = m_1$, i.e., $\mu_{11} + \mu_{22} = \mu_{21} + \mu_{12}$, then Corollary 3.3.1 implies that A_{12}^c is optimal if and only if

$$\max\{\beta_1, \alpha\} \leq \max\{\beta_2, \alpha\}\gamma.$$

Note that since $\beta_1 \geq \beta_2$, we have $m_1 \geq \frac{\beta_1}{\beta_2} \geq 1$. Then,

$$\max\{\beta_2, \alpha\}\gamma \geq \max\{\beta_2, \alpha\}\frac{\beta_1}{\beta_2} = \max\{\beta_1, \alpha\frac{\beta_1}{\beta_2}\} \geq \max\{\beta_1, \alpha\}.$$

Thus, when $\gamma = m_1$, A_{12}^c is always no worse than A_{21}^c.

If $\gamma \neq m_1$,

1. If $\alpha \geq \beta_1$, equation (3.17) would be

$$\alpha\big[(2\beta_1 - 1) - (2\beta_2 - 1)\gamma\big] \geq (1 - \gamma)(\beta_1 + \beta_2 - 1).$$

   Then, A_{12}^c is optimal if and only if either

$$\alpha \geq G_1, \gamma < m_1, \tag{A.5}$$

or

$$\alpha \leq G_1, \gamma > m_1. \tag{A.6}$$

2. If $\beta_1 > \alpha \geq \beta_2$, equation (3.17) would be

$$\alpha(2\beta_1 - 1)[\beta_1 + (1-\beta_2)\gamma] \geq (\beta_1 + \beta_2 - 1)\beta_1.$$

   Then A_{12}^c is optimal if and only if

$$\alpha \geq G_2. \tag{A.7}$$

3. If $\beta_1 > \beta_2 > \alpha$, equation (3.17) would be

$$(2\beta_1 - 1)(1-\beta_2)\beta_2\gamma \geq (2\beta_2 - 1)(1-\beta_1)\beta_1.$$

   Then A_{12}^c is optimal if and only if

$$\gamma \geq m_2. \tag{A.8}$$

However, the above results are not clear since we still need to compare G_1, G_2 with β_1, β_2 to get complete and non-overlapping ranges of α for each of the assignments to be optimal. Let

$$m_3 = \frac{2\beta_1(1-\beta_1) - (1-\beta_2)}{(2\beta_1 - 1)(1-\beta_2)}.$$

Note that,

$$\begin{aligned} G_1 - \beta_1 &\propto \big[(2\beta_1 - 1) - (2\beta_2 - 1)\gamma\big]\big[(\beta_1 + \beta_2 - 1)(1-\gamma) - \beta_1(2\beta_1 - 1) + \beta_1(2\beta_2 - 1)\gamma\big] \\ &= (2\beta_1 - 1)^2(1-\beta_2)\Big(\frac{2\beta_1 - 1}{2\beta_2 - 1} - \gamma\Big)\Big[\frac{2\beta_1(1-\beta_1) - (1-\beta_2)}{(2\beta_1 - 1)(1-\beta_2)} - \gamma\Big] \\ &\propto (m_1 - \gamma)(m_3 - \gamma). \end{aligned} \tag{A.9}$$

Thus, $G_1 \geq \beta_1 \Leftrightarrow (\gamma - m_1)(\gamma - m_3) \geq 0$. Similarly, we can obtain that

$$G_2 - \beta_1 \propto m_3 - \gamma, \tag{A.10}$$

$$G_2 - \beta_2 \propto m_2 - \gamma. \tag{A.11}$$

Moreover, when $\beta_1 \geq \beta_2 > \frac{1}{2}, m_2 - m_3 \propto \beta_2(1-\beta_2) - \beta_1(1-\beta_1) = (\beta_1 - \beta_2)(\beta_1 + \beta_2 - 1) \geq 0$. Combining the above results, we can obtain that:

1. When $\gamma > m_1$, since $m_1 \geq m_2 \geq m_3$, (A.9) and (A.11) yield $G_1 > \beta_1, G_2 < \beta_2$.
   (a) If $\alpha \geq \beta_1$, by (A.6), A_{12}^c is optimal if and only if $\beta_1 \leq \alpha \leq G_1$.
   (b) If $\beta_1 > \alpha \geq \beta_2$, (A.7) holds since $G_2 < \beta_2$, and A_{12}^c is always optimal.
   (c) If $\beta_2 > \alpha$, since (A.8) holds, A_{12}^c is always optimal.

   Therefore, when $\gamma > m_1$, A_{12}^c is optimal if and only if $\alpha \leq G_1$.

2. When $m_2 \leq \gamma < m_1$, then (A.9) and (A.11) yield $G_1 \leq \beta_1, G_2 \leq \beta_2$.
   (a) If $\alpha \geq \beta_1$, then (A.5) holds since $\alpha \geq \beta_1 \geq G_1$, and A_{12}^c is always optimal.
   (b) If $\beta_1 > \alpha \geq \beta_2$, (A.7) holds since $G_2 \leq \beta_2$, and A_{12}^c is always optimal.
   (c) If $\beta_2 > \alpha$, since (A.8) holds, A_{12}^c is always optimal.

   Combining this with the previous analysis for $\gamma = m_1$ yields that when $m_2 \leq \gamma \leq m_1$, A_{12}^c is always optimal.

3. When $m_3 \leq \gamma < m_2$, then (A.9)-(A.11) yield $G_1 \leq \beta_1, \beta_2 < G_2 \leq \beta_1$.
   (a) If $\alpha \geq \beta_1$, then (A.5) holds since $\alpha \geq \beta_1 \geq G_1$, and A_{12}^c is always optimal.
   (b) If $\beta_1 > \alpha \geq \beta_2$, by (A.7), A_{12}^c is optimal if and only if $\beta_1 > \alpha \geq G_2$.
   (c) If $\beta_2 > \alpha$, since (A.8) does not hold, A_{12}^c is not optimal.

   Therefore, when $m_3 \leq \gamma < m_2$, A_{12}^c is optimal if and only if $\alpha \geq G_2$.

4. When $\gamma < m_3$, then (A.9) and (A.10) yield $G_1 > \beta_1, G_2 > \beta_1$.

   (a) If $\alpha \geq \beta_1$, by (A.5), A_{12}^c is optimal if and only if $\alpha \geq G_1$.

   (b) If $\beta_1 > \alpha \geq \beta_2$, (A.7) does not hold since $G_2 > \beta_1 > \alpha$, and A_{12}^c is not optimal.

   (c) If $\beta_2 > \alpha$, since (A.8) does not hold, A_{12}^c is not optimal.

   Therefore, when $\gamma < m_3$, A_{12}^c is optimal if and only if $\alpha \geq G_1$.

Note that when $\beta_1 \geq \beta_2 > \frac{1}{2}$,

$$\begin{aligned}
G_1 - G_2 \propto & \big[(2\beta_1 - 1) - (2\beta_2 - 1)\gamma\big] \times \\
& \Big\{(2\beta_1 - 1)(1-\gamma)\big[\beta_1 + (1-\beta_2)\gamma\big] - \beta_1(2\beta_1 - 1) + \beta_1(2\beta_2 - 1)\gamma\Big\} \\
= & \big[(2\beta_1 - 1) - (2\beta_2 - 1)\gamma\big]\gamma\big[2\beta_1(1-\beta_1) - (1-\beta_2) - (2\beta_1 - 1)(1-\beta_2)\gamma\big] \\
\propto & (m_1 - \gamma)(m_3 - \gamma).
\end{aligned}$$

Thus $G_1 \geq G_2 \Leftrightarrow (\gamma - m_1)(\gamma - m_3) \geq 0$, and we can merge cases 3 and 4 and describe the results as in the proposition. $\square$

A.2.2 Proof of Proposition 3.3.5

Recall that when $\beta_1 \geq \beta_2 > \frac{1}{2}$, $x_1 = \max\{\beta_1, \alpha\}\Sigma_1, x_2 = \max\{\beta_2, \alpha\}\Sigma_2$, and

$$\frac{1-\beta_1}{\beta_2} \leq \frac{1-\beta_1}{1-\beta_2} \leq 1 \leq \frac{\beta_1}{\beta_2} < \frac{\beta_1}{1-\beta_2}. \tag{A.12}$$

Note that $\mu_{11} \leq \mu_{22} \Leftrightarrow \gamma \geq \frac{\beta_1}{1-\beta_2}$, and $\mu_{21} \leq \mu_{12} \Leftrightarrow \gamma \geq \frac{1-\beta_1}{\beta_2}$. Moreover,

$$G_4 - \beta_2 \propto (2\beta_2 - 1)(1-\beta_1) \geq 0, \tag{A.13}$$

$$G_7 - \beta_1 \propto (2\beta_1 - 1)(1-\beta_2) > 0. \tag{A.14}$$

Therefore,

(1) When $\gamma < \frac{1-\beta_1}{\beta_2}$, then $\mu_{22} < \mu_{12} < \mu_{21} < \mu_{11}$, and hence

$$T_{12}^c - T_{21}^c = \frac{x_2\mu_{11}}{x_2 + \mu_{11} - \mu_{22}} - \frac{x_2\mu_{21}}{x_2 + \mu_{21} - \mu_{12}} \propto (\mu_{11} - \mu_{21})x_2 - \mu_{11}\mu_{12} + \mu_{21}\mu_{22}$$
$$\propto (2\beta_1 - 1)\max\{\beta_2, \alpha\} + 1 - \beta_1 - \beta_2. \tag{A.15}$$

(a) If $\alpha \geq \beta_2$, then (A.15) yields that A_{12}^c is optimal if and only if $\alpha \geq G_4$.

(b) If $\alpha < \beta_2$, then (A.15) can be simplified as $(2\beta_2 - 1)(\beta_1 - 1) \leq 0$. Thus, A_{21}^c is optimal.

Since $G_4 \geq \beta_2$ by (A.13), we have shown that when $\gamma < \frac{1-\beta_1}{\beta_2}$, A_{12}^c is optimal if and only if $\alpha \geq G_4$.

(2) When $\gamma \geq \frac{\beta_1}{1-\beta_2}$, then $\mu_{21} < \mu_{11} \leq \mu_{22} < \mu_{12}$, and hence

$$T_{12}^c - T_{21}^c = \frac{x_1\mu_{22}}{x_1 + \mu_{22} - \mu_{11}} - \frac{x_1\mu_{12}}{x_1 + \mu_{12} - \mu_{21}} \propto (\mu_{22} - \mu_{12})x_1 + \mu_{11}\mu_{12} - \mu_{21}\mu_{22}$$
$$\propto -(2\beta_2 - 1)\max\{\beta_1, \alpha\} + \beta_1 + \beta_2 - 1. \tag{A.16}$$

(a) If $\alpha \geq \beta_1$, then (A.16) yields that A_{12}^c is optimal if and only if $\alpha \leq G_7$.

(b) If $\alpha < \beta_1$, then (A.16) can be simplified as $(2\beta_1 - 1)(1 - \beta_2) > 0$. Thus, A_{12}^c is always optimal.

Since $G_7 > \beta_1$ by (A.14), we have shown that when $\gamma \geq \frac{\beta_1}{1-\beta_2}$, A_{12}^c is optimal if and only if $\alpha \leq G_7$.

(3) When $\frac{1-\beta_1}{\beta_2} \leq \gamma < \frac{\beta_1}{1-\beta_2}$, then $\mu_{21} \leq \mu_{12}$ and $\mu_{22} < \mu_{11}$, and hence

$$T_{12}^c - T_{21}^c = \frac{x_2\mu_{11}}{x_2 + \mu_{11} - \mu_{22}} - \frac{x_1\mu_{12}}{x_1 + \mu_{12} - \mu_{21}}$$
$$\propto (\mu_{11} - \mu_{12})x_1x_2 + (\mu_{12} - \mu_{21})\mu_{11}x_2 + (\mu_{22} - \mu_{11})\mu_{12}x_1$$
$$\propto (\beta_1 - \beta_2\gamma)\max\{\beta_1, \alpha\}\max\{\beta_2, \alpha\} + \big[\beta_2\gamma - (1 - \beta_1)\big]\beta_1\max\{\beta_2, \alpha\}$$

$$+ \big[(1-\beta_2)\gamma - \beta_1\big]\beta_2 \max\{\beta_1, \alpha\}. \tag{A.17}$$

When $\gamma = \frac{\beta_1}{\beta_2}$, (A.17) can be simplified as follows.

$$\begin{aligned}
&(2\beta_1 - 1)\beta_1 \max\{\beta_2, \alpha\} - (2\beta_2 - 1)\beta_1 \max\{\beta_1, \alpha\} \\
=&\beta_1 \Big[\max\{(2\beta_1 - 1)\beta_2, (2\beta_1 - 1)\alpha\} - \max\{(2\beta_2 - 1)\beta_1, (2\beta_2 - 1)\alpha\}\Big] \\
\geq&\beta_1 \min\{(2\beta_1 - 1)\beta_2 - (2\beta_2 - 1)\beta_1, (2\beta_1 - 1)\alpha - (2\beta_2 - 1)\alpha\} \\
=&\beta_1(\beta_1 - \beta_2)\min\{1, 2\alpha\} \geq 0,
\end{aligned}$$

where we have used Lemma 3.6.1 to obtain the last inequality. Thus, when $\gamma = \frac{\beta_1}{\beta_2}$, A_{12}^c is optimal.

When $\gamma \neq \frac{\beta_1}{\beta_2}$,

(i) If $\alpha \geq \beta_1$, then by (A.17), $T_{12}^c \geq T_{21}^c$ if and only if

$$\alpha(\beta_1 - \beta_2\gamma) \geq \beta_1(1 - \beta_1 + \beta_2) - \beta_2(1 + \beta_1 - \beta_2)\gamma.$$

Therefore, A_{12}^c is optimal if and only if either

$$\gamma > \frac{\beta_1}{\beta_2}, \alpha \leq G_5, \tag{A.18}$$

or

$$\gamma < \frac{\beta_1}{\beta_2}, \alpha \geq G_5. \tag{A.19}$$

(ii) $\beta_1 > \alpha \geq \beta_2$, equation (A.17) becomes

$$(2\beta_1 - 1)\beta_1\alpha + \big[(1-\beta_2)\gamma - \beta_1\big]\beta_1\beta_2.$$

Then A_{12}^c is optimal if and only if

$$\alpha \geq G_6. \tag{A.20}$$

(iii) If $\beta_2 > \alpha$, equation (A.17) becomes

$$\beta_1\beta_2\big[(1-\beta_2)\gamma - (1-\beta_1)\big].$$

Then A_{12}^c is optimal if and only if

$$\gamma \geq \frac{1-\beta_1}{1-\beta_2}. \tag{A.21}$$

The above results are not clear since we still need to compare G_5, G_6 with β_1, β_2 to get complete and non-overlapping ranges of α for each of the assignments to be optimal. Let

$$m_4 = \frac{\beta_1(1+\beta_2-2\beta_1)}{\beta_2(1-\beta_2)};$$

then

$$G_5 - \beta_1 \propto (\frac{\beta_1}{\beta_2} - \gamma)(m_4 - \gamma), \tag{A.22}$$

$$G_6 - \beta_1 \propto m_4 - \gamma, \tag{A.23}$$

$$G_6 - \beta_2 \propto \frac{1-\beta_1}{1-\beta_2} - \gamma. \tag{A.24}$$

Moreover, when $\beta_1 \geq \beta_2 > \frac{1}{2}$, by (A.12) we have

$$m_4 - \frac{1-\beta_1}{1-\beta_2} \propto -(\beta_1-\beta_2)(2\beta_1-1) \leq 0,$$
$$m_4 - \frac{1-\beta_1}{\beta_2} \propto 2\beta_1(1-\beta_1) - (1-\beta_2).$$

Thus,

(i) if $2\beta_1(1-\beta_1) > (1-\beta_2)$, then $\frac{1-\beta_1}{\beta_2} < m_4 \le \frac{1-\beta_1}{1-\beta_2}$;

(ii) if $2\beta_1(1-\beta_1) \le (1-\beta_2)$, then $m_4 \le \frac{1-\beta_1}{\beta_2}$.

Combining the above results with (A.12), we can obtain that when $\frac{1-\beta_1}{\beta_2} \le \gamma < \frac{\beta_1}{1-\beta_2}$ and $\gamma \ne \frac{\beta_1}{\beta_2}$, we have the following cases:

(3.a) When $\frac{\beta_1}{\beta_2} < \gamma \le \frac{\beta_1}{1-\beta_2}$, by (A.22) and (A.24), $G_5 > \beta_1, G_6 < \beta_2$.

i. If $\alpha \ge \beta_1$, by (A.18), A_{12}^c is optimal if and only if $\beta_1 \le \alpha \le G_5$.

ii. If $\beta_1 > \alpha \ge \beta_2$, (A.20) holds since $G_6 < \beta_2$, and A_{12}^c is always optimal.

iii. If $\beta_2 > \alpha$, since (A.21) holds, A_{12}^c is always optimal.

Thus, when $\frac{\beta_1}{\beta_2} < \gamma \le \frac{\beta_1}{1-\beta_2}$, A_{12}^c is optimal if and only if $\alpha \le G_5$.

(3.b) When $\frac{1-\beta_1}{1-\beta_2} \le \gamma < \frac{\beta_1}{\beta_2}$, by (A.22) and (A.24), $G_5 < \beta_1, G_6 \le \beta_2$.

i. If $\alpha \ge \beta_1$, then (A.19) holds since $\alpha \ge \beta_1 > G_5$, and A_{12}^c is always optimal.

ii. If $\beta_1 > \alpha \ge \beta_2$, (A.20) holds since $G_6 \le \beta_2$, and A_{12}^c is always optimal.

iii. If $\beta_2 > \alpha$, since (A.21) holds, A_{12}^c is always optimal.

Combining this with the previous analysis for $\gamma = \frac{\beta_1}{\beta_2}$ yields that when $\frac{1-\beta_1}{1-\beta_2} \le \gamma \le \frac{\beta_1}{\beta_2}$, A_{12}^c is always optimal.

(3.c1) When either $2\beta_1(1-\beta_1) \le (1-\beta_2)$ and $\frac{1-\beta_1}{\beta_2} \le \gamma < \frac{1-\beta_1}{1-\beta_2}$, or $2\beta_1(1-\beta_1) > (1-\beta_2)$ and $m_4 \le \gamma < \frac{1-\beta_1}{1-\beta_2}$, then by (A.22)-(A.24), $G_5 \le \beta_1, \beta_2 < G_6 \le \beta_1$.

i. If $\alpha \ge \beta_1$, (A.19) holds since $G_5 \le \beta_1$, and A_{12}^c is optimal.

ii. If $\beta_1 > \alpha \ge \beta_2$, by (A.20), A_{12}^c is optimal if and only if $\beta_1 > \alpha \ge G_6$.

iii. If $\beta_2 > \alpha$, since (A.21) does not hold, A_{12}^c is not optimal.

Thus, when either $2\beta_1(1-\beta_1) \le (1-\beta_2)$ and $\frac{1-\beta_1}{\beta_2} \le \gamma < \frac{1-\beta_1}{1-\beta_2}$, or $2\beta_1(1-\beta_1) > (1-\beta_2)$ and $m_4 \le \gamma < \frac{1-\beta_1}{1-\beta_2}$, A_{12}^c is optimal if and only if $\alpha \ge G_6$.

(3.c2) When $2\beta_1(1-\beta_1) > (1-\beta_2)$ and $\frac{1-\beta_1}{\beta_2} \le \gamma < m_4$, by (A.22) and (A.23), $G_5 > \beta_1, G_6 > \beta_1$.

i. If $\alpha \geq \beta_1$, by (A.19), A_{12}^c is optimal if and only if $\alpha \geq G_5$.

ii. If $\beta_1 > \alpha \geq \beta_2$, (A.20) does not hold since $G_6 > \beta_1$, and A_{12}^c is not optimal.

iii. If $\beta_2 > \alpha$, since (A.21) does not hold, A_{12}^c is not optimal.

Thus, when $2\beta_1(1-\beta_1) > (1-\beta_2)$ and $\frac{1-\beta_1}{\beta_2} \leq \gamma < m_4$, A_{12}^c is optimal if and only if $\alpha \geq G_5$.

We now combine cases (3.c1) and (3.c2). Note that

$$\begin{aligned}
G_5 - G_6 &\propto (\gamma\beta_2 - \beta_1)\Big[\gamma^2\beta_2^2(1-\beta_2) + \gamma\beta_2(2\beta_1^2 - 2\beta_1\beta_2 + \beta_2 - 1) \\
&\quad + \beta_1(1-\beta_1)(1+\beta_2-2\beta_1)\Big] \\
&= (\gamma\beta_2 - \beta_1)\big[\gamma\beta_2 - (1-\beta_1)\big]\big[\gamma\beta_2(1-\beta_2) - \beta_1(1+\beta_2-2\beta_1)\big] \\
&\propto (\gamma - \frac{\beta_1}{\beta_2})(\gamma - \frac{1-\beta_1}{\beta_2})(\gamma - m_4). \qquad (A.25)
\end{aligned}$$

Thus, when $\frac{1-\beta_1}{\beta_2} \leq \gamma < \frac{1-\beta_1}{1-\beta_2}$, $G_5 - G_6 \propto m_4 - \gamma$. Moreover, when either $2\beta_1(1-\beta_1) \leq (1-\beta_2)$ and $\frac{1-\beta_1}{\beta_2} \leq \gamma < \frac{1-\beta_1}{1-\beta_2}$, or $2\beta_1(1-\beta_1) > (1-\beta_2)$ and $m_4 \leq \gamma < \frac{1-\beta_1}{1-\beta_2}$, we have $G_6 \geq G_5$; and when $2\beta_1(1-\beta_1) > (1-\beta_2)$ and $\frac{1-\beta_1}{\beta_2} \leq \gamma < m_4$, we have $G_6 \leq G_5$. Therefore, we can combine cases (3.c1) and (3.c2) as:

(3.c) when $\frac{1-\beta_1}{\beta_2} \leq \gamma < \frac{1-\beta_1}{1-\beta_2}$, A_{12}^c is optimal if and only if $\alpha \geq \max\{G_5, G_6\}$.

We now combine cases (1) and (2) with cases (3.c) and (3.a), respectively. Similar to the way we obtain (A.25), we have

$$G_4 - G_5 \propto -(\gamma - \frac{\beta_1}{\beta_2})(\gamma - \frac{1-\beta_1}{\beta_2})(\beta_1 - \beta_2), \qquad (A.26)$$

$$G_4 - G_6 \propto \gamma - \frac{1-\beta_1}{\beta_2}, \qquad (A.27)$$

$$G_5 - G_7 \propto -(\gamma - \frac{\beta_1}{\beta_2})(\gamma - \frac{\beta_1}{1-\beta_2})(\beta_1 - \beta_2). \qquad (A.28)$$

When $\gamma < \frac{1-\beta_1}{\beta_2}$, (A.26) yields $G_4 < G_5$. On the other hand, when $\frac{1-\beta_1}{\beta_2} \leq \gamma < \frac{1-\beta_1}{1-\beta_2}$, (A.26) and (A.27) yield $G_4 \geq G_5$, $G_4 \geq G_6$. Thus, we can combine cases (1) and (3.c) as described in the first case of the proposition. And by (A.28), when $\gamma > \frac{\beta_1}{\beta_2}$, $G_5 \geq G_7 \Leftrightarrow \gamma \leq \frac{\beta_1}{1-\beta_2}$, and we can merge cases (2) and (3.a) as described in the last case of the proposition. $\square$

A.3 Proofs for Section 3.5 with Collaborative Servers

We provide the proofs of Propositions 3.5.2, 3.5.3, and 3.5.5 in Appendices A.3.1, A.3.2, and A.3.3, respectively.

A.3.1 Proof of Proposition 3.5.2

There are two available collaborative task assignment policies, A^c_{12}, A^c_{21}, depending on the primary assignment. Thus, for collaborative task assignment to be no worse than non-collaboration, the throughput of the optimal collaborative task assignment needs to be no lower than the throughput of non-collaboration, i.e., $\max\{T^c_{12}, T^c_{21}\} \geq T^{nc}$. Note that

$$\begin{aligned}
T^c_{12} - T^{nc} \propto\ & (\mu_{11}+\mu_{12})(\mu_{21}+\mu_{22})x_1x_2 \sum_{k=0}^{B_1+B_2+1} \mu_{11}^k \mu_{22}^{B_1+B_2+1-k} \\
& - \Big[\mu_{11}\mu_{12}(\mu_{21}+\mu_{22}) + \mu_{21}\mu_{22}(\mu_{11}+\mu_{12})\Big]\times \\
& \Big(x_1x_2 \sum_{k=0}^{B_1+B_2} \mu_{11}^k \mu_{22}^{B_1+B_2-k} + x_1\mu_{11}^{B_1+B_2+1} + x_2\mu_{22}^{B_1+B_2+1}\Big) \\
=\ & x_1x_2(\mu_{11}+\mu_{12})(\mu_{21}+\mu_{22})\Big(\mu_{11}^{B_1+B_2+1} + \mu_{22}^{B_1+B_2+1} + \mu_{11}\mu_{22} \sum_{k=0}^{B_1+B_2-1} \mu_{11}^k \mu_{22}^{B_1+B_2-1-k}\Big) \\
& - x_1x_2\mu_{11}\mu_{12}(\mu_{21}+\mu_{22})\Big(\mu_{11}^{B_1+B_2} + \mu_{22} \sum_{k=0}^{B_1+B_2-1} \mu_{11}^k \mu_{22}^{B_1+B_2-1-k}\Big) \\
& - x_1x_2\mu_{21}\mu_{22}(\mu_{11}+\mu_{12})\Big(\mu_{22}^{B_1+B_2} + \mu_{11} \sum_{k=0}^{B_1+B_2-1} \mu_{11}^k \mu_{22}^{B_1+B_2-1-k}\Big) \\
& - h_1\big(x_1\mu_{11}^{B_1+B_2+1} + x_2\mu_{22}^{B_1+B_2+1}\big)
\end{aligned}$$

$$
\begin{aligned}
&= x_1x_2\Big[\mu_{11}^{B_1+B_2+2}(\mu_{21}+\mu_{22}) + \mu_{22}^{B_1+B_2+2}(\mu_{11}+\mu_{12}) \\
&+ (\mu_{11}\mu_{22} - \mu_{21}\mu_{12})\mu_{11}\mu_{22} \sum_{k=0}^{B_1+B_2-1} \mu_{11}^{k}\mu_{22}^{B_1+B_2-1-k}\Big] \\
&- h_1\big(x_1\mu_{11}^{B_1+B_2+1} + x_2\mu_{22}^{B_1+B_2+1}\big).
\end{aligned}
$$

Similarly,

$$
\begin{aligned}
T_{21}^{c} - T^{nc} \propto\ & x_1x_2\Big[\mu_{21}^{B_1+B_2+2}(\mu_{11}+\mu_{12}) + \mu_{12}^{B_1+B_2+2}(\mu_{21}+\mu_{22}) \\
&+ (\mu_{21}\mu_{12} - \mu_{11}\mu_{22})\mu_{21}\mu_{12} \sum_{k=0}^{B_1+B_2-1} \mu_{21}^{k}\mu_{12}^{B_1+B_2-1-k}\Big] \\
&- h_1\big(x_1\mu_{21}^{B_1+B_2+1} + x_2\mu_{12}^{B_1+B_2+1}\big).
\end{aligned}
$$

The result follows. $\square$

A.3.2 Proof of Proposition 3.5.3

When $\alpha \geq 1 + |\beta_1 - \beta_2| \geq 1 \geq h$ (see equation (3.19)), teamwork with task partitioning is no worse than non-collaboration by Proposition 3.4.3, and is no worse than collaborative task assignment by Proposition 3.5.1.

When $\alpha < 1 + |\beta_1 - \beta_2|$, collaborative task assignment is better than teamwork, thus we only need to compare collaborative task assignment and non-collaboration. Note that when $\alpha \leq \alpha_0$, $D_i(\alpha) = D_i(\alpha_0)$ are constant for $i = 1, 2$. Moreover, $D_i(\alpha) \geq D_i(\alpha_0)$ always holds for $i = 1, 2$. Thus, if $\max\{D_1(\alpha_0), D_2(\alpha_0)\} \geq 1$, then $\max\{D_1, D_2\} \geq 1$, and by Proposition 3.5.2, collaborative task assignment is optimal.

Note that $D_1(\alpha)$ and $D_2(\alpha)$ are first constant and then strictly increasing in α with $\lim_{\alpha\to\infty} D_1(\alpha) = \lim_{\alpha\to\infty} D_2(\alpha) = \infty$. Moreover, when $\alpha \geq \max\{\beta_1, 1-\beta_1, \beta_2, 1-\beta_2\}$, then $x_1 = \alpha\Sigma_1, x_2 = \alpha\Sigma_2$, and $D_1(\alpha)$ and $D_2(\alpha)$ are linearly increasing in α. Thus, if $\max\{D_1(\alpha_0), D_2(\alpha_0)\} < 1$, we can find a unique α^* such that $\max\{D_1(\alpha^*), D_2(\alpha^*)\} = 1$. It then follows from Proposition 3.5.2 that if $\alpha^* \leq 1 + |\beta_1 - \beta_2|$, then non-collaboration is optimal for $\alpha < \alpha^*$ and collaborative task assignment is best for $\alpha^* \leq \alpha < 1 + |\beta_1 - \beta_2|$. We conclude the proof by proving that $\max\{D_1(h), D_2(h)\} \geq 1$, which implies that $\alpha^* \leq h \leq 1 + |\beta_1 - \beta_2|$.

Since $x_1 \geq \alpha\Sigma_1, x_2 \geq \alpha\Sigma_2$, and $D_1(\alpha)$ and $D_2(\alpha)$ are non-decreasing in α, we can obtain that:

$$\begin{aligned}
D_1(h) \geq & \Big[\mu_{11}^{B_1+B_2+2}(\mu_{21}+\mu_{22}) + \mu_{22}^{B_1+B_2+2}(\mu_{11}+\mu_{12}) \\
& + (\mu_{11}\mu_{22} - \mu_{21}\mu_{12})\mu_{11}\mu_{22} \sum_{k=0}^{B_1+B_2-1} \mu_{11}^{k}\mu_{22}^{B_1+B_2-1-k}\Big] \\
& \times \frac{h\Sigma_1\Sigma_2}{h_1(\mu_{11}^{B_1+B_2+1}\Sigma_1 + \mu_{22}^{B_1+B_2+1}\Sigma_2)} \equiv D_{1l}(h), \\
D_2(h) \geq & \Big[\mu_{21}^{B_1+B_2+2}(\mu_{11}+\mu_{12}) + \mu_{12}^{B_1+B_2+2}(\mu_{21}+\mu_{22}) \\
& + (\mu_{21}\mu_{12} - \mu_{11}\mu_{22})\mu_{21}\mu_{12} \sum_{k=0}^{B_1+B_2-1} \mu_{21}^{k}\mu_{12}^{B_1+B_2-1-k}\Big] \\
& \times \frac{h\Sigma_1\Sigma_2}{h_1(\mu_{21}^{B_1+B_2+1}\Sigma_1 + \mu_{12}^{B_1+B_2+1}\Sigma_2)} \equiv D_{2l}(h).
\end{aligned}$$

By equation (3.18), we can obtain that

$$\frac{h\Sigma_1\Sigma_2}{h_1} = \frac{\Sigma_1+\Sigma_2}{h_2} = \frac{\Sigma_1+\Sigma_2}{(\mu_{11}+\mu_{12})(\mu_{21}+\mu_{22})}.$$

Using this equation, we have

$$
\begin{aligned}
D_{1l}(h) - 1 \propto\ & \left[\mu_{11}^{B_1+B_2+2}(\mu_{21}+\mu_{22}) + \mu_{22}^{B_1+B_2+2}(\mu_{11}+\mu_{12})\right](\Sigma_1+\Sigma_2) \\
& + (\mu_{11}\mu_{22} - \mu_{21}\mu_{12})\mu_{11}\mu_{22}\sum_{k=0}^{B_1+B_2-1}\mu_{11}^{k}\mu_{22}^{B_1+B_2-1-k}(\Sigma_1+\Sigma_2) \\
& - (\mu_{11}+\mu_{12})(\mu_{21}+\mu_{22})(\mu_{11}^{B_1+B_2+1}\Sigma_1 + \mu_{22}^{B_1+B_2+1}\Sigma_2) \\
& = \mu_{11}^{B_1+B_2+1}(\mu_{21}+\mu_{22})\left[\mu_{11}(\Sigma_1+\Sigma_2) - (\mu_{11}+\mu_{12})\Sigma_1\right] \\
& + \mu_{22}^{B_1+B_2+1}(\mu_{11}+\mu_{12})\left[\mu_{22}(\Sigma_1+\Sigma_2) - (\mu_{21}+\mu_{22})\Sigma_2\right] \\
& + (\mu_{11}\mu_{22} - \mu_{21}\mu_{12})(\Sigma_1+\Sigma_2)\sum_{k=0}^{B_1+B_2-1}\mu_{11}^{k+1}\mu_{22}^{B_1+B_2-k} \\
& = (\mu_{11}\mu_{22} - \mu_{21}\mu_{12})\times \\
& \left[\mu_{11}^{B_1+B_2+1}(\mu_{21}+\mu_{22}) + \mu_{22}^{B_1+B_2+1}(\mu_{11}+\mu_{12}) + (\Sigma_1+\Sigma_2)\sum_{k=0}^{B_1+B_2-1}\mu_{11}^{k+1}\mu_{22}^{B_1+B_2-k}\right].
\end{aligned}
$$

Similarly, we can obtain that

$$
\begin{aligned}
D_{2l}(h) - 1 \propto\ & (\mu_{21}\mu_{12} - \mu_{11}\mu_{22})\times \\
& \left[\mu_{21}^{B_1+B_2+1}(\mu_{11}+\mu_{12}) + \mu_{12}^{B_1+B_2+1}(\mu_{21}+\mu_{22}) + (\Sigma_1+\Sigma_2)\sum_{k=0}^{B_1+B_2-1}\mu_{21}^{k+1}\mu_{12}^{B_1+B_2-k}\right].
\end{aligned}
$$

Thus, at least one of $D_{1l}(h)$ and $D_{2l}(h)$ is no less than 1, which yields that $\max\{D_1(h), D_2(h)\} \geq 1$. This completes the proof. $\square$

A.3.3 Proof of Proposition 3.5.5

For (1), the result follows from the proof of Proposition 3.4.1.

For (3), we know from (1) and Section 3.4.3 that teamwork with and without task partitioning and non-collaboration are equivalent. The result now follows from the proof of Proposition 3.5.1 and the fact that collaborative task assignment contains static and flexible task assignment as special cases.

For (2), note that, when $\beta_1 = \beta_2$, $h = 1$, and D_1, D_2 can be transformed as follows:

$$\begin{aligned} D_1 &= \frac{\mu_{11}^{B_1+B_2+1}\Sigma_1 + \mu_{22}^{B_1+B_2+1}\Sigma_2}{\Sigma_1\Sigma_2} \times \frac{x_1x_2}{\mu_{11}^{B_1+B_2+1}x_1 + \mu_{22}^{B_1+B_2+1}x_2}; \\ D_2 &= \frac{\mu_{21}^{B_1+B_2+1}\Sigma_1 + \mu_{12}^{B_1+B_2+1}\Sigma_2}{\Sigma_1\Sigma_2} \times \frac{x_1x_2}{\mu_{21}^{B_1+B_2+1}x_1 + \mu_{12}^{B_1+B_2+1}x_2}. \end{aligned} \tag{A.29}$$

Note that, the second part of D_1 is non-decreasing with respect to x_1, x_2.

When $\alpha > 1$, we obtain the result directly from Propositions 3.4.2 and 3.5.1. When $\alpha < 1$, we have $x_1 \le \Sigma_1, x_2 \le \Sigma_2$. Thus,

$$\frac{x_1x_2}{\mu_{11}^{B_1+B_2+1}x_1 + \mu_{22}^{B_1+B_2+1}x_2} \le \frac{\Sigma_1\Sigma_2}{\mu_{11}^{B_1+B_2+1}\Sigma_1 + \mu_{22}^{B_1+B_2+1}\Sigma_2}.$$

Combining this with equation (A.29) yield that, $D_1 \le 1$ when $\alpha < 1$. Similarly, we can obtain $D_2 \le 1$. Thus, $\max\{D_1, D_2\} \le 1$. By Propositions 3.5.1 and 3.5.2, when $\alpha < 1$, non-collaboration is optimal. □

A.4 Comparisons of Server Coordination Methods

We compare static and flexible task assignments with teamwork with task partitioning and non-collaboration in Appendices A.4.1 and A.4.2, respectively.

A.4.1 Comparison of Static Task Assignment and Other Server Coordination Methods

In this section, we compare static task assignment with teamwork with task partitioning and non-collaboration by calculating the differences of their throughputs. Some of the proofs are omitted to conserve space.

First, we compare static task assignment and teamwork with task partitioning. Intuitively, servers work separately in static task assignment but always together in teamwork. Thus we expect teamwork to be better when the server synergy α is high. The following proposition verifies this intuition.

Proposition A.4.1. *Let*

$$D_6 \equiv \frac{h_2\mu_{21}\mu_{12}\sum_{k=0}^{B_1+B_2+1}\mu_{21}^k\mu_{12}^{B_1+B_2+1-k}}{h_1\sum_{k=0}^{B_1+B_2+2}\mu_{21}^k\mu_{12}^{B_1+B_2+2-k}}.$$

Then, teamwork with task partitioning is no worse than static task assignment if and only if

$$\alpha \geq \max\{D_3, D_6\} \times h.$$

Since $D_3 \times h, D_6 \times h$ can be either greater or less than 1, teamwork with task partitioning can be either better or worse than static task assignment when $\alpha = 1$. To see this, suppose that $B_1 = B_2 = 0$ and $\mu_{11} = k\mu_{21}, \mu_{22} = k\mu_{12}$ for some $k > 0$. Then teamwork with task partitioning is no worse than static task assignment if and only if $\frac{\mu_{21}}{\mu_{12}} + \frac{\mu_{12}}{\mu_{21}} \geq \max\{k - 1, \frac{1}{k} - 1\}$. Therefore, when $k = 1$, teamwork with task partitioning is better; when k is large or close to zero, static task assignment is better. This example shows that teamwork with task partitioning is desirable when server collaboration is efficient (i.e., α large) and the servers are not heavily specialized ($\frac{\mu_{11}}{\mu_{21}}$ and $\frac{\mu_{22}}{\mu_{12}}$ are moderate); otherwise, static task assignment is more preferable.

Next, we compare non-collaboration and static task assignment.

Proposition A.4.2. *Static task assignment is no worse than non-collaboration if and only if*

$$\max\{D_3, D_6\} \geq 1.$$

Note that D_3, D_6 can be either greater or less than 1. To see this, if $B_1 = B_2 = 0$, $\mu_{11} = k\mu_{21}, \mu_{22} = k\mu_{12}$ for some $k \in \mathcal{R}$, then static task assignment is better than non-collaboration if $\frac{\mu_{21}}{\mu_{12}} + \frac{\mu_{12}}{\mu_{21}} \leq \max\{k^2 - k, \frac{1}{k^2} - \frac{1}{k}\}$. Thus, when $k = 1$, non-collaboration is better; when k is large or close to zero, static task assignment is better. Again static task assignment is preferable when the servers are heavily specialized.

The following proposition concludes our comparisons of static task assignment with teamwork with task partitioning and non-collaboration.

Proposition A.4.3. *1. When $\alpha < \max\{1, D_3, D_6\} \times h$,*

(a) If $\max\{D_3, D_6\} < 1$, non-collaboration is optimal;

(b) If $\max\{D_3, D_6\} \geq 1$, static task assignment is optimal.

2. When $\alpha \geq \max\{1, D_3, D_6\} \times h$, teamwork with task partitioning is optimal.

Proof. First, remember that by Proposition 3.4.3, teamwork with task partitioning is no worse than non-collaboration if and only if $\alpha \geq h$. And by Proposition A.4.1, teamwork with task partitioning is no worse than static task assignment if and only if $\alpha \geq \max\{D_3, D_6\} \times h$. Thus, when $\alpha \geq \max\{1, D_3, D_6\} \times h$, teamwork with task partitioning is the best method.

When $\alpha < \max\{1, D_3, D_6\} \times h$, combining Propositions 3.4.3, A.4.1, and A.4.2 yield the desired results. □

The intuition for Proposition A.4.3 is similar to that for the comparison of collaborative task assignment and other server coordination methods in Section 3.5.1. The value of $\max\{D_3, D_6\}$ provides information on server specialty. When the synergy level is not high, non-collaboration is the best when servers are not highly specialized; otherwise static task assignment is the best because it takes advantage of server specialty and avoids using servers with extremely low service rates at some subtask. When the synergy level is high, teamwork with task partitioning is the best since it takes advantage of efficient server collaboration.

A.4.2 Comparison of Flexible Task Assignment and Other Server Coordination Methods

In this section, we compare flexible task assignment with teamwork with task partitioning and non-collaboration. Without loss of generality, assume $\mu_{11} \geq \mu_{21}$. The proofs are omitted to conserve space.

First, we compare flexible task assignment and teamwork with task partitioning.

Proposition A.4.4. *If $\mu_{11} \geq \mu_{21}$, then teamwork with task partitioning is no worse than flexible task assignment if and only if*

$$\alpha \geq \max\{D_3, D_4, D_5\} \times h.$$

Note that $D_3 \times h, D_4 \times h, D_5 \times h$ can be either greater or less than 1. Thus, when $\alpha = 1$, teamwork with task partitioning can be either better or worse than flexible task assignment. For example, when $B_1 = B_2 = 0$, if $\mu_{11} = 2, \mu_{21} = \mu_{12} = 1, \mu_{22} = \frac{1}{16}$, so that the service rate of server 2 at station 2 is significantly lower than the other service rates, flexible task assignment is better since it can avoid this low service rate by working separately. By contrast, if $\mu_{11} = 2, \mu_{21} = \mu_{12} = 1, \mu_{22} = \frac{1}{2}$, so that the gap between the best and the worst service rates is not very large, then teamwork with task partitioning is better since it takes advantage of additive combined service rates.

The comparison of flexible task assignment and non-collaboration has been addressed in Proposition 3.5.4, so we do not repeat it here. The following proposition concludes our comparisons of flexible task assignment with teamwork with task partitioning and non-collaboration. The proof of the proposition is similar to Proposition A.4.3 and is omitted to conserve space.

Proposition A.4.5. *If we label the servers such that $\mu_{11} \geq \mu_{21}$, then*

1. *When $\alpha < \max\{1, D_3, D_4, D_5\} \times h$,*
   (a) *If $\max\{D_3, D_4, D_5\} < 1$, non-collaboration is optimal;*
   (b) *If $\max\{D_3, D_4, D_5\} \geq 1$, flexible task assignment is optimal.*
2. *When $\alpha \geq \max\{1, D_3, D_4, D_5\} \times h$, teamwork with task partitioning is optimal.*

Intuitively, servers only collaborate under teamwork with task partitioning. Thus when the synergy level is high, we prefer teamwork with task partitioning since it takes advantage of efficient server collaboration. When the synergy level is not high, although the exact values of the thresholds are complex, the example that follows Proposition 3.5.4 suggests that flexible task assignment is better when the servers are highly specialized, while non-collaboration is better otherwise.

A.5 Teamwork Approaches

In this section, we provide model descriptions of teamwork with and without task partitioning. Our analysis in this section focuses on a single station and can be applied to any station in a system with multiple stations.

For teamwork without task partitioning, the servers work as a team on a combined task. We start by determining the service rates of the two servers at the station. Recall that the service requirement of subtask j is S_j with $E[S_j] = 1$, for $j = 1, 2$. Therefore, the time it takes server i to finish both subtasks is

$$\frac{S_1}{\mu_{i1}} + \frac{S_2}{\mu_{i2}},$$

for $i = 1, 2$. Then, the service rate of server i working on the combined task is the reciprocal of the average service time:

$$\frac{1}{\frac{E[S_1]}{\mu_{i1}} + \frac{E[S_2]}{\mu_{i2}}} = \frac{1}{\frac{1}{\mu_{i1}} + \frac{1}{\mu_{i2}}},$$

for $i = 1, 2$. Thus, the combined average service rate of teamwork without task partitioning for the combined task is

$$\alpha\Big(\frac{1}{\frac{1}{\mu_{11}} + \frac{1}{\mu_{12}}} + \frac{1}{\frac{1}{\mu_{21}} + \frac{1}{\mu_{22}}}\Big).$$

Hence, the actual service time to finish the combined task of teamwork without task partitioning approach is

$$S^t = \frac{\frac{S_1+S_2}{E[S_1+S_2]}}{\alpha\Big(\frac{1}{\frac{1}{\mu_{11}}+\frac{1}{\mu_{12}}} + \frac{1}{\frac{1}{\mu_{21}}+\frac{1}{\mu_{22}}}\Big)} = \frac{S_1 + S_2}{2\alpha\Big(\frac{1}{\frac{1}{\mu_{11}}+\frac{1}{\mu_{12}}} + \frac{1}{\frac{1}{\mu_{21}}+\frac{1}{\mu_{22}}}\Big)},$$

where we divide $S_1 + S_2$ by $E[S_1 + S_2]$ to ensure that the total service requirement for the combined task has mean one. Observe that $T^t = \frac{1}{E[S^t]}$, as desired.

For teamwork with task partitioning, recall that the combined service rate of the two servers on subtask j is $\alpha(\mu_{1j} + \mu_{2j})$ for $j = 1, 2$. Thus, the time it takes teamwork with task partitioning to finish subtask j is

$$\frac{S_j}{\alpha(\mu_{1j} + \mu_{2j})},$$

for $j = 1, 2$. And the total service time of teamwork with task partitioning to finish both subtasks is

$$S^{tp} = \frac{S_1}{\alpha(\mu_{11} + \mu_{21})} + \frac{S_2}{\alpha(\mu_{12} + \mu_{22})}.$$

Observe that $T^{tp} = \frac{1}{E[S^{tp}]}$, as desired.

Note that, when the servers are generalists, that is $\mu_{ij} = \mu_i \gamma_j$ for $i, j = 1, 2$, S^t and S^{tp} can be simplified as follows:

$$S^t = \frac{1}{\alpha(\mu_1 + \mu_2)} \Big(\frac{S_1 + S_2}{2}\Big)\Big(\frac{1}{\gamma_1} + \frac{1}{\gamma_2}\Big),$$
$$S^{tp} = \frac{1}{\alpha(\mu_1 + \mu_2)} \Big(\frac{S_1}{\gamma_1} + \frac{S_2}{\gamma_2}\Big).$$

Thus, $E[S^t] = E[S^{tp}]$ when the servers are generalists. Moreover, when $\gamma_1 = \gamma_2$ (i.e., the task difficulties of both subtasks are the same), then $S^t = S^{tp}$, and the two teamwork approaches obtain the same results. Intuitively, when the servers are generalists, since the servers always work together as a team in both teamwork approaches, the effect caused by the individual server abilities (i.e., μ_i) is eliminated. Thus, when the task difficulties are the same, the two teamwork approaches are equivalent.

By calculation, we can obtain that, if $Var(S_i) = \sigma_i^2$, for $i = 1, 2$, then

$$Var(S^t) - Var(S^{tp}) = -\frac{1}{[\alpha(\mu_1 + \mu_2)]^2} \big(\frac{1}{\gamma_1} - \frac{1}{\gamma_2}\big)\Big[\sigma_1^2(\frac{3}{4\gamma_1} + \frac{1}{4\gamma_2}) - \sigma_2^2(\frac{1}{4\gamma_1} + \frac{3}{4\gamma_2})\Big].$$

Thus, when $\sigma_1^2 = \sigma_2^2 = \sigma^2$ (e.g., when S_1, S_2 are exponentially distributed with rates $\xi_1 = \xi_2 = 1$, then $\sigma_1^2 = \sigma_2^2 = 1$), we obtain

$$Var(S^t) - Var(S^{tp}) = -\frac{\sigma^2}{2[\alpha(\mu_1 + \mu_2)]^2}\Big(\frac{1}{\gamma_1} - \frac{1}{\gamma_2}\Big)^2 \leq 0.$$

That is, when the servers are generalists and the service requirements of the two subtasks have the same variance, then the variance of teamwork (without task partitioning) is never larger than that of teamwork with task partitioning. Moreover, the variance of teamwork is strictly lower than that of teamwork with task partitioning when the task difficulties are different.

APPENDIX B

APPENDIX FOR CHAPTER 5

Proof of Lemma 5.3.1. We label the jobs at each station such that at station j, job i is the ith good job arriving at station j, for $j \in \{1, \ldots, N\}, i \geq 1$. For $j \in \{1, \ldots, N\}, i \geq 1$, let $\phi_j(i)$ be the service requirement of ith completed job at station j. And for $\pi \in \Pi, j \in \{1, \ldots, N\}, i \geq 1$, let $C_j^\pi(i)$ be the time of ith service completion at station j, $D_j^\pi(i)$ be the departure time of ith job from station j, $K_j^\pi(i)$ be the labels of the jobs that are completed with no defects at station j, and $G_j^\pi(i)$ be the departure time of ith good job (with no defects) at station j. Moreover, let $p_{j,i}^\pi$ be the defect probability of the last server that completes service of ith job at station j under policy $\pi \in \Pi$, for $j \in \{1, \ldots, N\}, i \geq 1$. Then $p_{j,i}^{\pi'} = p_{j,i}^\pi$ for $j \in \{1, \ldots, N\}, i \geq 1$ by the definition of policy π'. Let $\mu_{j,i}^\pi$ be the average service rate of the ith job at station j under policy π, so that $\frac{\phi_j(i)}{\mu_{j,i}^\pi}$ is the time spent by ith job at station j under policy π. Specifically, let $m_{j,i}^\pi$ be the number of server reassignments that occur during the time the ith job spends at station j when the policy π is employed, let $\phi_j(i,m), m \in \{1, \ldots, m_{j,i}^\pi + 1\}$ denote the service requirement fulfilled between the $(m-1)$th and mth server reassignments, and let $\mu_{j,i}^\pi(m), m \in \{1, \ldots, m_{j,i}^\pi + 1\}$ denote the service rate of the server assigned to station j for the time between the $(m-1)$th and mth server reassignments. Then, $\phi_j(i) = \sum_{m=1}^{m_{j,i}^\pi+1} \phi_j(i,m)$ and $\frac{\phi_j(i)}{\mu_{j,i}^\pi} = \sum_{m=1}^{m_{j,i}^\pi+1} \frac{\phi_j(i,m)}{\mu_{j,i}^\pi(m)}$. Since policy π' always uses the server with highest service rate at each station, we can obtain that $\mu_{j,i}^\pi \leq \mu_{j,i}^{\pi'}$ and $\frac{\phi_j(i)}{\mu_{j,i}^\pi} \geq \frac{\phi_j(i)}{\mu_{j,i}^{\pi'}}$ for $j \in \{1, \ldots, N\}, i \geq 1$.

We prove this lemma by proving that $D_j^{\pi'}(i) \leq D_j^\pi(i)$, $G_j^{\pi'}(i) \leq G_j^\pi(i)$ for $j \in \{1, \ldots, N\}, i \geq 1$. Without loss of generality, assume the system is initially empty. For any $\pi \in \Pi$, we generate the sample path of π according to the following algorithm. Note that, i_j is tracking the number of successful departures from station j, and l_j is tracking the total number of departures from station j, for $j \in \{1, \dots, N\}$.

```
Algorithm 1: Sample Path Generator of $\pi \in \Pi$
  Input: N, $B_j$, $p^{\pi}_{j,i}$,$\mu^{\pi}_{j,i}$, for $j \in \{1, \dots, N\}, i \geq 1$
1 Initialize: $i_j = l_j = 0$, for $j = 1, \dots, N$; $j = 1$; $G^{\pi}_j(i) = D^{\pi}_j(i) = 0$ for
  $j \notin \{1, \dots, N\}, i \leq 0$.
2 Check,
    (I) If $j \geq 2$ and $l_j + 1 > i_{j-1}$, then back to the top of step 2.
    (II) Otherwise, $l_j = l_j + 1$ and go to step 3.
3 Compute $C^{\pi}_j(l_j) = \max\{G^{\pi}_{j-1}(l_j) + \frac{\phi_j(l_j)}{\mu^{\pi}_{j,l_j}}, D^{\pi}_j(l_j - 1) + \frac{\phi_j(l_j)}{\mu^{\pi}_{j,l_j}}\}$.
    Generate $u_{j,l_j} \sim U(0, 1)$.
    Check,
    (I) If $u_{j,l_j} > p^{\pi}_{j,l_j}$, then
    $i_j = i_j + 1$,
    $K^{\pi}_j(i_j) = l_n$,
    $D^{\pi}_j(l_j) = \max\{C^{\pi}_j(l_j), D^{\pi}_{j+1}(i_j - B_j - 1)\}$,
    $G^{\pi}_j(i_j) = D^{\pi}_j(l_j)$,
    $n = \max\{N, j + 1\}$, back to step 2.
    (II) Otherwise,
    $D^{\pi}_j(l_j) = C^{\pi}_j(l_j)$, back to step 2.
```

If $\mu^{\pi}_{j,i} = \mu^{\pi'}_{j,i} = \mu_{j,j}$ for $j \in \{1, \dots, N\}, i \geq 1$, then policy $\pi = \pi'$, and the lemma is obvious. Otherwise, consider two sample paths ω, ω', where we generate ω and ω' according to Algorithm 1 using policy π and π', respectively. Since $p^{\pi'}_{j,i} = p^{\pi}_{j,i}$, we have $K^{\pi'}_{j,i} = K^{\pi}_{j,i}$ for $j \in \{1, \dots, N\}, i \geq 1$. Let l'_j be the label of the job with the smallest index at the smallest station such that $\mu^{\pi}_{j,l_j} < \mu_{j,j}$. The existence of l'_j is guaranteed since there exists $j \in \{1, \dots, N\}, i \geq 1$ such that $\mu^{\pi}_{j,i} \neq \mu^{\pi'}_{j,i}$. Let i'_j be the corresponding label of successful departures from station j. Then, $G^{\pi'}_{j-1}(l'_j) = G^{\pi}_{j-1}(l'_j)$, $D^{\pi'}_{j-1}(l'_j) = D^{\pi}_{j-1}(l'_j)$. Note that, Algorithm 1 gives priority to the jobs at the later station, that is, it generates the departure times of the jobs at the later station first as long as this job has departured from the previous station. Thus, $D^{\pi'}_{j+1}(i'_j - B_j - 1) = D^{\pi}_{j+1}(i'_j - B_j - 1)$. Then, since $\mu^{\pi}_{j,l_j} < \mu_{j,j}$, we have $C^{\pi'}_j(l'_j) < C^{\pi}_j(l'_j)$, and $D^{\pi'}_j(l'_j) < D^{\pi}_j(l'_j), G^{\pi'}_j(i'_j) \leq G^{\pi}_j(i'_j)$. Proceeding in this manner, we can obtain $D^{\pi'}_j(i) \leq D^{\pi}_j(i)$, and $G^{\pi}_j(i) \leq G^{\pi}_j(i)$ for $j \in \{1, \ldots, N\}, i \geq 1$. $\square$

Proof of Lemma 5.3.2. Next, we prove π^* is better than π' by proving the number of jobs waiting in front of each station under policy π^* is always higher than under policy π. To better illustrate the status of each station in the system, we reformulate the system as a continuous-time Markov chain using a different definition as follows.

For all $\pi \in \Pi$, and $t \geq 0$, let $Y^{\pi}(t) = (Y^{\pi}_1(t), \ldots, Y^{\pi}_{2N-1}(t))$, where $Y^{\pi}_{2j}(t) \in \{0, 1, \ldots, B_j\}$ denotes the number of jobs in the buffer between station j and $j+1$ at time t under policy π for $j = 1, \ldots, N-1$, and $Y^{\pi}_{2j-1}(t) \in \{0, 1, 2\}$ denotes the status of station j at time t under the policy π for $j = 1, \ldots, N$, where $0, 1$, and 2 refer to the starved status, operating status, and blocked status, respectively. Then, $Y^{\pi}(t)$ is a continuous time Markov chain with state space S'. Let $\mu^{\pi}_j(s)$ and $p^{\pi}_j(s)$ be the service rate and defect probability of the server at station $j \in \{1, \ldots, N\}$ in state $s \in S'$ under policy $\pi \in \Pi$. Then, $\mu^{\pi^*}_j(s) = \mu^{\pi'}_j(s) = \mu_{jj}$, and $p_{jj} = p^{\pi^*}_j(s) \leq p^{\pi'}_j(s)$, for $j \in \{1, \ldots, N\}, s \in S'$. We will prove that $Y^{\pi^*}(t) \geq Y^{\pi'}(t)$, for $t \geq 0$.

Consider two sample paths ω^*, ω', where we use policy π^* and π' respectively. Since $\mu^{\pi^*}_j(s) = \mu^{\pi'}_j(s)$ for all $s \in S'$, and exponential distribution is memoryless, we can couple the two sample paths by using common random numbers to generate the service times. Specifically, let t_n, $n \geq 0$, be the time of the nth event (i.e. service completion at some station with or without defects) that happens in any of the two sample paths. Let $t_0 = 0$. For $n \geq 0$, let $s^{\pi}_n \in S'$ denote the state of the system at time t_n under policy π, and let $I^{\pi}_n \subseteq \{1, \ldots, N\}$ denote the set of working stations under policy π for $\pi \in \{\pi', \pi^*\}$, and we generate new service times $\{S_{n,j}\}$ from exponential distribution with rate $\mu_{j,j}$ for all working station $j \in I^{\pi'}_n \cup I^{\pi^*}_n$. Then, $t_{n+1} = t_n + \min\{S_{n,j} : \forall j \in I^{\pi'}_n \cup I^{\pi^*}_n, \}$, and the station with the smallest service time will have a service completion at t_{n+1}. Moreover, right before the time of the next event at some station j_0 at time t_{n+1}, we generate u_{n,j_0} from uniform distribution with range $(0,1)$ as the indicator of either the job is defective or not. If $j_0 \in I_n^{\pi^*}, u_{n,j_0} \geq (<) p_{j_0,j_0}$, then at t_{n+1}, there would be a successful (defective) service completion at station j_0 in ω^*; if $j_0 \in I_n^{\pi'}, u_{n,j_0} \geq (<) p_{j_0}^{\pi'}(s_n^{\pi'})$, then at t_{n+1}, there would be a successful (defective) service completion at station j_0 in ω'. Since $p_{j_0,j_0} \leq p_{j_0}^{\pi'}(s_n^{\pi'})$, if the service completion is defective under policy π^*, it must also be defective under policy π'.

We prove this lemma by induction. Since the system is initially empty, $Y^{\pi^*}(0) = Y^{\pi'}(0) = \{0, \ldots, 0\}$. Assume that $Y^{\pi^*}(t) \geq Y^{\pi'}(t)$ for $t \in [0, t_n)$, and the next event is service completion at station j_1. Then, under policy π^*, for $1 < j_1 < N$,

1. When $j_1 \in I_n^{\pi^*}$, and the completed job is not defective,
   (a) if the buffer right after station j_1 is full, then $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) + e_{2j_1-1}$;
   (b) if the buffer right after station j_1 is not full, station j_1+1 is not starved, and station j_1-1 is blocked, then $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) - e_{2j_1-3} + e_{2j_1}$;
   (c) if station j_1+1 is starved, and station j_1-1 is blocked, then $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) - e_{2j_1-3} + e_{2j_1+1}$;
   (d) if the buffer right after station j_1 is not full, station j_1+1 is not starved, station j_1-1 is not blocked, and the buffer right before station j_1 is empty, then $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) - e_{2j_1-1} + e_{2j_1}$;
   (e) if station j_1+1 is starved, station j_1-1 is not blocked, and the buffer right before station j_1 is empty, then $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) - e_{2j_1-1} + e_{2j_1+1}$;
   (f) if the buffer right after station j_1 is not full, station j_1+1 is not starved, station j_1-1 is not blocked, and the buffer right before station j_1 is not empty, then $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) - e_{2j_1-2} + e_{2j_1}$;
   (g) if station j_1+1 is starved, station j_1-1 is not blocked, and the buffer right before station j_1 is not empty, then $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) - e_{2j_1-2} + e_{2j_1+1}$.
2. When $j_1 \in I_n^{\pi^*}$, and the completed job is defective,

(a) if station $j_1 - 1$ is blocked, then $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) - e_{2j_1-3}$;

(b) if station $j_1 - 1$ is not blocked, and the buffer right before station j_1 is empty, then $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) - e_{2j_1-1}$;

(c) if station $j_1 - 1$ is not blocked, and the buffer right before station j_1 is not empty, then $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) - e_{2j_1-2}$.

3. When $j_1 \notin I_n^{\pi^*}$, then station j_1 is blocked, $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-)$.

For $j_1 = N$, the result of $Y^{\pi^*}(t_n)$ is the same as the three sub-cases under case 2.

For $j_1 = 1$, when $1 \in I_n^{\pi^*}$, and the completed job is not defective,

1. if the buffer right after station 1 is full, $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) + e_1$;

2. if the buffer right after station 1 is not full, station 2 is not starved, $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) + e_2$;

3. if station 2 is starved, $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-) + e_3$;

when station 1 is blocked, or the completed job is defective under π^*, then $Y^{\pi^*}(t_n) = Y^{\pi^*}(t_n-)$. The possible transitions of $Y^{\pi'}(t)$ at t_n can be listed similar to the transitions of $Y^{\pi^*}(t)$ as the cases above, except that $Y^{\pi'}(t_n)$ might be unchanged because of station j_1 being starved. By comparing the state of $Y^{\pi'}(t)$ and $Y^{\pi^*}(t)$ after the transition at t_n for all possible states of $Y^{\pi'}(t)$ and $Y^{\pi^*}(t)$ such that $Y^{\pi^*}(t_n-) \geq Y^{\pi'}(t_n-)$, we can conclude that $Y^{\pi^*}(t) \geq Y^{\pi'}(t)$ for $t \in [t_n, t_{n+1})$. Thus, by induction, we can obtain that $Y^{\pi^*}(t) \geq Y^{\pi'}(t)$, for $t \geq 0$. Since the service rates of all job at all stations are equal in π^* and π', π^* has higher successful rate than π' at the last station, it follows that π' is better than π. $\square$

Alternative Proof of Corollary 5.4.5. For simplicity, when $N = 2$, let $B_1 = B$. The existence of an optimal Markovian stationary deterministic policy has been discussed in the proof of Theorem 5.4.1, we omit it for brevity. We again use Policy Iteration to show this proposition. Note that, the optimal policy we prove here is a special case of δ_1^k with

$k = B + 2$. Thus, we use the notations in the proof of Theorem 5.4.1 but simplify the expressions by letting $s^* = B + 2$. Choose the initial decision $\delta_0 = \delta_1^{B+2}$, then

$$r(s, \delta_0(s)) = \begin{cases} 0 & \text{for } s = 0, \\ \hat{\mu}_{22} & \text{for } 1 \le s \le B + 2. \end{cases}$$

$$p(s'|s, \delta_0(s)) = \begin{cases} \hat{\mu}_{11} & \text{for } 0 \le s \le B + 1, s' = s + 1, \\ \mu_{22} & \text{for } 1 \le s \le B + 2, s' = s - 1, \\ 1 - \hat{\mu}_{11} & \text{for } s = s' = 0, \\ 1 - (\hat{\mu}_{11} + \mu_{22}) & \text{for } 1 \le s \le B + 1, s = s', \\ 1 - \mu_{22} & \text{for } s = s' = B + 2, \\ 0 & \text{otherwise.} \end{cases}$$

Since the policy yields an irreducible Markov chain, we can solve the following equation to find a scalar g_0 and a vector h_0:

$$r_{\delta_0} - g_0 e + (P_{\delta_0} - I)h_0 = 0, \tag{B.1}$$

such that $h_0(0) = 0$, where e is the unit vector and I is the identity matrix.

Then,

$$g_0 = \frac{\hat{\mu}_{22} \sum_{j=0}^{B+1} \hat{\mu}_{11}^{j+1} \mu_{22}^{B+1-j}}{\sum_{j=0}^{B+2} \hat{\mu}_{11}^{j} \mu_{22}^{B+2-j}} = \frac{\Theta_1(B+2)}{\Theta_2(B+2)}.$$

For $\forall s \in S$,

$$h_0(s) = \frac{g_0}{\hat{\mu}_{11}^s} \sum_{j=0}^{s-1} (j+1) \hat{\mu}_{11}^j \mu_{22}^{s-1-j} - \frac{\hat{\mu}_{22}}{\hat{\mu}_{11}^{s-1}} \sum_{j=0}^{s-2} (j+1) \hat{\mu}_{11}^j \mu_{22}^{s-2-j}.$$

For the next step of the policy iteration algorithm, we choose

$$\delta_1(s) \in \arg\max_{a \in A_s} \left\{ r(s,a) + \sum_{j \in S} p(j|s,a) h_0(j) \right\}, \forall s \in S. \tag{B.2}$$

We now show that $\delta_0(s) = \delta_1(s)$ for all $s \in S$. In other words, the following inequality holds for all $s \in S, a \in A_s \setminus \{\delta_0(s)\}$:

$$\epsilon(s,a) = r(s,a) + \sum_{j \in S} p(j|s,a) h_0(j) - (r(s,\delta_0(s)) + \sum_{j \in S} p(j|s,\delta_0(s)) h_0(j)) \leq 0. \tag{B.3}$$

For all $s \in S$, $\delta_0(s) = a_{12}$. Thus, for $0 \leq s < B+2$, we will specify $\epsilon(s,a)$ for actions $\{a_{10}, a_{20}, a_{21}\}$; for $s = B+2$, we will specify $\epsilon(s,a)$ for action a_{21}.

For $s = 0$, station 2 is starved and $a_{i1} = a_{i0}$ for $i = 1, 2$, thus

$$\epsilon(0, a_{21}) = \epsilon(0, a_{20}) = \frac{1}{\hat{\mu}_{11}} (\hat{\mu}_{21} - \hat{\mu}_{11}) g_0 \leq 0.$$

$$\epsilon(0, a_{10}) = 0.$$

For $s = 1, \ldots, B+1$,

$$\epsilon(0, a_{10}) = -\frac{\hat{\mu}_{22}}{\Theta_2(B+2)} \hat{\mu}_{11}^{B+3-s} \sum_{j=0}^{s-1} \hat{\mu}_{11}^{j} \mu_{22}^{s-1-j} \leq 0.$$

$$\epsilon(0, a_{20}) = -\frac{\hat{\mu}_{22}}{\Theta_2(B+2)} (\hat{\mu}_{11} - \hat{\mu}_{21}) \sum_{j=0}^{B+1-s} \hat{\mu}_{11}^{j} \mu_{22}^{B+1-j} + \epsilon(0, a_{10}) \leq 0.$$

For $s = 1, \ldots, B+2$,

$$\begin{aligned}\epsilon(s, a_{21}) = &- \frac{1}{\Theta_2(B+2)}\Big[(\hat{\mu}_{22} - \hat{\mu}_{12}) \sum_{j=B+2-s}^{B+1} \hat{\mu}_{11}^{j+1} \mu_{22}^{B+1-j} \\ &+ (p_{12} - p_{22})\mu_{12} \sum_{j=0}^{B+2-s} \hat{\mu}_{11}^{j} \mu_{22}^{B+2-j} \\ &+ (\hat{\mu}_{11} - \hat{\mu}_{21})\hat{\mu}_{22} \sum_{j=0}^{B+1-s} \hat{\mu}_{11}^{j} \mu_{22}^{B+1-j}\Big] \\ &\leq 0\end{aligned} \tag{B.4}$$

This proves that $\delta_0(s) = \delta_1(s)$ for all $s \in S$. By Theorem 9.5.1 of Puterman [41], this proves that the policy that always assign server j to station j for $j \in \{1, 2\}$ is optimal. $\square$

REFERENCES

[1] H.-S. Ahn, I. Duenyas, and M. E. Lewis, “Optimal control of a two-stage tandem queuing system with flexible servers,” *Probability in the Engineering and Informational Sciences*, vol. 16, no. 4, pp. 453–469, 2002.

[2] H.-S. Ahn, I. Duenyas, and R. Q. Zhang, “Optimal stochastic scheduling of a two-stage tandem queue with parallel servers,” *Advances in Applied Probability*, vol. 31, no. 4, pp. 1095–1117, 1999.

[3] ——, “Optimal control of a flexible server,” *Advances in Applied Probability*, vol. 36, no. 1, pp. 139–170, 2004.

[4] H.-S. Ahn and M. E. Lewis, “Flexible server allocation and customer routing policies for two parallel queues when service rates are not additive,” *Operations Research*, vol. 61, no. 2, pp. 344–358, 2013.

[5] H.-S. Ahn and R. Righter, “Dynamic load balancing with flexible workers,” *Advances in Applied Probability*, vol. 38, no. 3, pp. 621–642, 2006.

[6] S. Andradóttir and H. Ayhan, “Throughput maximization for tandem lines with two stations and flexible servers,” *Operations Research*, vol. 53, no. 3, pp. 516–531, 2005.

[7] S. Andradóttir, H. Ayhan, and D. G. Down, “Server assignment policies for maximizing the steady-state throughput of finite queueing systems,” *Management Science*, vol. 47, no. 10, pp. 1421–1439, 2001.

[8] ——, “Dynamic server allocation for queueing networks with flexible servers,” *Operations Research*, vol. 51, no. 6, pp. 952–968, 2003.

[9] ——, “Dynamic assignment of dedicated and flexible servers in tandem lines,” *Probability in the Engineering and Informational Sciences*, vol. 21, no. 4, pp. 497–538, 2007.

[10] ——, “Maximizing the throughput of tandem lines with flexible failure-prone servers and finite buffers,” *Probability in the Engineering and Informational Sciences*, vol. 22, no. 2, pp. 191–211, 2008.

[11] ——, “Queueing systems with synergistic servers,” *Operations research*, vol. 59, no. 3, pp. 772–780, 2011.

[12] ——, "Design principles for flexible systems," *Production and Operations Management*, vol. 22, no. 5, pp. 1144–1156, 2013.

[13] ——, "Optimal assignment of servers to tasks when collaboration is inefficient," *Queueing Systems*, vol. 75, no. 1, pp. 79–110, 2013.

[14] S. Andradóttir, H. Ayhan, and E. Kırkızlar, "Flexible servers in tandem lines with setup costs," *Queueing Systems*, vol. 70, no. 2, pp. 165–186, 2012.

[15] N. T. Argon and S. Andradóttir, "Partial pooling in tandem lines with cooperation and blocking," *Queueing Systems*, vol. 52, no. 1, pp. 5–30, 2006.

[16] N. T. Argon and Y.-C. Tsai, "Dynamic control of a flexible server in an assembly-type queue with setup costs," *Queueing Systems*, vol. 70, no. 3, pp. 233–268, 2012.

[17] I. Atencia and P. Moreno, "The discrete-time geo/geo/1 queue with negative customers and disasters," *Computers & Operations Research*, vol. 31, no. 9, pp. 1537–1548, 2004.

[18] J. J. Bartholdi III and D. D. Eisenstein, "A production line that balances itself," *Operations Research*, vol. 44, no. 1, pp. 21–34, 1996.

[19] J. A. Buzacott, "Commonalities in reengineered business processes: Models and issues," *Manage. Sci.*, vol. 42, no. 5, pp. 768–782, 1996.

[20] P. Cao and J. Xie, "Optimal control of a multiclass queueing system when customers can change types," *Queueing Systems*, vol. 82, no. 3-4, pp. 285–313, 2016.

[21] C Courcoubetis and M. Reiman, "Optimal control of a queueing system with simultaneous service requirements," *IEEE transactions on automatic control*, vol. 32, no. 8, pp. 717–727, 1987.

[22] D. G. Down, G. Koole, and M. E. Lewis, "Dynamic control of a single-server system with abandonments," *Queueing Systems*, vol. 67, no. 1, pp. 63–90, 2011.

[23] A. Economou and S. Kapodistria, "Synchronized abandonments in a single server unreliable queue," *European Journal of Operational Research*, vol. 203, no. 1, pp. 143–155, 2010.

[24] M Elshafei, M Khan, and S. Duffuaa, "Repeat inspection planning using dynamic programming," *International journal of production research*, vol. 44, no. 2, pp. 257–270, 2006.

[25] T. M. Farrar, "Optimal use of an extra server in a two station tandem queueing network," *IEEE Transactions on Automatic Control*, vol. 38, no. 8, pp. 1296–1299, 1993.

[26] S. D. P. Flapper, J. C. Fransoo, R. A. Broekmeulen, and K. Inderfurth, "Planning and control of rework in the process industries: A review," *Production Planning & Control*, vol. 13, no. 1, pp. 26–34, 2002.

[27] I. Gurvich and J. A. Van Mieghem, "Collaboration and multitasking in networks: Prioritization and achievable capacity," *Management Science*, vol. 64, no. 5, pp. 2390–2406, 2018.

[28] J. J. Hasenbein and B. Kim, "Throughput maximization for two station tandem systems: A proof of the andradóttir–ayhan conjecture," *Queueing Systems*, vol. 67, no. 4, pp. 365–386, 2011.

[29] W. Hopp and M. Van Oyen, "Agile workforce evaluation: A framework for cross-training and coordination," *IIE Transactions*, vol. 36, Jan. 2003.

[30] A. Hordijk and G. Koole, "On the assignment of customers to parallel queues," *Probability in the Engineering and Informational Sciences*, vol. 6, no. 4, 495–511, 1992.

[31] T. Işık, S. Andradóttir, and H. Ayhan, "Optimal control of queueing systems with non-collaborating servers," *Queueing Systems*, vol. 84, no. 1, pp. 79–110, 2016.

[32] E. Kırkızlar, S. Andradóttir, and H. Ayhan, "Robustness of efficient server assignment policies to service time distributions in finite-buffered lines," *Naval Research Logistics (NRL)*, vol. 57, no. 6, pp. 563–582, 2010.

[33] ——, "Flexible servers in understaffed tandem lines," *Production and Operations Management*, vol. 21, no. 4, pp. 761–777, 2012.

[34] ——, "Profit maximization in flexible serial queueing networks," *Queueing Systems*, vol. 77, no. 4, pp. 427–464, 2014.

[35] A. Krishnamoorthy, P. K. Pramod, and S. R. Chakravarthy, "Queues with interruptions: A survey," *Top*, vol. 22, no. 1, pp. 290–320, 2014.

[36] E. J. Lodree, N. Altay, and R. A. Cook, "Staff assignment policies for a mass casualty event queuing network," *Annals of Operations Research*, pp. 1–32, 2017.

[37] E. Özkan and J. P. Kharoufeh, "Optimal control of a two-server queueing system with failures," *Probability in the Engineering and Informational Sciences*, vol. 28, no. 4, pp. 489–527, 2014.

[38] D. G. Pandelis and M. P. Van Oyen, “Sample path optimal policies for serial lines with flexible workers,” *Journal of Applied Probability*, vol. 49, no. 2, pp. 582–589, 2012.

[39] I. Papachristos and D. G. Pandelis, “Optimal dynamic allocation of collaborative servers in two station tandem systems,” *IEEE Transactions on Automatic Control*, vol. 64, no. 4, pp. 1640–1647, 2018.

[40] L. Perron, “Planning and scheduling teams of skilled workers,” *Journal of Intelligent Manufacturing*, vol. 21, no. 1, pp. 155–164, 2010.

[41] M. L. Puterman, “Markov decision processes: Discrete stochastic dynamic programming,” 1994.

[42] R. Qin, D. A. Nembhard, and W. L. Barnes II, “Workforce flexibility in operations management,” *Surveys in Operations Research and Management Science*, vol. 20, no. 1, pp. 19–33, 2015.

[43] S. Tekin, S. Andradóttir, and D. G. Down, “Dynamic server allocation for unstable queueing networks with flexible servers,” *Queueing Systems*, vol. 70, no. 1, pp. 45–79, 2012.

[44] R. H. Teunter and S. D. P. Flapper, “Lot-sizing for a single-stage single-product production system with rework of perishable production defectives,” *Or Spectrum*, vol. 25, no. 1, pp. 85–96, 2003.

[45] D. Towsley and S. K. Tripathi, “A single server priority queue with server failures and queue flushing,” *Operations Research Letters*, vol. 10, no. 6, pp. 353–362, 1991.

[46] Y.-C. Tsai and N. Argon, “Dynamic server assignment policies for assembly-type queues with flexible servers,” *Naval Research Logistics (NRL)*, vol. 55, pp. 234 – 251, Apr. 2008.

[47] M. P. Van Oyen, E. G. Gel, and W. J. Hopp, “Performance opportunity for workforce agility in collaborative and noncollaborative work systems,” *IIE Transactions*, vol. 33, no. 9, pp. 761–777, 2001.

[48] X. Wang, S. Andradóttir, and H. Ayhan, “Dynamic server assignment with task-dependent server synergy,” *IEEE Transactions on Automatic Control*, vol. 60, no. 2, pp. 570–575, 2014.

[49] M. H. Yarmand and D. G. Down, “Server allocation for zero buffer tandem queues,” *European Journal of Operational Research*, vol. 230, no. 3, pp. 596–603, 2013.

[50] ——, “Maximizing throughput in zero-buffer tandem lines with dedicated and flexible servers,” *IIE Transactions*, vol. 47, no. 1, pp. 35–49, 2015.

[51] U. Yechiali, “Queues with system disasters and impatient customers when system is down,” *Queueing Systems*, vol. 56, no. 3-4, pp. 195–202, 2007.

[52] G. Zayas-Cabán and H.-S. Ahn, “Dynamic control of a single-server system when jobs change status,” *Probability in the Engineering and Informational Sciences*, vol. 32, no. 3, pp. 353–395, 2018.

VITA

Junqi Hu was born in Jiayu, Hubei Province, China in December 1991. She received a B.A. in Economics and a B.S. in Mathematics from Wuhan University, China. She started her graduate studies at the Georgia Institute of Technology, Atlanta, GA, in 2014. She received her M.S. degree in Operations Research in 2016 and her Ph.D. degree in Operations Research in 2020 under the supervision of Dr. Sigrún Andradóttir and Dr. Hayriye Ayhan. Her research interests lie in dynamic control of manufacturing and service systems, Markov decision processes, and queueing networks.