12-2008

Optimal and Heuristic Resource Allocation Policies in Serial Production Systems

Ramesh Arumugam
Clemson University, rarumug@clemson.edu

Optimal and Heuristic Resource Allocation Policies in Serial Production Systems

---

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Industrial Engineering

---

by
Ramesh Arumugam
December 2008

Accepted by:
Dr. Maria E. Mayorga and Dr. Kevin M. Taaffe, Committee Chair
Dr. William G. Ferrell

Abstract

We have studied the optimal server allocation policies for a tandem queueing system under different system settings. Motivated by an industry project,we have studied a two stage tandem queueing system with arrival to the system and having two flexible servers capable of working at either of the stations. In our research, we studied the system under two different circumstances: modeling the system to maximize throughput without cost considerations, modeling the system to include switching and holding costs along with revenue for finished goods. In the maximizing throughput scenario, we considered two different types of server allocations: collaborative and non-collaborative. For the collaborative case, we identified the optimal server allocation policies for the servers and have proved the structure of the optimal server allocation policy using mathematical iteration techniques. Moreover, we found that, it is optimal to allocate both the servers together all the time to get maximum throughput. In the non-collaborative case, we have identified the optimal server allocation policies and found that it is not always optimal to allocate both the servers together.

With the inclusion of costs, we studied the system under two different scenarios: system with switching costs only and system having both switching and holding costs. In both the cases, we have studied the optimal server allocation policies for the servers. Due to the complicated structure of the optimal server allocation policy, we have studied three different heuristics to approximate the results of the optimal policy. We found that the performance of one of the heuristics is very close to the optimal policy values.

Dedication

For my family and friends, who offered unconditional love and support throughout the course of my Masters program.

Acknowledgments

I would like to sincerely thank both my advisors Dr. Maria Mayorga and Dr. Kevin Taaffe for their constant support and advice in helping me complete my Masters program with a thesis. I am very thankful to them to let me do real time industry projects along with my course work and research work during the past two years. Their wide knowledge in my research arena has been a great value to me and have made my learning objective complete. I would also like to thank Dr. William Ferrell for being one of my committee member and providing me with key inputs. I would like to thank Dr. Mary Beth Kurz for her valuable suggestions to make my thesis data processing easier.

I would like thank my department head Dr. Anand Gramopadhye and other faculties and staff of the Industrial Engineering Department who helped me in various times.

I kindly thank my friends Mr. Ping-nan Chiang, Ms. Lisa Bosman and Mr. Vinoth Sankaran for their friendly help and support throughout the course of my Masters program.

Finally, I would like to thank everybody who made my Life at Clemson productive, useful and fun.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In today's world, running an industry with maximum efficiency is a challenging task. When there are many resources like people and machines in an industrial setting, allocating them optimally to get the maximum output or minimum cost is a difficult task amongst other financial barriers. Most of the industries have traditional methods for allocating these resources and carrying out the regular production process in addition to the lack of knowledge on optimal allocations. We identified one such inefficient server allocation in one of the industrial projects we did with TaylorMade Adidas Golf.

In TaylorMade Adidas Golf, we modeled a 5 step serial production system. In this system, one of the resources (servers) was shared at two stations. The server starts working at one of the stations and moves to the other station to process the jobs in queue depending on the requirement at that station. This server was moving from one station to another at a random time by rule of thumb leading to inventory built up at each of the stations. As we modeled the production process of TaylorMade to address various capacity issues in their system, we started noticing this inefficiency in the system. Therefore, our motivation to dive into this research started with this industry project.

Thereafter, we started studying a simplified system of the above kind. Our initial motivation was to model the system to enable us to study the different characteristics of the system like throughput, utilization, queue length etc. We considered two stations and two flexible servers capable of working at either of the stations and studied the server allocation policies for the servers to maximize the number of finished products out of the system. We also considered an arrival to the system to mimic the actuality. We used dynamic programming technique to model and analyze the server allocation policies across this tandem queueing system. We have described our detailed research in this area in Chapter 1.

We extended our research to model the server movement across stations more realistically. In the real world, whenever a server moves from one station to another, there is a loss of production time or some sort of cost associated with it affecting the total output of the system. Therefore, we introduced a switching cost into the model for whenever a server switches from one station to another. Along with that, we included holding costs for the number of jobs waiting in the system. Also, when the finished products leave the system, there is a reward associated with it. So, in our extension, we studied the server allocation policies by minimizing all the costs associated with the system including the positive reward. This scenario is the same as maximizing the throughput with negative costs. Our chapter 2 describes our research extension in detail.

We reviewed the literature to look for the past research in the arena of server allocation to stations in maximizing throughput (initial motivation) and minimizing cost (research extension). Extensive literature review on each of the topics is present in their respective chapters. Also, the individual chapters describe the mathematical formulation, analysis, results and conclusions in detail. Unlike past research, we have studied the optimal server allocation policy for different scenarios and have also shown the policy structure of all the cases studied. These policy structures are very useful to industries in deciding their server allocation policies. Due to the complicated structure of some of the policies, we studied some heuristic policies to substitute the optimal server allocation policies. These heuristic policies are simple enough to make it implementable in the real world. This makes our research useful and valuable to various industries and researchers.

Chapter 2

Inventory Based Allocation Policies for Flexible Servers in Serial Systems with no costs

2.1 Introduction

Effectively managing a production system is not an easy task. Operational considerations in industry today include workstation cycle time, server requirements, and process flow, among others. With so many factors to consider, firms often implement a customized production policy. Unfortunately, such policies are often based on anecdotal experience or are legacy policies which, while they may have worked well for a previous setting, do not work well as system parameters change. Such a phenomenon was observed in the industry example that motivated the study of our problem. In this firm the server allocation policy across machines appeared very inefficient. Since the firm was not experiencing extremely high product demand, this inefficiency was masked. As business rules change and resource limitations are updated, firms must continue to re-evaluate their production policy to ensure they are achieving the desired objective. We study a firm that would like to maximize product throughput with their current set of resources. These resources have been cross-trained to perform jobs at various workstations. In this paper, we limit our discussion to the two station case.

Consider a two stage serial production system in which each stage requires at least one server to perform the job. The servers are flexible, i.e., they can work on jobs at either station, and their processing rates do not depend on each other. There are two cases for how jobs are processed: collaborative performance, and non-collaborative performance. The collaborative case assumes that the servers will work together on a single job at either of the stations. The non-collaborative case assumes that both of the servers will work on individual jobs when residing at the same workstation. Moreover, there is a finite arrival rate to the system, and there is infinite room after the second station to store finished product. We also assume that travel times and costs required for servers to move from one station to another station are negligible. We use a manufacturing blocking mechanism to account for product queueing at each station. For the two queueing systems of the form described above, we are interested in determining the dynamic server assignment policy that maximizes the long-run average throughput. In this paper, we will present structural analysis and numerical results for the case in which servers work collaboratively. After a detailed treatment of collaborative servers, we will introduce the non-collaborative case and provide insights based on numerical results only.

A significant amount of literature exists on static server assignment problems. In such cases, the server assignment is permanent and is done ex-ante i.e., there is no dynamic server allocation. In the context of static server assignment, different station arrangements along with the server allocation in a serial production line have been studied. Particular research of this interest can be found in Yamazaki, Sakasegawa and Shanthikumar [14] and Hillier and So [16]. Yamazaki et al. [14] have found that the station arrangements in the serial production line determine the throughput of the system. Hillier and So [16] have simultaneously optimized server and work allocation to maximize throughput. Thereafter, considerable research on server allocation has continued on both parallel and serial systems. More specifically, research has expanded to include dynamic server allocation policies and tandem queueing systems.

Much of the existing work in the area of optimal dynamic assignment of servers assumes a production system with parallel queues. A common objective when considering an optimal assignment of servers to multiple interconnected queues is to minimize holding costs. Under a heavy traffic assumption, Harrison and Lopez [15], Williams [38], Bell and Williams [35], and Mandelbaum and Stoylar [23] have developed optimal server-assignment policies that minimize infinite horizon holding costs for variants of systems with parallel queues, flexible servers and outside arrivals. Under similar assumptions, Squillante, Xia, Yao and Zhang [36] used simulation to study the policies for a parallel queueing system. Farrar [12] and Pandelis and Teneketzes [29] studied the optimal allocation of servers to stations with two queues and no arrivals thereby minimizing holding costs. In both the papers, they were concerned with scheduling the jobs that were initially present in the system assuming zero arrivals to the system.

Additionally, Rosberg, Varaiya, and Walrand [32], Hajek [8], and Ahn, Duenyas and Lewis [1] have studied the service allocation to minimize holding costs in a two-station setting with Poisson arrivals. Van Oyen and Teneketzis [28] provided an optimal policy for the system where identical servers can work collaboratively to reduce the cycle time per job. However, Van Oyen and Teneketzis [28], Reiman and Wein [31], and Buzacott [10] were concerned with static assignment of collaborative servers. Other related works in this area have been completed by Ross, Wein and Reiman [31], Oyen, Gel and Hopp [27], Irvani, Posner and Buzacott [19], Pandelis and Teneketzes [29], and Duenyas, Gupta and Olsen [11].

In addition to the parallel queueing systems, researchers have addressed the system configuration with serial queues (or workstations) with the objective of minimizing cost. Ostolaza, Thomas and McClain [25] and McClain, Thomas, and Sox [24] studied dynamic line balancing in tandem systems, where shared tasks can be performed at either of two successive stations. Their work was extended by Zavadlav, McClain, and Thomas [39], who studied several server assignment policies for systems with fewer servers than machines, where cross-functional working capabilities were assumed for the servers. Also, Ahn, Duenyas and Zhang [2] have studied the optimal allocation of servers for a two-class, two stage tandem queueing system with parallel servers and no arrivals. Ahn, Duenyas and Lewis [1] have studied the optimal allocation of servers for a two-class, two stage tandem queueing system with one dedicated server, one flexible server and with external arrivals to the system. Both these papers characterized the policies that minimized holding costs.

To the best of our knowledge, there are a limited number of works that study server allocation policies that *maximize throughput* in serial systems. Bischak [9] proposed a server assignment policy in which the servers can move between stations in a U-shaped module. Considering high processing time variation, it was shown with the use of simulation that higher throughput can be achieved for this system with servers moving across stations than the system with fixed servers for each station. Recently, Andradottir, Ayhan and Down [4, 5] and Andradottir and Ayhan [3] are the have studied the dynamic assignment of servers to maximize the long-run average throughput in queueing networks with flexible servers. In each case, it was assumed that an infinite supply of jobs was available in front of the first station. Andradottir et al. [4] characterized the optimal dynamic server assignment policy for a two-stage finite tandem queue with two servers. They also presented a heuristic for the case in which there are an equal number of servers and stations. Andradottir and Ayhan [3] studied the allocation of M servers to two tandem stations and characterized the optimal long term average throughput. Also, they developed heuristic server assignment policies for systems with three or more servers. Lastly, Andradottir et al. [5] proved that the optimal throughput is attained by having all servers flexible rather than having dedicated servers to stations. All three papers [3, 4, 5] assume that servers work collaboratively, i.e., all servers can work simultaneously on a single job. Our research also focuses on maximizing throughput. We focus on dynamic server allocation across a two-station tandem line with two flexible servers. Our work differs from previous research in that we allow for finite buffers in front of both stations and external arrivals to the system.

2.2 Model Description for the Collaborative Case

We consider a two-station tandem queueing system. All products arrive to the system at station 1 according to a Poisson process with rate $\lambda > 0$. Each arriving product should be processed by both of the stations in series before leaving the system as a finished good. We assume that there are finite buffers, B_1 and B_2, in front of stations 1 and 2, respectively. Service at each of the stations can be performed by either of two flexible servers. The service time of each server i ($i = 1, 2$) is exponentially distributed with rate μ_i. Without loss of generality, we assume that $\mu_1 \geq \mu_2$. We begin by studying the allocation policy for servers that can work collaboratively. (The state representation of the system will remain the same when we address the non-collaborative case in Section 2.5). A representation of the system is provided in Figure 3.1.

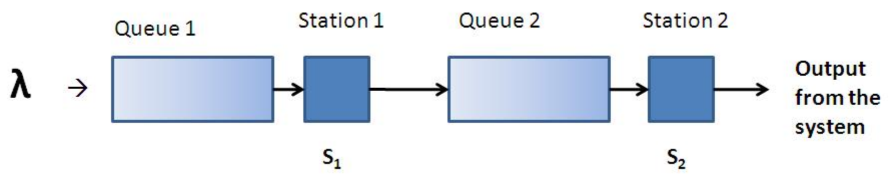

Figure 2.1: Model description

Denote B_1 and B_2 as the buffer sizes in front of the first and second stations, respectively. We define the following state variables:

$X(t) =$Status of station 1 at time t, $X(t) \in \{0, 1, 2\}$,

where 0, 1, 2 represent station 1 being idle, working, or blocked, respectively;

$N_1(t) =$Number in queue (not including those in service) at station 1 at time t,

$N_1(t) \in \{0, 1, 2, ...B_1\}$,

$N_2(t) =$Total number in queue and in service at station 2 at time t,

$N_2(t) \in \{0, 1, 2, \dots B_2 + 1\}$.

Note that not all combinations of states are part of the feasible region, for example, when station 1 is idle, there can be no job waiting in queue 1; similarly station 1 can only be blocked when the buffer at station 2 is full. The state space is then given by $\boldsymbol{\mathcal{X}} = \{(i, j, k)|i \in \{0, 1, 2\}, j \in \{0, 1, \dots, B_1\}, k \in \{0, 1, \dots, B_2 + 1\}$, if $i = 0$ then $j = 0$, if $i = 2$ then $k = B_2 + 1\}$.

Let Π denote the set of Markov deterministic policies, and suppose that for any $u \in \Pi$ and $t \geq 0$, that $(X^u(t), N_1^u(t), N_2^u(t))$ represent the state of the system under policy u at time t. At time t, the production manager is able to control the allocation of servers to stations. The different possible actions are: both servers at station 1; server 1 at station 1 and server 2 at station 2; server 2 at station 1 and server

1 at station 2; and both servers at station 2. Call this the server allocation decision $u(t)$, and denote the four actions by 0, 1, 2, and 3, respectively. A control policy u specifies the action taken at any time given the state of the system. Given a Markov deterministic policy u, $(X^u(t), N_1^u(t), N_2^u(t))$ is a continuous time Markov chain. We are interested in maximizing the system long-run average throughput. Most of the literature tackles throughput problems by analyzing the departure process (a counting process) and then using policy iteration to find the optimal policy. Instead we use renewal reward theory (Ross [34]) to solve an equivalent value iteration problem, where a reward is incurred every time a job departs the system.

Table 2.1: Server allocation policies for the collaborative case.

Policy (u)	Service rate at Station 1 (s_1)	Service rate at Station 2 (s_2)
0	$\mu_1 + \mu_2$	0
1	μ_1	μ_2
2	μ_2	μ_1
3	0	$\mu_1 + \mu_2$

Since the state transitions are Markovian, it suffices to consider policies where decisions are made at discrete time epochs. Thus, instead of a continuous time control problem, we apply uniformization in the spirit of Lippman [22] and solve an equivalent discrete time problem. Define γ as the maximum rate of transitions: $\gamma = \mu_1 + \mu_2 + \lambda$. Without loss of generality, we scale δ (the discount factor) and γ such that $\delta + \gamma = 1$. Starting with the collaborative case, we begin by describing the optimality equation for the discrete-time equivalent finite horizon discounted reward problem. Later we show that our results extend to the infinite horizon and average reward problems. For the collaborative case, the servers work together (or collaboratively) on a single job. That is, if both servers are at the same station, they process a part at a rate of $(\mu_1 + \mu_2)$. Let s_1 and s_2 denote the service rates at stations 1 and 2, respectively, based on the chosen server allocation policy u. The different policies result in the following service rates at each station, presented in Table 3.1. Define $v_k(x, n_1, n_2)$ to denote the value of being in state (x, n_1, n_2) in the k^{th} iteration, and let $v_0(x, n_1, n_2) = 0$ for all x, n_1, n_2. Then the finite horizon (n-stage) discounted reward problem starting in state (x, n_1, n_2) is written as follows:

$$v_{k+1}(x, n_1, n_2) = \max_{(s_1,s_2)\in S} \left[\lambda T_a v_k(x, n_1, n_2) + s_1 T_{p_1} v_k(x, n_1, n_2) + s_2 T_{p_2} v_k(x, n_1, n_2)\right], \tag{2.1}$$

where $k < n$ and $S = \{(\mu_1 + \mu_2, 0), (\mu_1, \mu_2), (\mu_2, \mu_1), (0, \mu_1 + \mu_2)\}$. In the above equation T_a, T_{p_1}, and T_{p_2} are operators corresponding to arrival, production at station 1, and production at station 2, transitions, respectively. The operators are defined as:

$$T_a v(x, n_1, n_2) = \begin{cases} v(x, n_1, n_2) & \text{if} \quad n_1 = B_1 \\ v(x, n_1 + 1, n_2) & \text{otherwise} \end{cases}$$

$$T_{p_1} v(x, n_1, n_2) = \begin{cases} v(x, n_1, n_2) & \text{if} \quad x = 0 \text{ or } 2 \\ v(0, n_1, n_2 + 1) & \text{if} \quad x = 1, n_1 = 0, n_2 < B_2 + 1 \\ v(1, n_1 - 1, n_2 + 1) & \text{if} \quad x = 1, n_1 \geq 1, n_2 < B_2 + 1 \\ v(2, n_1, n_2) & \text{if} \quad x = 1, n_1 \geq 0, n_2 = B_2 + 1 \end{cases}$$

$$T_{p_2} v(x, n_1, n_2) = \begin{cases} v(x, n_1, n_2) & \text{if} \quad x = 0 \text{ or } n_2 = 0 \\ v(1, n_1, n_2 - 1) + r & \text{if} \quad x = 1, n_1 \geq 0, n_2 \leq B_2 + 1 \\ v(1, n_1 - 1, n_2 - 1 + 1) + r & \text{if} \quad x = 2, n_1 \geq 1, n_2 = B_2 + 1 \\ v(0, n_1, n_2 - 1 + 1) + r & \text{if} \quad x = 2, n_1 = 0, n_2 = B_2 + 1 \end{cases}$$

Note that $r = 1$ since this implies that a reward is received every time that a finished good leaves the system. However, we have retained it as 'r' in the formulation for ease of exposition.

When queue 1 is full (reached the limit of B_1), the state does not change if an arrival event occurs. But if queue 1 is not full, an arrival event causes a transition from (x, n_1, n_2) to state $(x, n_1 + 1, n_2)$. Similarly, if station 1 is idle, a service completion event at station 1 does not have any state transition. Depending on the number of jobs in queue 2, however, a service completion at station 1 may involve a state transition. If queue 2 is less than full, a service completion event at station 1 will cause a state transition to $(x, n_1 - 1, n_2 + 1)$. If queue 2 is full, a service completion event at station 1 will cause a state transition to the blocked state $(2, n_1, n_2)$. A state tranisition resulting from a service 2 completion will depend on the size of queue 2, and the number in queue and service at station 1.

Since for each policy u, the pair of service rates at each station is unique, we can rewrite (2.1) in terms of a single control s, where s represents the service rate at station 2 (and $\mu_1 + \mu_2 - s$ represents the service rate at station 1).

$$\begin{aligned} v_{k+1}(x, n_1, n_2) = \max_{s \in S'} [&\lambda T_a v_k(x, n_1, n_2) + (\mu_1 + \mu_2 - s) T_{p_1} v_k(x, n_1, n_2) \\ &+ (s) T_{p_2} v_k(x, n_1, n_2)], \end{aligned}$$

where $S' = \{0, \mu_2, \mu_1, \mu_1 + \mu_2\}$. Without loss of generality, we have assumed that $\mu_1 \geq \mu_2$. Notice that this implies that the service rate at station 2 is increasing in s.

2.3 Optimal Server Allocation Policy in the Collaborative Case

Our first result shows that when maximizing throughput it is optimal for both servers to always work together, either at station 1 or at station 2. This result is summarized in Theorem 2.3.1.

Theorem 2.3.1 *In the collaborative case, there exists an optimal server allocation policy in which both servers are always either assigned to station 1 ($s = 0$) or to station 2 ($s = \mu_1 + \mu_2$).*

Proof 1 *Let s^*_{k+1} denote the optimal server rate at station 2 at the $(k+1)^{st}$ iteration. Then,*

$$\begin{aligned}
s^*_{k+1}(x, n_1, n_2) &= \underset{s\in S'}{argmax}[\lambda T_a v_k(x, n_1, n_2) + (\mu_1 + \mu_2 - s)T_{p_1} v_k(x, n_1, n_2) \\
&\qquad + sT_{p_2} v_k(x, n_1, n_2)] \\
&= \underset{s\in S'}{argmax}\,[\lambda T_a v_k(x, n_1, n_2) + (\mu_1 + \mu_2)T_{p_1} v_k(x, n_1, n_2) \\
&\qquad + s[T_{p_2} v_k(x, n_1, n_2) - T_{p_1} v_k(x, n_1, n_2)]] \\
&= \underset{s\in S'}{argmax}\,\{s[T_{p_2} v_k(x, n_1, n_2) - T_{p_1} v_k(x, n_1, n_2)]\} \\
\text{Let } f_k(x, n_1, n_2) &= [T_{p_2} v_k(x, n_1, n_2) - T_{p_1} v_k(x, n_1, n_2)] \\
\text{Then } s^*_{k+1}(x, n_1, n_2) &= \underset{s\in S'}{argmax}\{s\ f_k(x, n_1, n_2)\}
\end{aligned}$$

Clearly, if $f_k < 0$, then $s^*_{k+1} = 0$, while if $f_k > 0$, then $s^*_{k+1} = \mu_1 + \mu_2$. When $f_k = 0$ all policies are optimal. Therefore, $\forall\ k \geq 0$, there exists $s^*_k(x, n_1, n_2) \in \{0, \mu_1 + \mu_2\}$.$\square$

From this theorem it follows that there always exists an optimal policy in which we allocate both servers to the same workstation. That is, depending on the current state of the system, the optimal policy alternates between these two options, and any policy in which the servers do not work together can be ignored.

Next we discuss the structure of the optimal policy. Based on Theorem 2.3.1, we can provide some very intuitive results. For example, if there is no product at station 2 ($n_2 = 0$), then we should place the servers at station 1; similarly, if there are no products at station 1 or station 1 is blocked ($x = \{0, 2\}$) then we should place the servers at station 2. In the case that station 1 is busy ($x = 1$), we expect that as the workload to a station increases, we assign more servers to that station. This is in fact the case, as depicted by the example in Figure 2.2. We see that the optimal policy depends on the inventory level at each of the stations.

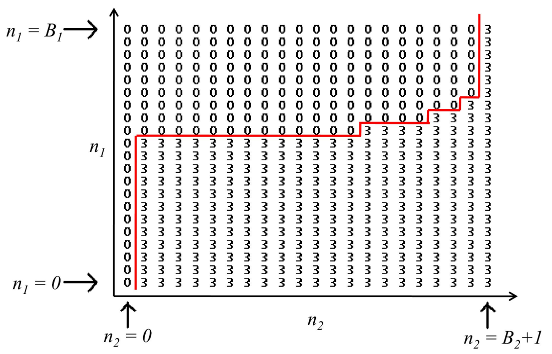

Figure 2.2: Optimal server allocation policy u when $x = 1, \lambda = 3, \mu_1 = \mu_2 = 4, B_1 = B_2 = 20$ and $r = 1$

In the remainder of this section we will show that the optimal reward to-go function $v_k(x, n_1, n_2)$ satisfies a set of properties and that there exists a structured optimal policy for all finite horizon problems. To do this, we define ϑ to be the set of functions from our state space $\boldsymbol{\mathcal{X}}$ onto $\Re$ satisfying the following properties.

Properties of the set ϑ

1. $(a) v_k(x, n_1, n_2) + r \geq v_k(x, n_1, n_2 + 1), \qquad \forall x, n_1, 1 \leq n_2 \leq B_2 + 1;$

   $(b) v_k(1, B_1, 1) \geq v_k(1, B_1, 0)$
2. $(a) v_k(1, n_1, n_2 + 1) \geq v_k(1, n_1 + 1, n_2), \qquad n_1 \geq 1;$

   $(b) v_k(0, 0, n_2 + 1) \geq v_k(1, 0, n_2), \qquad n_2 < B_2 + 1;$
3. $(a) v_k(0, 0, B_2 + 1) + r \geq v_k(2, 0, B_2 + 1)$

   $(b) v_k(1, n_1, B_2 + 1) + r \geq v_k(2, n_1 + 1, B_2 + 1), \qquad n_1 \geq 1;$
4. Submodularity in n_1

   $$(a) v_k(1, n_1, n_2 - 1) - v_k(1, n_1 - 1, n_2 + 1) \geq$$
   $$v_k(1, n_1 + 1, n_2 - 1) - v_k(1, n_1, n_2 + 1), \qquad n_1 \geq 1, 1 \leq n_2 < B_2 + 1;$$
   $$(b) v_k(1, n_1, B_2) - v_k(2, n_1, B_2 + 1) \geq$$
   $$v_k(1, n_1 + 1, B_2) - v_k(2, n_1 + 1, B_2 + 1), \qquad \forall n_1;$$
5. Submodularity in n_2

   $$(a) v_k(1, n_1, n_2 - 1) - v_k(1, n_1 - 1, n_2 + 1) \leq$$
   $$v_k(1, n_1, n_2) - v_k(1, n_1 - 1, n_2 + 2), \qquad n_1 \geq 1, 1 \leq n_2 < B_2;$$
   $$(b) v_k(1, 0, n_2 - 1) - v_k(0, 0, n_2 + 1) \leq$$
   $$v_k(1, 0, n_2) - v_k(0, 0, n_2 + 2). \qquad 1 \leq n_2 < B_2;$$

The above properties have intuitive meaning. Property 1 implies that it is better to produce (and earn a reward when possible) now than to do nothing. Property 2 states that it is better to push products out of station 1 than to do nothing. Property 3 implies that if station 1 is blocked, it is better to send servers to station 2 (since they are of no use at station 1). Lastly, Properties 4 and 5 state that the marginal gain in reward between pushing product out of station 2 or out of station 1 is non-increasing in n_1 and non-decreasing in n_2. Note that the set ϑ is closed under addition and scalar multiplication. Also note that $v_0(x, n_1, n_2) = 0 \in \vartheta$, $\forall (x, n_1, n_2) \in \boldsymbol{\mathcal{X}}$.

We prove the structure of the optimal policy in several steps. In the first step (Theorem 2.3.2), we will show that if $v_k \in \vartheta$ then, there exists a structured optimal policy for the $k+1$ stage problem. In the second step (Theorem 2.3.3), we will show that if $v_k \in \vartheta$ then $v_{k+1} \in \vartheta$ $\forall k$. Finally, we will show in Theorem 2.3.4 that together these results imply that there exists a structured optimal policy for the finite horizon, infinite horizon, and average reward problems.

Theorem 2.3.2 *If $v_k \in \vartheta$, there exists an optimal policy for the k+1 period problem described as follows.*

1. $s^*_{k+1}(0, 0, n_2) = \mu_1 + \mu_2, \qquad \forall n_2 \in \boldsymbol{\mathcal{X}}$
2. $s^*_{k+1}(x, n_1, 0) = 0, \qquad \forall x, n_1 \in \boldsymbol{\mathcal{X}}$
3. $s^*_{k+1}(2, n_1, B_2 + 1) = \mu_1 + \mu_2, \qquad \forall n_1 \in \boldsymbol{\mathcal{X}}$
4. s^*_{k+1} is non-increasing in n_1,

   i.e. $s^*_{k+1}(1, n_1, n_2) \geq s^*_{k+1}(1, n_1 + 1, n_2), \quad \forall n_1, n_2 \in \boldsymbol{\mathcal{X}}$
5. s^*_{k+1} is non-decreasing in n_2,

   i.e. $s^*_{k+1}(1, n_1, n_2) \leq s^*_{k+1}(1, n_1, n_2 + 1), \quad \forall n_1, n_2 \in \boldsymbol{\mathcal{X}}$

Proof 2 *From Theorem 2.3.1, recall that $f_k(x, n_1, n_2) \leq 0$ implies $s^*_{k+1} = 0$ and $f_k(x, n_1, n_2) \geq 0$ implies $s^*_{k+1} = \mu_1 + \mu + _2$. We use our knowledge about $f_k(x, n_1, n_2)$ in determining the optimal policy.*

<u>Part 1:</u> *If suffices to show that $f_k(0, 0, n_2) = T_{p_2} v_k(0, 0, n_2) - T_{p_1} v_k(0, 0, n_2) \geq 0$. To see this note that*

$$f_k(0, 0, n_2) = \begin{cases} v_k(0, 0, 0) - v_k(0, 0, 0), & \textit{if } n_2 = 0; \\ v_k(0, 0, n_2 - 1) + r - v_k(0, 0, n_2), & \textit{otherwise.} \end{cases}$$

If $n_2 = 0$ then $f_k = 0$, otherwise it follows from Property 1(a) that $f_k \geq 0$.

Part 2: *If suffices to show that $f_k(x, n_1, 0) = T_{p_2} v_k(x, n_1, 0) - T_{p_1} v_k(x, n_1, 0) \leq 0$. To see this note that*

$$f_k(x, n_1, 0) = \begin{cases} v_k(0,0,0) - v_k(0,0,0), & \text{if } x = 0, n_1 = 0; \\ v_k(1, n_1, 0) - v_k(1, n_1 - 1, 1), & \text{if } x = 1, n_1 \geq 1; \\ v_k(1,0,0) - v_k(0,0,1), & \text{if } x = 1, n_1 = 0. \end{cases}$$

In the last two cases, $f_k \leq 0$ from Property 2, and in the first case $f_k = 0$.

Part 3: *If suffices to show that $f_k(2, n_1, B_2+1) = T_{p_2} v_k(2, n_1, B_2+1) - T_{p_1} v_k(2, n_1, B_2+1) \geq 0$.*

$$f_k(2, n_1, B_2 + 1) = \begin{cases} v_k(1, n_1 - 1, B_2 + 1) + r - v_k(2, n_1, B_2 + 1), & \text{if } n_1 \geq 1; \\ v_k(0, 0, B_2 + 1) + r - v_k(2, 0, B_2 + 1), & \text{if } n_1 = 0. \end{cases}$$

We see from the above equations that, in both cases, $f_k \geq 0$ from Property 3.

Part 4: *To prove that $s^*_{k+1}(1, n_1, n_2) \geq s^*_{k+1}(1, n_1 + 1, n_2)$ it suffices to show that if $f_k(1, n_1, n_2) \leq 0$, then $f_k(1, n_1+1, n_2) \leq 0$. This implies that if $s^*_k(1, n_1, n_2) = 0$, then $s^*_k(1, n_1 + 1, n_2) = 0$. Therefore, suppose, $f_k(1, n_1, n_2) = T_{p_2} v_k(1, n_1, n_2) - T_{p_1} v_k(1, n_1, n_2) \leq 0$. We separate the state space into three regions to prove Part 4.*

1. *If $n_1 \geq 1, 1 \leq n_2 < B_2 + 1$, then*

$$\begin{aligned} f_k(1, n_1, n_2) &= v_k(1, n_1, n_2 - 1) + r - v_k(1, n_1 - 1, n_2 + 1) \\ &\geq v_k(1, n_1 + 1, n_2 - 1) + r - v_k(1, n_1, n_2 + 1) = f_k(1, n_1 + 1, n_2), \end{aligned}$$

where the inequality follows from Property 4(a). Thus, $f_k(1, n_1+1, n_2) \leq 0$.

2. *If* $n_1 \geq 1$ *and* $n_2 = 0$, *then by definition,* $f_k(1, n_1 + 1, 0) = v_k(1, n_1 + 1, 0) - v_k(1, n_1, 1)$. *Therefore, from Property 2(a), we directly see that* $f_k(1, n_1 + 1, 0) \leq 0$.

3. *If* $n_1 \geq 1, n_2 = B_2 + 1$, *then*

$$\begin{aligned} f_k(1, n_1, n_2) &= v_k(1, n_1, B_2) + r - v_k(2, n_1, B_2 + 1) \\ &\geq v_k(1, n_1 + 1, B_2) + r - v_k(2, n_1 + 1, B_2 + 1) = f_k(1, n_1 + 1, n_2) \end{aligned}$$

where the inequality follows from Property 4(b). Thus, $f_k(1, n_1+1, n_2) \leq 0$.

<u>Part 5:</u> *To prove that* $s^*_{k+1}(1, n_1, n_2) \leq s^*_{k+1}(1, n_1, n_2 + 1)$, *it suffices to show that if* $f_k(1, n_1, n_2) \geq 0$, *then* $f_k(1, n_1, n_2 + 1) \geq 0$. *Therefore, suppose,* $f_k(1, n_1, n_2) = T_{p_2}v_k(1, n_1, n_2) - T_{p_1}v_k(1, n_1, n_2) \geq 0$. *We divide the state space into two distinct regions to prove Part 5.*

1. *If* $n_1 \geq 1, 1 \leq n_2 < B_2$, *then*

$$\begin{aligned} f_k(1, n_1, n_2) &= v_k(1, n_1, n_2 - 1) + r - v_k(1, n_1 - 1, n_2 + 1) \\ &\leq v_k(1, n_1, n_2) + r - v_k(1, n_1 - 1, n_2 + 2) = f_k(1, n_1, n_2 + 1), \end{aligned}$$

where the inequality above follows from Property 5(a). Thus, $f_k(1, n_1, n_2 + 1) \geq 0$.

2. *If* $n_1 = 0, 1 \leq n_2 \leq B_2$, *then*

$$\begin{aligned} f_k(1, 0, n_2) =& v_k(1, 0, n_2 - 1) + r - v_k(0, 0, n_2 + 1) \\ &\leq v_k(1, 0, n_2) + r - v_k(0, 0, n_2 + 2) = f_k(1, 0, n_2 + 1), \end{aligned}$$

where the inequality above follows from Property 5(b). Thus, $f_k(1, n_1, n_2 + 1) \geq 0$. $\Box$

From Theorem 2.3.2, we see that if our reward-to-go function in period k satisfies the properties in ϑ, then the optimal allocation policy for the $(k+1)^{st}$ period follows a given structure. In the next step, we show that if the optimal policy described in Theorem 2.3.2 will in turn result in a reward-to-go function in period $k+1$ that also satisfies the properties in ϑ, then $v_{k+1} \in \vartheta$. We will prove this through a series of Lemmas. We refer the reader to the Appendix A for the proofs of these lemmas.

Lemma 2.3.1 *If* $v_k \in \vartheta$*, then* $Tv_k \in \vartheta$*, where*

$$Tv_k = \max_{s \in S'}[(\mu_1 + \mu_2)Tp_1v_k - sTp_1v_k + sTp_2v_k].$$

Lemma 2.3.2 *If* $v_k \in \vartheta$*, then* $T_av_k \in \vartheta$*.*

We can now present Theorem 3.

Theorem 2.3.3 *If* $v_k \in \vartheta$*, then* $v_{k+1} \in \vartheta$ *for all* $k \geq 0$*.*

Proof 3 *If* $v_k \in \vartheta$ *then there exists a structured optimal policy for the* $k+1$ *period problem as described in Theorem 2.3.2 and* Tv_k*,* $T_av_k \in \vartheta$ *from Lemmas 2.3.1 and 2.3.2, respectively. Recall that the set of functions* ϑ *are closed under (non-negative) scalar multiplication and addition; then the result immediately follows by definition of* v_{k+1}*.*$\Box$

Thus far, we have shown that the optimality equation is in ϑ, and from this, there exists a structured optimal policy for a finite-horizon problem. We show that the result can be extended to the infinite-horizon and average reward cases. To this end, define the optimality equation for the infinite-horizon discounted reward and average reward cases as follows:

$$\begin{aligned}
v(x,n_1,n_2) &= \max_{s\in S'}[\lambda T_a v(x,n_1,n_2) + (\mu_1+\mu_2-s)T_{p_1}v(x,n_1,n_2) \\
&\quad + sT_{p_2}v(x,n_1,n_2)], \\
g + w(x,n_1,n_2) &= \max_{s\in S'}[\lambda T_a w(x,n_1,n_2) + (\mu_1+\mu_2-s)T_{p_1}w(x,n_1,n_2) \\
&\quad + sT_{p_2}w(x,n_1,n_2)],
\end{aligned}$$

where g is the optimal average expected reward per unit time and $w(x,n_1,n_2)$ is the relative value function in state (x,n_1,n_2) for the average reward case.

Theorem 2.3.4 *For the infinite-horizon discounted reward case, the following results hold:*

1. *$v \in \vartheta$.*
2. *There exists an optimal policy described by Theorem 2.3.2 with the exceptions that $v(x,n_1,n_2)$ replaces $v_k(x,n_1,n_2)$ and $s^*(x,n_1,n_2)$ replaces $s^*_{k+1}(x,n_1,n_2)$.*

Proof 4 *Beginning with the infinite-horizon discounted problem, we base our results on Porteus [30]. First note that the set of functions ϑ is complete; using the L^∞ metric, the limit of any convergent sequence of functions in ϑ is also in ϑ. Let $v_\infty(x,n_1,n_2)$ be the limit of any convergent sequence and let the set of structured decision rules be the decision rules given in Theorem 2. We see that Theorem 2 implies that there exists a structured optimal policy for the one-stage maximization problem with terminal reward $v_\infty(x,n_1,n_2)$. Furthermore, Properties 1 - 5 are preserved in the optimality equation for the one-stage problem with terminal value $v_\infty(x,n_1,n_2)$.*

Hence, by Theorem 5.1 in Porteus [30], $v \in \vartheta$ and there exists a structured optimal policy and the result follows.

For the average-reward case the arguments used will depend on whether the system is finite or infinite. When $B_1, B_2 < \infty$ then the state space is finite and we can use well known arguments to let $\delta \to 0$ (Ross [33]) and deduce that the results follow for the average-reward problem.$\square$

We point out that when the state-space is allowed to be infinite, results concerning average-reward optimal policies are more difficult to establish since a direct argument based on letting δ tend to 0 may no longer be valid.

2.4 Sensitivity Analysis

We have shown that the optimal policy is characterized by a monotone, state-dependent switching curve. However, the exact point at which the curve switches will depend on problem parameters. We are interested in finding trends of how the switching curve shifts. In particular, we study changes in the arrival rate, λ, the ratio of average sever completion rates μ_1/μ_2, and change in buffer sizes, B_1, B_2. We performed more than 120 experiments in which we change one system parameter while fixing the rest. These changes included small to large buffer sizes, μ_1/μ_2 ratios from 1 to 10 and increases in λ value. We found that only changes in λ, μ_1 and μ_2 affect the location of the switching curve. In other words, the switching curve depends on the effective traffic intensity $\rho = \frac{\lambda}{\left(\frac{\mu_1+\mu_2}{2}\right)}$. Figure 2.3 illustrates how the switching curve shifts as ρ increases.

We observe that while $\rho < 1$ and increasing, the switching curve shifts down and there is a greater region over the state space in which it is optimal to place both servers at station 1, i.e., policy $u = 0, s^* = 0$ (Increased region over which $u = 0$ is

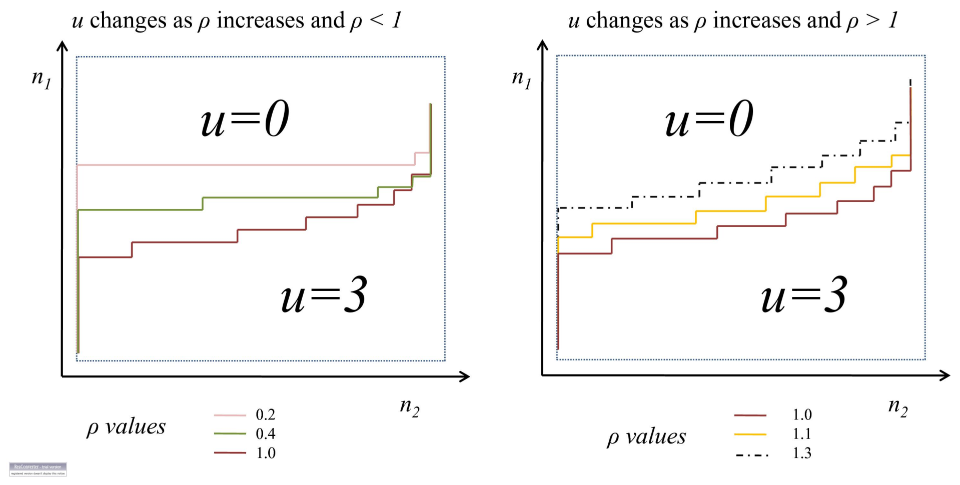

Figure 2.3: Change in structure for different values of ρ when $\mu_1 = 4, \mu_2 = 2, B_1 = B_2 = 20$, and $r = 1$

optimal). When $\rho > 1$, the switching curve shifts back up as ρ increases, and there is a greater region over the state space in which it is optimal to place both servers at station 2, i.e., policy $u = 3, s^* = \mu_1 + \mu_2$. To summarize, $s^*(1, n_1, n_2)$ is decreasing in ρ when $\rho \leq 1$ and it is increasing when $\rho > 1$.

2.4.1 Throughput Calculations

In this section, we present the throughput results from the optimal server assignment policy discussed in Section 2.3. The long-run average throughput converges to a particular value based on the arrival rate λ. When the arrival rate λ is less than the average service rate of $\frac{(\mu_1+\mu_2)}{2}$, the throughput solely depends on λ and the probability that the buffer in front of station 1 is not full. But when the arrival rate λ is greater than the average service rate, the throughput is restricted by the service rates of the servers. The different values of the long run average throughput are provided in Table 2.2. The results follow from the fact the the optimal policy is non-idling as stated in Theorem 2.3.1. When utilization is less than 1 the throughput is limited by the arrival rate; on the other hand, when the utilization is greater than 1, the system approaches the case that an infinite supply of jobs is available as in Andradottir et al [4] and the throughput rate is limited by the rate of the servers. The proof is therefore omitted.

Table 2.2: Throughput convergence values

Value of λ	Throughput convergence value
$\lambda < (\mu_1 + \mu_2)/2$	$\lambda[\Pr\{n_1 < B_1\}]$
$\lambda \geq (\mu_1 + \mu_2)/2$	$(\mu_1 + \mu_2)/2$

2.5 Non-Collaborative Servers

In the case of non-collaborative servers each server will now work independently on one job or part. Again, we assume there is no switching cost for the servers. We follow the same state representation as described in Section 2.2. In this case, the policy u represents, not a combined service rate, but the rate of service of each server at each station, as shown in Table 2.3.

Table 2.3: Server allocation policies for the non-collaborative case.

Policy (u)	Service rate at station 1 (s_1)	Service rate at Station 2 (s_2)
0	μ_1, μ_2	0
1	μ_1	μ_2
2	μ_2	μ_1
3	0	μ_1, μ_2

Note that $u = 1, 2$ is the same as for the case of collaborative servers, since when this control is in place the servers are split between the two stations. On the other hand, in this non-collaborative case $u = 0, 3$ indicate that each server will work at its own rate (and on a unique job), not at the collaborative rate shown in Section 2.2. For example, at $u = 0$, the first part arriving at station 1 will be processed at rate μ_1 by server 1, and server 2 will remain idle unless a second part arrives for processing. Without loss of generality, we assume that if two servers are at a station but there is only one part to process then the faster server will work on that part. Given the above policies, the resulting throughput maximization equation can now be shown as:

$$v_{k+1}(x,n_1,n_2) = \lambda T_a v_k(x,n_1,n_2) + \max \left\{ \begin{array}{ll} \mu_1 T_{p_1} v_k(x,n_1,n_2) & +\mu_2 T_{p_{11}} v_k(x,n_1,n_2), \\ \mu_1 T_{p_1} v_k(x,n_1,n_2) & +\mu_2 T_{p_2} v_k(x,n_1,n_2), \\ \mu_1 T_{p_2} v_k(x,n_1,n_2) & +\mu_2 T_{p_1} v_k(x,n_1,n_2), \\ \mu_1 T_{p_2} v_k(x,n_1,n_2) & +\mu_2 T_{p_{22}} v_k(x,n_1,n_2). \end{array} \right\}$$

$$\text{for } k = 0,1,\ldots n-1 \text{ and } v_0(x,n_1,n_2) = 0 \text{ for all } (x,n_1,n_2) \in \boldsymbol{\mathcal{X}}$$

The four different policies are: both servers working on at station 1, server 1 working at station 1 and server 2 working at station 2, server 1 working at station 2 and server 2 working at station 1, or both servers working on individual jobs at station 2.

Based on the above equation, the server allocation policy is decided. While T_a, T_{p_1}, T_{p_2} remain as they were discussed in Section 2.2, we now introduce additional state transition operators T_{p11} and $T_{p_{22}}$:

$$T_{p_{11}}v(x,n_1,n_2) = \begin{cases} v(x,n_1,n_2) & \text{if } x = 0, 2. \\ v(1,n_1,n_2) & \text{if } x = 1, n_1 = 0, n_2 \leq B_2 + 1 \\ v(1,n_1-1,n_2+1) & \text{if } x = 1, n_1 \geq 1, n_2 < B_2 + 1 \\ v(2,n_1,n_2) & \text{if } x = 1, n_1 \geq 1, n_2 = B_2 + 1 \end{cases}$$

$$T_{p_{22}}v(x,n_1,n_2) = \begin{cases} v(x,n_1,n_2) & \text{if} \quad n_2 = 0, 1 \\ v(1,n_1,n_2-1) + r & \text{if} \quad x = 1, n_1 \geq 0, 2 \leq n_2 \leq B_2 + 1 \\ v(1,n_1-1,n_2-1+1) + r & \text{if} \quad x = 2, n_1 \geq 1, n_2 = B_2 + 1 \\ v(0,n_1,n_2-1+1) + r & \text{if} \quad x = 2, n_1 = 0, n_2 = B_2 + 1 \end{cases}$$

Notice that the difference in the optimality equations of the non-collaborative and collaborative cases is that in the former two parts may be worked on simultaneously at a station, one by each server, whereas in the latter both servers work on the same part. However, in the collaborative case, a second server will increase the rate at which that part is completed. Also note that as opposed to the collaborative case in which we showed that it is always optimal to have servers work together, there may be instances in the non-collaborative case in which it is optimal to separate the servers. This is so that the servers do not remain idle.

Unlike the collaborative case, numerical results show that the optimal policy for the non-collaborative case is a combination of all four policies $u = \{0,\ 1,\ 2,\ 3\}$. There are three patterns embedded in the structure of policy. When we fix the number in queue 1 such that there is one job in service at station 1, the policy switches from splitting the servers between the two stations to having them work together at station 2 as the number in queue 2 increases. The second pattern exists when there is only one job at station 2 and there is at least one part in queue 1, the policy switches from splitting the servers between the two stations to having them work together at station 1 as the number in queue 1 increases. The third pattern exists when there is at least one job in queue 1 and at least two jobs in station 2, the server assignment policy alternates between allocating both servers to station 1 and allocating both servers to station 2. The basic structure of the policy along with the three patterns is shown in Figure 2.4. We can observe that when $n_1 \geq 1$ and $n_2 \geq 2$, the pattern is very similar to structure of the collaborative case in Figure 2.2.

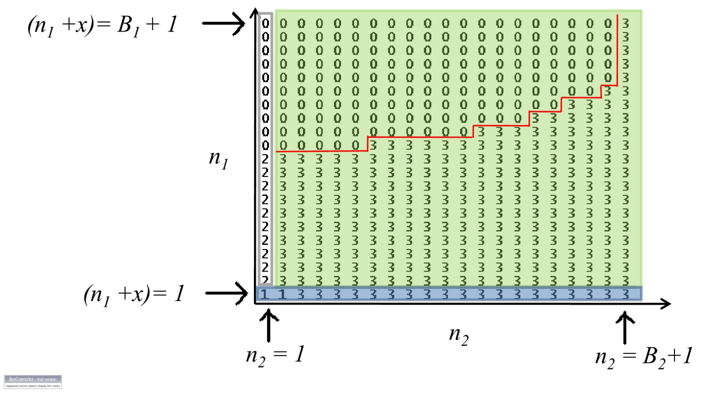

Figure 2.4: Optimal server allocation policy u when $x = 1, \lambda = 3.9, \mu_1 = 6, \mu_2 = 2, B_1 = B_2 = 20$ and $r = 1$

2.6 Conclusions

In this paper, we have modeled the two stage tandem queueing system with two flexible servers in order to maximize throughput. In the case that servers work collaboratively we identify a set of structural properties and prove that the optimal reward-to-go function satisfies these properties, i.e., the optimal allocation policy follows a set of structured rules. In particular, the optimal policy for the collaborative case in the finite-horizon, infinite-horizon, and long-run average problems is such that servers work together at a single station at any point in time. The results show that a non-idling policy is optimal. When there are parts waiting to be processed at both stations then the optimal allocation of servers follows a switching curve, i.e., the policy switches from both servers at one station to both servers at the other station depending on the number of waiting parts.

We have also found that the throughput of the tandem queueing system converges to a particular value depending on system utilization. When the utilization is greater than 1, then our model approaches the case of having an infinite supply of jobs available to the system, as in Andradottir et al. [4], and the results are consistent.

For the case in which servers work non-collaboratively, we identified the server allocation policies based on a numerical analysis. Unlike the collaborative case, the optimal server allocation policy in the non-collaborative case may involve “splitting” up servers. While this seems to contrast the collaborative case (in which servers are always allocated to the same station), the optimal policy is also a non-idling policy, and follows a switching curve.

This research expands previous work in the area in two ways. First, we find the optimal policies for tandem systems in which there is not an infinite supply of jobs but rather finite arrivals to the system. Furthermore, we seek to maximize throughput rather than minimize cost. As opposed to other papers that also seek to maximize throughput, we focus on the structure of the optimal policy (rather than solely on the throughput value) by using a value-iteration approach (instead of a policy iteration approach). In the future we plan to assess the impact of switching costs and time-delays on the structure of the optimal server allocation policy.

Chapter 3

Inventory Based Allocation Policies for Flexible Servers in Serial Systems with Switching cost, Holding cost and Positive reward

3.1 Introduction

In a serial production setting, cross-trained servers may work at more than one station to complete production. These flexible servers may move from one station to another (switch tasks) depending on system requirements or as mandated by pre-defined production rules. However, whenever a server switches tasks there is often some lost production time due to any of the following reasons: (1) the time to move to the other task, (2) the time to set up at the other task, or (3) the "warm-up" period necessary for the worker to become familiar with the new task. A switching cost is often used as a way to indirectly account for lost production time due to re-allocating servers. Such costs must be balanced with conventional costs/rewards such as holding costs and sales profits. We find that the relationship between these costs and reward can greatly influence the optimal server allocation policies. Motivated by one such industry example, and building on previous work related to flexible servers, we study a serial production system with flexible servers facing both costs (switching and holding costs) and rewards (finished goods profits).

In this paper we present a stylized model of a firm which has the ability to dynamically allocate flexible servers. In particular, we consider a make-to-order two-station tandem queueing system in which each station requires at least one server to perform the operation. There are two flexible servers capable of working at either of the stations with service rates independent of each other. In this research, we assume that servers may work collaboratively (i.e., servers can work at the same station at the same time on the same job) with the objective of minimizing the total system-wide costs. Moreover, we assume there is an infinite buffer in front of both the stations. The possible costs to the system include: switching cost charged whenever a server switches from one station to another (fixed cost); holding cost rate (per unit, per unit time) for the number of jobs waiting in front of each of the stations; profit (positive reward per unit) for a finished product leaving the system.

There has been a significant stream of literature related to allocation of cross-trained (flexible) servers in different system settings. Early studies of server allocation policies looked at static (fixed) server assignments across stations. In the context of static server assignment problems, much work also exists which investigates how different station arrangements in a serial production line could affect costs or throughput ([14], [16]). Research was extended to include dynamic server allocation policies. In this context, significant literature exists on how the servers should be allocated across parallel, serial and tandem queueing systems. For a detailed literature review of the optimal server assignment problems on serial and parallel queueing systems and static server assignments, please refer to Arumugam, Mayorga and Taaffe [6]. Much of this work focuses on minimizing conventional costs (holding costs, operating costs) in make-to-stock systems. When considering the objective of maximizing throughput (without cost considerations), please refer to Androdottir et al. [3, 4, 5], as well as Arumugam et al. [6].

Researchers have quantified losses due to switching based either on the cost of switching or the time to switch. Glazebrook [13] was first to consider switching cost in allocating machines to jobs to determine their order of processing. Following this initial contribution, Oyen and Teneketzis [26] and Oyen, Pandelis and Teneketzis [37] studied the allocation of a single server across N parallel queues with switching cost and switching delays. They determined the server allocation policy by minimizing the holding and switching costs. Later Koole [21] considered two parallel stations with arrivals and studied a single server allocation by minimizing the discounted costs. Iravani, Posner and Buzacott [18] proposed a two station tandem queueing system with arrivals and studied the optimal server allocation policy of a single server in minimizing both the holding and switching costs. Kitaev and Serfozo [20] studied the M/M/1 queues with switching costs where the switching costs are based on the choice of arrival and service rates at decision epochs. Recently, Iravani, Buzacott and Posner [17] considered a single machine scheduling problem and compared it to a N-station tandem queueing system. They have considered holding, switching and shipment costs for a batch of M jobs. Batta, Berman and Wang [7] considered a service center with time-dependent demand. Dividing the entire set of customers into small groups, they minimize the switching and staffing costs, where switching costs are included whenever servers move between groups.

In this paper we account for losses due to switching through a fixed switching cost. Our goal is to dynamically allocate servers in order to minimize costs under different cost structures (switching, holding) when a positive reward exists for finished jobs. The positive reward can be interpreted as profit in a make-to-order system or as a way to account for throughput. While many researchers have considered the costs we describe, no work (to our knowledge) studies how the allocation polices might change when a positive reward is included. We will show that the inclusion of a reward for finished goods alters the structure of the optimal policy. When the optimal policy follows complex state-dependent structures we propose alternative heuristic policies. In a detailed numerical study we assess the performance of the heuristic policies and show that a simpler structure which keeps one server static yields close to optimal results.

3.2 Problem Description and Model Formulation

Consider a make-to-order two-station tandem queueing system in which orders for products arrive to the system according to a Poisson process with rate $\lambda > 0$. We assume that there are infinite buffers in front of both stations, and each arriving job must pass through both the stations in series before leaving the system. There are two flexible servers with service rates μ_1 and μ_2, respectively, and the individual server rates do not change based on their current station location. Without loss of generality, we can assume $\mu_1 \geq \mu_2$. Servers may work collaboratively at a station, and given that we have two stations, the *overall service rate* at one station is dependent on the *overall service rate* at the other station. Since the total service rate across the two stations is $\mu_1 + \mu_2$, we let s denote the service rate at station 2 and $\mu_1 + \mu_2 - s$ denote the service rate at station 1. A pictorial representation of the system is provided in Figure 3.1.

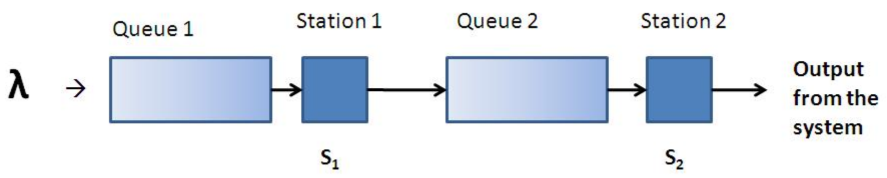

Figure 3.1: Model description

We define the following state variables:

$$
\begin{aligned}
N_1(t) &= \text{Number in queue and in service at station 1 at time } t, \\
&\quad \text{where } N_1(t) \in \{0, 1, 2, \ldots \infty\}, \\
N_2(t) &= \text{Total number in queue and in service at station 2 at time } t, \\
&\quad \text{where } N_2(t) \in \{0, 1, 2, \ldots \infty\}, \\
Z(t) &= \text{Service rate at station 2 at time } t, \text{ where } Z(t) \in \{0, \mu_2, \mu_1, \mu_1 + \mu_2\}.
\end{aligned}
$$

Let Π denote the set of Markov deterministic policies. For any $u \in \Pi$ and $t \geq 0$, denote $(N_1^u(t), N_2^u(t), Z^u(t))$ as the state of the system under policy u at time t. Moreover, the production manager can control the allocation of servers to stations. The different possible actions are: both servers at station 1; server 1 at station 1 and server 2 at station 2; server 2 at station 1 and server 1 at station 2; and both servers at station 2. We denote a particular allocation of servers to stations as policy u, and the four policies available can be represented by 0, 1, 2, and 3 respectively. Therefore, the different policies u result in the following service rates at each station, presented in Table 3.1. Given a Markov deterministic policy u, $(N_1^u(t), N_2^u(t), Z^u(t))$ can be stated as a continuous time Markov chain, as these actions can be taken at any time based on the state of the system.

Our goal is to minimize system-wide costs of this tandem queueing system where we consider a fixed switching cost (c) incurred each time a server switches

Table 3.1: Server allocation policies for the collaborative case.

Policy (u)	Service rate at Station 1 ($\mu_1 + \mu_2 - s$)	Service rate at Station 2 (s)
0	$\mu_1 + \mu_2$	0
1	μ_1	μ_2
2	μ_2	μ_1
3	0	$\mu_1 + \mu_2$

from one station to another, holding cost rates (h_1, h_2) charged per unit per unit time for the jobs waiting in front of stations 1 and 2, respectively, and a positive reward (r) received when a finished product is completed. Since the state transitions are Markovian, it suffices to consider policies where decisions are made at discrete time epochs. Thus, instead of a continuous time control problem, we apply uniformization in the spirit of Lippman [22] and solve an equivalent discrete time problem. Define γ as the maximum rate of transitions: $\gamma = \mu_1 + \mu_2 + \lambda$. Without loss of generality, we scale δ (the discount factor) and γ such that $\delta + \gamma = 1$. Define $v_k(n_1, n_2, z)$ as the value of being in state (n_1, n_2, z), and let $v_0(n_1, n_2, z) = 0$ for all (n_1, n_2, z).

Then the finite horizon discounted cost problem starting in state (n_1, n_2, z) is written as follows:

$$\begin{aligned} v_{k+1}(n_1, n_2, z) = \min_{s \in S} [&\lambda v_k(n_1 + 1, n_2, z) + (\mu_1 + \mu_2 - s) v_k(n_1 - 1, n_2 + 1, z) \\ &+ s(v_k(n_1, n_2 - 1, z) + r) + T_c(s, z) + h_1 n_1 + h_2 (n_2 - 1)^+], \end{aligned}$$

where $k < n$ and $S = \{0, \mu_2, \mu_1, \mu_1 + \mu_2\}$. In the above equation, T_c represents the switching cost operator and is calculated based on the following relationship between

s (the starting server allocation) and z (the new server allocation):

$$T_c(s,z) = \begin{cases} 0 & \text{if } |z-s| = 0 \\ c & \text{if } |z-s| = \mu_1 \text{ or } \mu_2 \\ 2c & \text{if } |z-s| = \mu_1 + \mu_2 \end{cases}$$

For example, if both servers start at station 2 ($s = \mu_1 + \mu_2$), the switching cost will be c if one server switches to station 1 or $2c$ if both servers switch to station 1.

3.3 Optimal Policy Structures

We study two variants of the switching cost problem: (1) Minimizing switching costs only and (2) Minimizing both switching and holding costs. In both cases, there is a positive reward for finished goods. We have characterized the structure of the optimal allocation policies through a computational analysis and we have presented the pictorial representation of these policies at the end of the paper for easy comparison. It is interesting to note that, when we let all costs equal zero and the reward equal one, our problem reduces to the maximizing throughput problem presented in Arumugam et al [6], in which they show that the optimal policy is characterized by a single monotone switching-curve, and in the collaborative case, it is optimal two make both servers always work together. However, we show that the monotone policy structure is lost when switching costs are introduced, whether or not holding costs are present. We offer managerial insights by considering how the magnitude (relative to the reward obtained for finished products) of these costs can affect the policy structure for the flexible servers. In order to examine the structure of the optimal policy we conducted a computational analysis using dynamic programming in a truncated state space. To do so, we initialize the relative value function such that

$v_0(n_1, n_2, z) = 0$, for all (n_1, n_2, z), then using the recursive optimality equation we apply the value iteration algorithm until the minimum average cost converges with an accuracy of .0005, from this we extract the optimal policy. For tractability purposes we truncate the state space to be of dimension $(400 \times 400 \times 4)$.

Positive Switching costs and Reward. We begin by studying a system with holding costs set to zero (i.e., $h_1 = h_2 = 0$). To avoid the non-trivial solutions, we assume that the switching cost c is greater than the reward value r ($c > r$). The optimal policy specifies the optimal server location at every state. Since we have a three dimensional state space we show the optimal policy as a function of the inventory level based on the initial location of the servers. Figure 3.2 presents the optimal server allocation policies for each of the four possible initial states. Although the optimal policy is characterized by a set of switching curves, in many cases, these are not necessarily monotone. Such a case is illustrated in Figure 3.2, in particular consider the pattern shown in the lower left corner of panel (a), where we begin with both servers at station 1. When there is no work-in-process at station 1 ($n_1 = 0$), and there are some jobs at station 2 ($n_2 > 0$), we want the slower server to be moved from station 1 to station 2. Once $n_1 = 1$, we expect the server to be re-allocated to station 1. However, due to the magnitude of the switching cost, this re-allocation does not immediately occur.

Positive Switching cost, Holding costs, and Reward. We now introduce holding costs, $h_1, h_2 > 0$, as in equation (3.2). We classify the tests into two cases: $h_1 < h_2$, $h_1 > h_2$.

The conventional case is the case that $h_1 < h_2$. This holds, for example, when the first station provides a value added operation. In this case, the optimal

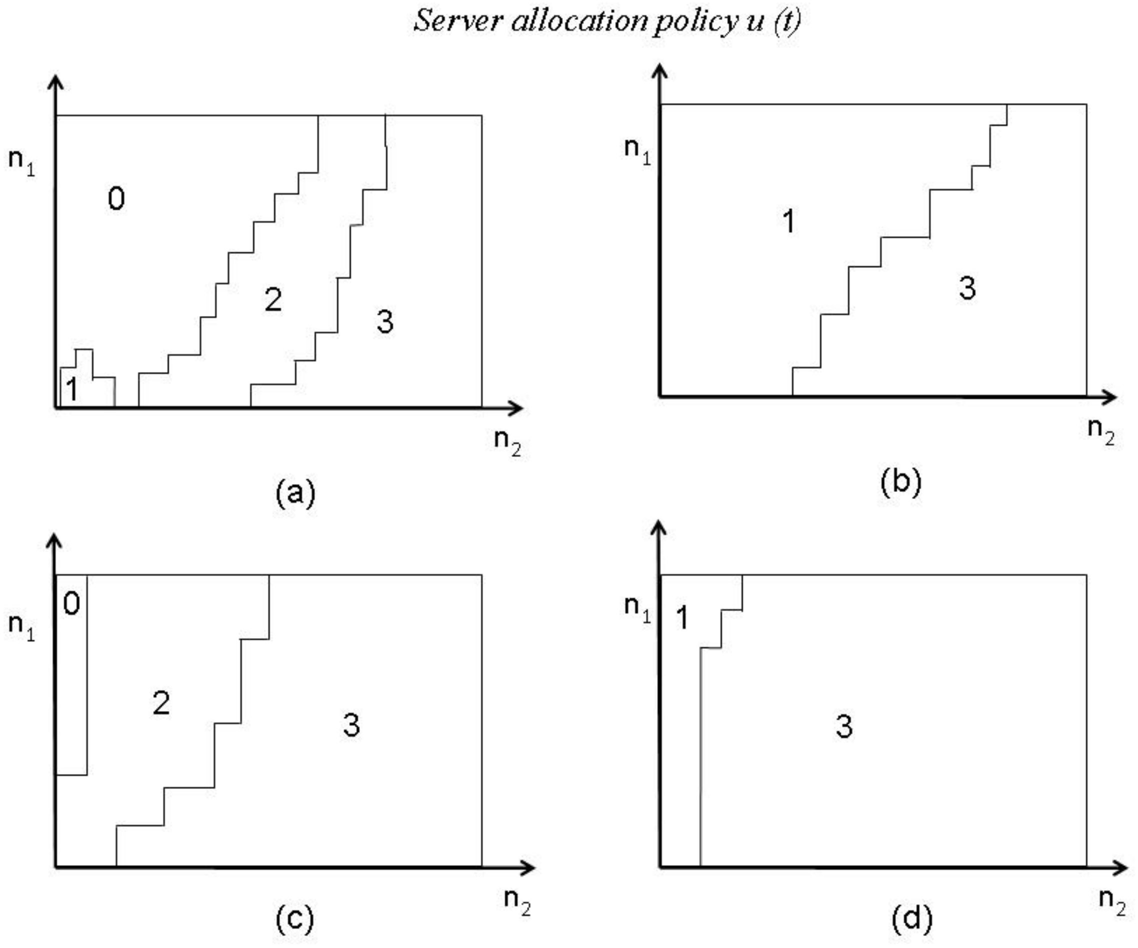

Figure 3.2: Server allocation policy with switching cost only, $(a) z = 0, (b) z = 1, (c) z = 2, (d) z = 3$

policy is depicted in Figure 3.3. Notice that the system allocates both servers to station 2 for all but the smallest work-in-process levels at station 2. When no product is at station 2, both servers can work at station 1 to reduce the product queue. For small queue sizes at station 2, only the faster server is moved to station 2. However, once the work-in-process at station 2 increases sufficiently, the slower server joins the faster server at station 2. The higher holding cost at station 2 (in addition to the positive reward obtained from pushing product out of station 2) encourages the servers to work at station 2 most of the time. The most notable change with the addition of holding costs (where $h_1 < h_2$) is that the servers will go to (or remain at) station 2 sooner (longer) than in the case without holding costs.

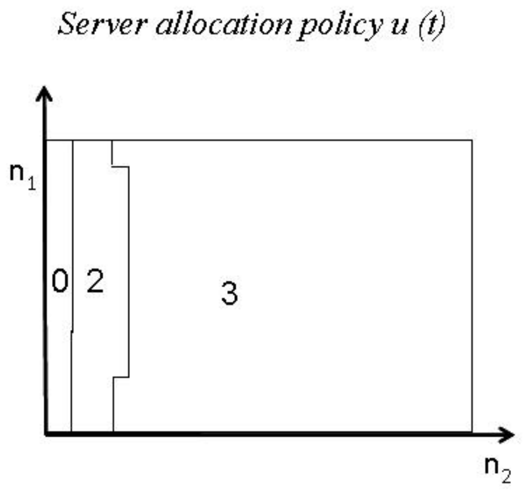

Figure 3.3: Server allocation policy with holding and switching costs, $h_1 < h_2, z = 0$

Next, consider the case that $h_1 > h_2$. This situation may arise, for example, if raw material is perishable, and the first operation alters such volatility. The higher holding cost at station 1 results in both servers working at station 1 for at least some states across all initial server locations. Once inventory builds at station 2, the servers begin shifting to station 2. With inventory at station 2 reaching a threshold value, the holding cost at station 2 now becomes important and has a more negative effect on profit than the larger holding cost at station 1. An illustration of this state-dependent optimal policy is provided in Figure 3.4. In particular, consider panel (b), in which the system starts with the faster server at station 1. We see that due to the presence of switching costs, the optimal policy attempts to keep the same server there for a long time. The slower server is only moved to station 1 when no product is at station 2. Note that, due to the higher holding costs at station 1, both servers will work at station 1 when there is no product at station 2, regardless of the initial position of the servers.

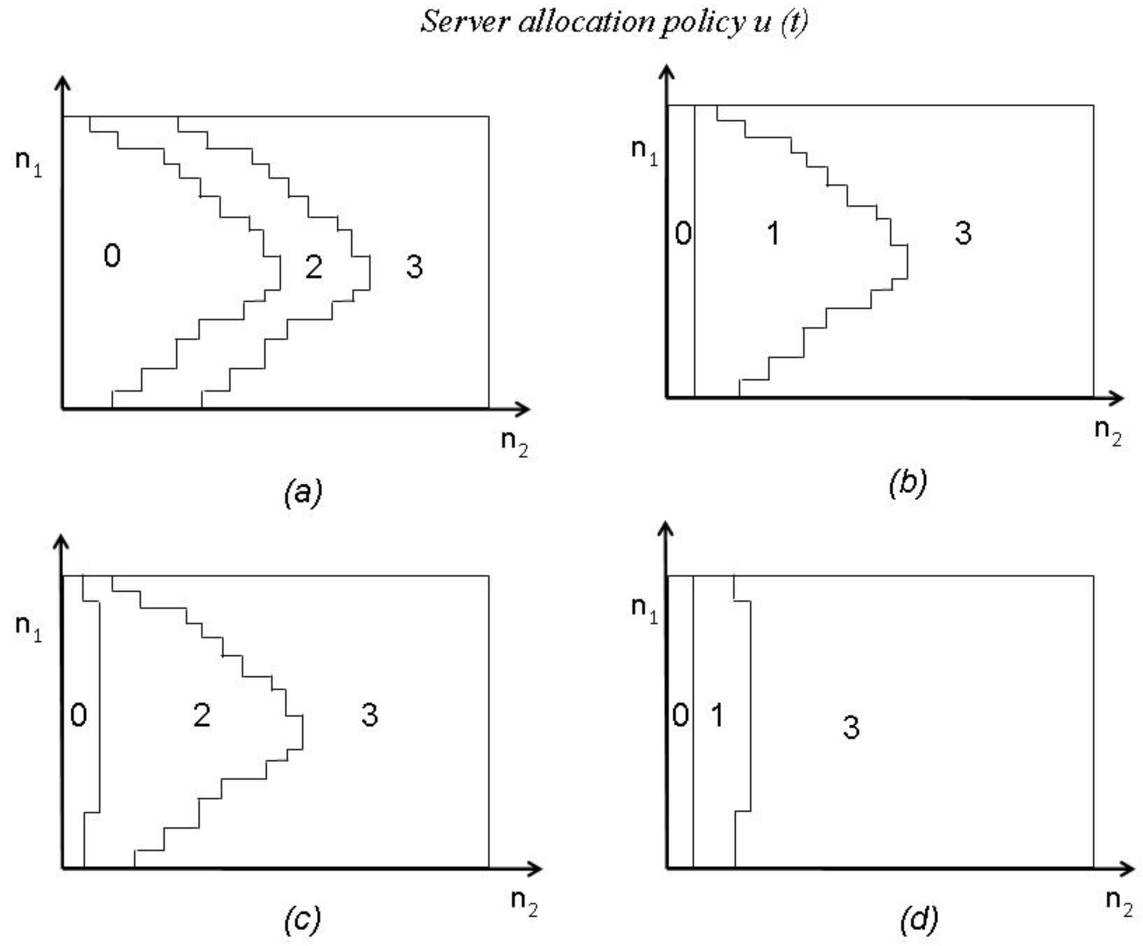

Figure 3.4: Server allocation policy with holding and switching costs, $h_1 > h_2, (a)z = 0, (b)z = 1, (c)z = 2, (d)z = 3$

In all these cases, we have shown that the optimal policy is not only state dependent, but is also often not monotone. This is directly due to the inclusion of switching costs. We know this because when $c = 0$, our problem either reduces to the problem of maximizing throughput as shown by Arumugam et al. [6] (when

$h_1 = h_2 = 0$), or to the problem of minimizing costs tackled by Ahn et al. [2] ($r = 0$). In both papers they were able to analytically prove the structure of the optimal server allocation policy, which was characterized by a monotone, stepwise switching curve. Therefore, the introduction of a switching cost results in a breakdown of a well-structured policy, making an analytical approach intractable.

While it is possible to solve for the optimal policy numerically, such complex policies can be difficult to implement in practice. Furthermore, successful execution of a complicated optimal policy should be rewarded by a substantial reduction in cost because implementing a complex set of rules in practice requires both financial and administrative overhead. However, as we will show in Section 3.4, only a marginal gain is realized by implementing the optimal policy, in which both servers are fully flexible. We instead consider three simpler types of control policies, ones in which we restrict the flexibility of the servers. These policies are described in detail in the next section.

3.4 Heuristic Policies: Definition and Performance

Although the policies described in the previous section are optimal, they may prove difficult to implement. Instead, we consider a number of implementable heuristic policies and compare their performance against the optimal policy. These policies have the merit of possessing fewer parameters to optimize which results in simpler policy structures. Furthermore, by analyzing their performance, we will show how big of a gain the manufacturer can realize if it is able to dynamically change the location of either server. The feature that these alternative policies share is that each restricts the flexibility of the servers.

1. **Static** In a static policy, servers fixed at each of the stations. This is the most

restrictive policy, in which the servers are allocated to a station a-priori and remain there. There are two possible allocation policies: the faster server is fixed at the first station, the faster server is fixed at the second station. The Static policy chooses the best among these two options. We note that since the servers are fixed at each of the stations, the arrival rate to the system should be less than the minimum service rates of the two servers in order to maintain stability (if the arrival rate is greater, it results in infinite queue). Therefore, this heuristic will be infeasible for some test cases.

2. **Flex1** In this heuristic we fix one server at one station and let the other remain flexible. Therefore, there are four possible combinations: the faster server is fixed at station 1, the faster server fixed at station 2, the slower server fixed at station 1, and the slower server fixed at station 2. The Flex1 policy chooses the best among these four possible options.

3. **Collab** In this policy servers are forced to collaborate, they may move dynamically as long as they move together. This policy is motivated by the problem studied in Arumugam et al. [6], in which is was optimal to move both servers together. Note that since both the servers always move together from one station to another, any move results in a fixed charged equal to twice the switching cost, c.

Next we analyze the performance of the three heuristic policies presented above. We compare the cost obtained by the optimal policy to that achieved by each of the heuristic policies. The performance measure we use is percent suboptimality, where

$$Gap^H = \frac{g^H - g^*}{g^*} \times 100.$$

That is, Gap^H denotes the deviation from optimality of heuristic policy H, where where g^* is the long-run-average cost of the optimal policy, and g^H is the long-run-average cost of applying heuristic policy H. As with the optimal problem, each heuristic is formulated as a dynamic program except that for the heuristics the control space is restricted accordingly, again we apply the value iteration algorithm to calculate the cost.

Our study tests the performance of the heuristic policies for all possible combinations of parameters of interest. We tested five parameters $(c, r, h_1/h_2, \mu_1/\mu_2, \rho)$ at different levels. The switching cost and reward were each varied by three different levels ($c = \{0.0, 0.5, 2.0\}$), ($r = \{1, 5, 10\}$). The holding cost rate and service rate ratios were varied by two different levels ($h_1/h_2 = \{0.01/0.05, 0.05/0.01\}$), ($\mu_1/\mu_2 = \{4, 2\}, \{6, 2\}$). The system utilization ($\rho = \frac{\lambda}{(\mu_1+\mu_2)/2}$), was varied by four different levels ($\rho = \{0.25, 0.50, 0.75, 0.90\}$). This is a total if 144 test cases. We present the results in two separate tables, Table 3.2 gives the results for $h_1 < h_2$ and Table 3.3 gives the results when $h_1 > h_2$. Each of the experiments listed in the tables gives the average over eight individual test runs (two levels of μ_1/μ_2 and four levels of ρ). Please refer to Appendix B for the whole list of test cases. Also, note that for the Static policy, results shown are the average over all feasible cases, infeasible cases result in an infinite gap.

From these results, we make several observations regarding the performance of the heuristics. In particular We can see that when the switching cost is zero, the **Collab** policy (forcing both the servers move together) is indeed optimal. This result is interesting because while this has been proven to be the case when maximizing throughput ([6]) and when minimizing holding costs only ([2]), it has never been shown that a bang-bang policy such as this is optimal when both holding costs and

Table 3.2: Summary of Experimental Results for all policies when $h_1 < h_2$. Numbers shown are in percent suboptimal.

Exp No.	c	r	Static	Flex1	Collab
1	0	1	3.84	0.37	0.00
2	0.5	1	2.77	0.05	20.56
3	2	1	2.77	0.00	46.40
4	0	5	0.76	0.07	0.00
5	2.5	5	0.54	0.00	9.62
6	10	5	0.55	0.00	20.29
7	0	10	0.37	0.04	0.00
8	5	10	0.27	0.00	6.90
9	20	10	0.29	0.01	14.35

Table 3.3: Summary of Experimental Results for all policies when $h_1 > h_2$. Numbers shown are in percent suboptimal.

Exp No.	c	r	Static	Flex1	Collab
1	0	1	3.66	0.53	0.00
2	0.5	1	2.79	0.00	14.54
3	2	1	2.76	0.01	32.67
4	0	5	0.72	0.10	0.00
5	2.5	5	0.54	0.00	6.92
6	10	5	0.55	0.00	14.58
7	0	10	0.36	0.05	0.00
8	5	10	0.27	0.00	4.97
9	20	10	0.29	0.01	10.73

reward are present.

In the most interesting case, when the switching cost is positive, the **Collab** policy no longer performs well. In this case the heuristic which fixed one server **Flex1**, performs best. This is true, not only on average but also on an individual case-by-case basis.

The **Static** heuristics is difficult to analyze, as it is infeasible for many test cases. For the feasible cases it performs better ion average that heuristics **Collab**. This is because the feasible cases occur when utilization is very low, in which cases the servers need not be switched as often, yet the Collab policy forces two servers to switch at once.

Recall that heuristics **Static** and **Flex1** involve choosing the best among several possible server configurations. While these results are not presented numerically, we discuss the resulting server allocation here to provide insights to the reader.

In the case of the **Static**, we find that, when the holding cost at station 1 is greater than the holding cost at station 2, it is optimal to have the faster server at station 1. Whereas if the holding cost at station 2 is greater than the holding cost at station 1, it is optimal to have the the faster server at station 2.

In the case of the **Flex1** policy, we find that, fixing a server at station 2 is always optimal. Which server is fixed depends on the relationship between holding costs. When $h_1 > h_2$ the slower server is fixed at station 2, but when $h_1 < h_2$ the faster server is fixed at station 2.

For both of these policies, we observe that the optimality gap is decreasing both the switching cost and in the reward. This is illustrated in Figures 3.5 and 3.6. This is because as switching cost increases, moving the server becomes less desirable and both heuristic policies restrict the movement of the servers. Similarly, as reward increases the optimal policy would like to keep the servers at station 2 as much as possible, thus restricting server movement (as with the heuristics) takes less away from the optimal policy.

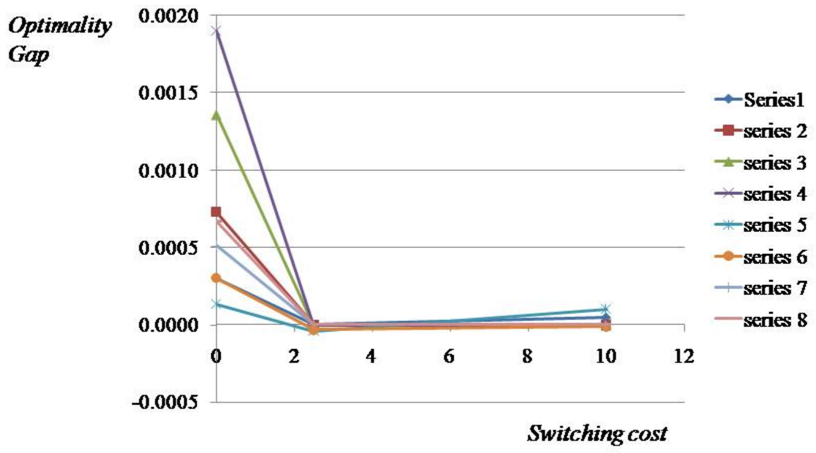

Figure 3.5: Optimality gap as switching cost increases for Collab.

3.5 Conclusions

We have studied problem of allocating flexible servers for a firm that operates a make-to-order serial production system. Our main objective was to characterize the server allocation policies of the two-servers two-station station case while minimizing costs. The costs considered include switching costs, for the servers to move from one station to another; and holding costs for the number of jobs waiting in queue to be processed. Furthermore, we include a positive reward, or revenue term, for stratifying demand by completing production. In particular, we studied how the inclusion of the different costs and a revenue terms affect the structure of the optimal policy. In every

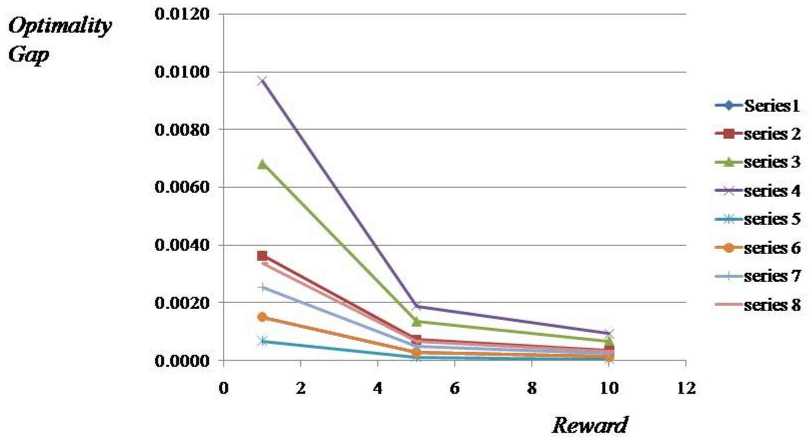

Figure 3.6: Optimality gap as reward increases for Collab.

case, we observed that the inclusion of switching costs results in a state-dependent policy. While these policies can be characterized by switching curves that depend on the inventory level, they are difficult to implement in practice for two reasons. First, since the policies are state-dependent, a different set of curves must be followed depending on the initial allocation of servers to stations. Second, these curves may not be monotone.

Due to the impracticality of applying such complicated policy structures in practice, we developed three simple heuristic policies and compared the results of these heuristics with the optimal policy. We found that a heuristic which fixes one server to a station a-priori, while letting the other server remain flexible performs close to optimal; within 0.05% of optimality on average (over 144 test cases). In particular, in the presence of a positive reward it is optimal to to have the fixed server working at station two. The the value of holding costs determine whether the faster or the slower server is fixed at station two.

While fixing one server is always the best heuristic when switching costs are present, we also point out that when system utilization (ρ) is low (< 0.5) a static policy (in which both servers are fixed a-priori) performs reasonable well (within 4% of optimality for all cases tested). However, applying the "wrong" heuristic policy can lead to very poor results (almost 50% from optimality).

In the future, we would like to extend our research to incorporate switching time directly into the model, by including a delay in service to begin when a server moves from station to station, or by accounting for a warmup period for the server to become familiar with the task. Lastly, it would be interesting to see how our results change if processing times are not only server dependent but also task dependent.

Chapter 4

Conclusions

In our research, we have modeled and analyzed a two stage tandem queueing system with two flexible servers. In our first chapter, we studied two different scenarios on how the servers should be allocated to maximize throughput of the system: collaborative and non-collaborative. In the collaborative working of servers, we found that it is optimal to allocate both the servers together at all times. Therefore, when servers switch from one station to another, they move together. From our analysis, we found that the switching pattern follows a monotone switching curve when there are jobs processed at each of the stations. Whereas in the non-collaborative working of servers, we found that it is no longer optimal to keep both the servers together. It is optimal to split the servers across the two stations when needed to get the maximum output. Also, in Chapter 1, we have proved the structure of the optimal policy for the collaborative working of servers using mathematical iteration techniques.

In Chapter 2, we extended our research to include the switching cost for the servers to move from one station to another and holding costs for the number of jobs waiting at each of the stations. From our analysis, we studied that introducing any of these costs makes the server allocation policy to lose its monotone switching pattern.

We studied the system with just the switching cost and also with having both the switching and holding costs. Once we realized that the optimal server allocation policy is hard to implement in the real-world, we studied 3 different heuristics for the server allocation on the same system: fixing each of the servers at each of the stations, fixing one server and making the other flexible and always moving both the servers together. Among all the heuristics, fixing one server heuristic performed really well and the convergence values of this heuristic was very close to the optimal server allocation policy values.

Our research contribution in this arena has been significant in that, we have described the mathematical formulation of all our problems and have shown the optimal server allocation policies for the systems studied. For our initial maximizing throughput case, we have proved the structure of the optimal server allocation policy. In our extension, we have also compared our complex optimal policies with simple heuristics for easy implementation of our research in the real world.

In the future, we are trying to incorporate switching time into the model. Therefore, when the servers move from one station to another, there is a time lag involved with it instead of having a quantified loss associated with it. This makes the system more complicated and difficult to study. There is extensive literature available on server allocation problems involving switching time.

Appendices

Appendix A

Here, we provide the proofs for Lemmas 1 and 2 in Section 2.3. We restate the lemmas here for convenience.

Lemma 1

If $v_k \in \vartheta$, then $Tv_k \in \vartheta$ where,

$$Tv_k = \max_{s \in S'}[(\mu_1 + \mu_2)Tp_1v_k - sTp_1v_k + sTp_2v_k]$$

Proof 5 *For the properties that define set v, we need to show that $Tv_k \in \vartheta$.*

Property 1(a)

Let $(s_a, s_b) = (s^*_{k+1}(x, n_1, n_2), s^*_{k+1}(x, n_1, n_2{+}1))$. Since $s^*_{k+1}(x, n_1, n_2) \leq s^*_{k+1}(x, n_1, n_2+$ $1)$, from theorem 2, there are only three possible combinations for this pair. Therefore, $(s_a, s_b) \in \{(0,0), (0, \mu_1 + \mu_2), (\mu_1 + \mu_2, \mu_1 + \mu_2)\}$.

<u>**Case 1:**</u> *Let $(s_a, s_b) = (0, 0)$. Then,*

$$\begin{aligned} Tv_k(x, n_1, n_2) + r &= r + (\mu_1 + \mu_2)v_k(x, n_1 - 1, n_2 + 1) - 0 + 0 \\ &= r + (\mu_1 + \mu_2)v_k(x, n_1 - 1, n_2 + 1) \\ &\geq (\mu_1 + \mu_2)v_k(x, n_1 - 1, n_2 + 2) \\ &= Tv_k(x, n_1, n_2 + 1), \end{aligned}$$

where the inequality above follows from Property 1(a) and the fact that $\mu_1+\mu_2 < r = 1$.

Case 2: *Let* $(s_a, s_b) = (0, \mu_1 + \mu_2)$. *Then,*

$$\begin{aligned} Tv_k(x, n_1, n_2) + r &= r + [(\mu_1 + \mu_2)v_k(x, n_1 - 1, n_2 + 1) - 0 + 0] \\ &= r + (\mu_1 + \mu_2)v_k(x, n_1 - 1, n_2 + 1) \\ &\geq (\mu_1 + \mu_2)[v_k(x, n_1, n_2) + r] \\ &= Tv_k(x, n_1, n_2 + 1), \end{aligned}$$

where the inequality above follows from Property 2 and the fact that $r > (\mu_1 + \mu_2)r$.

Case 3: *Let* $(s_a, s_b) = (\mu_1 + \mu_2, \mu_1 + \mu_2)$. *Then,*

$$\begin{aligned} Tv_k(x, n_1, n_2) + r &= r + [(\mu_1 + \mu_2)v_k(x, n_1 - 1, n_2 + 1) \\ &\quad - (\mu_1 + \mu_2)v_k(x, n_1 - 1, n_2 + 1) + (\mu_1 + \mu_2)[v_k(x, n_1, n_2 - 1) + r] \\ &= r + (\mu_1 + \mu_2)[v_k(x, n_1, n_2 - 1) + r] \\ &\geq (\mu_1 + \mu_2)[v_k(x, n_1, n_2) + r] \\ &= Tv_k(x, n_1, n_2 + 1), \end{aligned}$$

where the inequality above follows from Property 1(a) and the fact that $r > (\mu_1+\mu_2)r$.

Property 1(b)

Let $(s_a, s_b) = (s^*_{k+1}(1, B_1, 1), s^*_{k+1}(1, B_1, 0))$. *Note that there are only three possible combinations for this pair. Therefore,* $(s_a, s_b) \in \{(0, 0), (\mu_1+\mu_2, 0), (\mu_1+\mu_2, \mu_1+\mu_2)\}$.

Case 1: *Let* $(s_a, s_b) = (0, 0)$. *Then,*

$$\begin{aligned} Tv_k(1, B_1, 1) &= (\mu_1 + \mu_2)v_k(1, B_1 - 1, 2) - 0 + 0 \\ &\geq v_k(1, B_1 - 1, 1) \\ &= Tv_k(1, B_1, 0), \end{aligned}$$

where the inequality above follows from Property 1(b).

Case 2: *Let* $(s_a, s_b) = (\mu_1 + \mu_2, 0)$. *Then,*

$$\begin{aligned}
Tv_k(1, B_1, 1) &= (\mu_1 + \mu_2)v_k(1, B_1 - 1, 2) \\
&\quad - (\mu_1 + \mu_2)v_k(1, B_1 - 1, 2) + (\mu_1 + \mu_2)[v_k(1, B_1, 0) + r] \\
&\geq (\mu_1 + \mu_2)v_k(1, B_1, 0) \\
&= Tv_k(1, B_1, 0).
\end{aligned}$$

Case 3: *Let* $(s_a, s_b) = (\mu_1 + \mu_2, \mu_1 + \mu_2)$. *Then,*

$$\begin{aligned}
Tv_k(1, B_1, 1) &= (\mu_1 + \mu_2)v_k(1, B_1 - 1, 2) \\
&\quad - (\mu_1 + \mu_2)v_k(1, B_1 - 1, 2) + (\mu_1 + \mu_2)[v_k(1, B_1, 0) + r] \\
&\geq (\mu_1 + \mu_2)v_k(1, B_1, 0) \\
&= Tv_k(1, B_1, 0).
\end{aligned}$$

Property 2(a)

Let $(s_a, s_b) = (s^*_{k+1}(x, n_1, n_2 + 1), s^*_{k+1}(x, n_1 + 1, n_2))$. *Note that there are only three possible combinations for this pair. Therefore,* $(s_a, s_b) \in \{(0, 0), (\mu_1 + \mu_2, 0), (\mu_1 + \mu_2, \mu_1 + \mu_2)\}$. *In other words,* $s^*_{k+1}(x, n_1, n_2 + 1) \geq s^*_{k+1}(x, n_1 + 1, n_2)$

Case 1: *Let* $(s_a, s_b) = (0, 0)$. *Then,*

$$\begin{aligned}
Tv_k(x, n_1, n_2 + 1) &= [(\mu_1 + \mu_2)v_k(x, n_1 - 1, n_2 + 2) - 0 + 0] \\
&\geq [(\mu_1 + \mu_2)v_k(x, n_1, n_2 + 1)] \\
&= Tv_k(x, n_1 + 1, n_2),
\end{aligned}$$

where the inequality above follows from Property 2(a).

Case 2: *Let* $(s_a, s_b) = (\mu_1 + \mu_2, 0)$. *Then,*

$$\begin{aligned} Tv_k(x, n_1, n_2+1) &= [(\mu_1+\mu_2)v_k(x, n_1-1, n_2+2) \\ &\quad - (\mu_1+\mu_2)v_k(x, n_1-1, n_2+2) + (\mu_1+\mu_2)[v_k(x, n_1, n_2) + r]] \\ &= (\mu_1+\mu_2)[v_k(x, n_1, n_2) + r] \\ &\geq (\mu_1+\mu_2)v_k(x, n_1, n_2+1) \\ &= Tv_k(x, n_1+1, n_2), \end{aligned}$$

where the inequality above follows from Property 1(a).

Case 3: *Let* $(s_a, s_b) = (\mu_1 + \mu_2, \mu_1 + \mu_2)$. *Then,*

$$\begin{aligned} Tv_k(x, n_1, n_2+1) &= [(\mu_1+\mu_2)v_k(x, n_1-1, n_2+2) \\ &\quad - (\mu_1+\mu_2)v_k(x, n_1-1, n_2+2) + (\mu_1+\mu_2)[v_k(x, n_1, n_2) + r]] \\ &= (\mu_1+\mu_2)[v_k(x, n_1, n_2) + r] \\ &\geq (\mu_1+\mu_2)[v_k(x, n_1+1, n_2-1) + r] \\ &= Tv_k(x, n_1+1, n_2), \end{aligned}$$

where the inequality above follows from Property 2(a).

Property 2(b)

Let $(s_a, s_b) = (s^*_{k+1}(0, 0, n_2+1), s^*_{k+1}(1, 0, n_2))$. *Note that there are only two possible combinations for this pair. Therefore,* $(s_a, s_b) \in \{(0,0), (\mu_1+\mu_2, \mu_1+\mu_2)\}$.

Case 1: *Let* $(s_a, s_b) = (0, 0)$. *Then,*

$$\begin{aligned} Tv_k(0, 0, n_2+1) &= (\mu_1+\mu_2)v_k(0, 0, n_2+1) - 0 + 0 \\ &= (\mu_1+\mu_2)v_k(0, 0, n_2+1) \\ &= Tv_k(1, 0, n_2). \end{aligned}$$

Case 2: . *Let* $(s_a, s_b) = (\mu_1 + \mu_2, \mu_1 + \mu_2)$. *Then,*

$$\begin{aligned} Tv_k(0,0,n_2+1) &= (\mu_1+\mu_2)v_k(0,0,n_2+1) \\ &\quad - (\mu_1+\mu_2)v_k(0,0,n_2+1) + (\mu_1+\mu_2)[v_k(0,0,n_2)+r] \\ &\geq (\mu_1+\mu_2)v_k(1,0,n_2-1) \\ &= Tv_k(1,0,n_2), \end{aligned}$$

where the inequality above follows from Property 2(b).

Property 3(a)

Let $(s_a, s_b) = (s^*_{k+1}(0,0,B_2+1), s^*_{k+1}(2,0,B_2+1))$. *Note that there are only one possible combinations for this pair. Therefore,* $(s_a, s_b) \in \{(\mu_1+\mu_2, \mu_1+\mu_2)\}$.

Case 1: *Let* $(s_a, s_b) = (\mu_1 + \mu_2, \mu_1 + \mu_2)$. *Then,*

$$\begin{aligned} Tv_k(0,0,B_2+1) + r &= r + (\mu_1+\mu_2)v_k(0,0,B_2+1) \\ &\quad - (\mu_1+\mu_2)v_k(0,0,B_2+1) + (\mu_1+\mu_2)[v_k(0,0,B_2)+r] \\ &\geq (\mu_1+\mu_2)[v_k(0,0,B_2+1)+r] \\ &= Tv_k(2,0,B_2+1), \end{aligned}$$

where the inequality above follows from Property 1(a) and the fact that $r > (\mu_1+\mu_2)r$.

Property 3(b)

Let $(s_a, s_b) = (s^*_{k+1}(1,n_1,B_2+1), s^*_{k+1}(2,n_1+1,B_2+1))$. *Note that there is only one possible combination for this pair. Therefore,* $(s_a, s_b) \in \{(\mu_1+\mu_2, \mu_1+\mu_2)\}$.

Case 1: *Let* $(s_a, s_b) = (\mu_1 + \mu_2, \mu_1 + \mu_2)$. *Then,*

$$\begin{aligned}
Tv_k(1, n_1, B_2 + 1) + r &= r + [(\mu_1 + \mu_2)v_k(2, n_1, B_2 + 1) \\
&\quad - (\mu_1 + \mu_2)v_k(2, n_1, B_2 + 1) + (\mu_1 + \mu_2)[v_k(1, n_1, B_2) + r]] \\
&= r + (\mu_1 + \mu_2)[v_k(1, n_1, B_2) + r] \\
&\geq (\mu_1 + \mu_2)[v_k(1, n_1, B_2 + 1) + r] \\
&= Tv_k(2, n_1 + 1, n_2),
\end{aligned}$$

where the inequality above follows from Property 1(a) and the fact that $r > (\mu_1+\mu_2)r$.

Property 4(a)

Let $(s_a, s_b, s_c, s_d) = (s^*_{k+1}(1, n_1-1, n_2+1), s^*_{k+1}(1, n_1, n_2+1), s^*_{k+1}(1, n_1, n_2-1), s^*_{k+1}(1, n_1+1, n_2 - 1))$. *Note that there five possible combinations for this pair. Therefore,* $(s_a, s_b, s_c, s_d) \in \{(0, 0, 0, 0), (\mu_1 + \mu_2, 0, 0, 0), (\mu_1 + \mu_2, \mu_1 + \mu_2, 0, 0), (\mu_1 + \mu_2, \mu_1 + \mu_2, \mu_1 + \mu_2, 0), (\mu_1 + \mu_2, \mu_1 + \mu_2, \mu_1 + \mu_2, \mu_1 + \mu_2)\}$

<u>**Case 1:**</u> *Let* $(s_a, s_b, s_c, s_d) = (0, 0, 0, 0)$. *Then,*

$$\begin{aligned}
&Tv_k(1, n_1 - 1, n_2 + 1) - Tv_k(1, n_1, n_2 + 1) = \\
&\quad = [(\mu_1 + \mu_2)v_k(1, n_1 - 2, n_2 + 2) - 0 + 0] - [(\mu_1 + \mu_2)v_k(1, n_1 - 1, n_2 + 2) - 0 + 0] \\
&\quad = (\mu_1 + \mu_2)v_k(1, n_1 - 2, n_2 + 2) - (\mu_1 + \mu_2)v_k(1, n_1 - 1, n_2 + 2) \\
&\quad \leq (\mu_1 + \mu_2)v_k(1, n_1 - 1, n_2) - (\mu_1 + \mu_2)v_k(1, n_1, n_2) \\
&\quad = Tv_k(1, n_1, n_2 - 1) - Tv_k(1, n_1 + 1, n_2 - 1),
\end{aligned}$$

where the inequality above follows from Property 4(a).

<u>**Case 2:**</u> *Let* $(s_a, s_b, s_c, s_d) = (\mu_1 + \mu_2, 0, 0, 0)$. *Then,*

$$
\begin{aligned}
&Tv_k(1,n_1-1,n_2+1) - Tv_k(1,n_1,n_2+1) = \\
&\quad = [(\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) - (\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) \\
&\qquad + (\mu_1+\mu_2)v_k[(1,n_1-1,n_2)+r]] - [(\mu_1+\mu_2)v_k(1,n_1-1,n_2+2) - 0 + 0] \\
&\quad = [(\mu_1+\mu_2)v_k[(1,n_1-1,n_2)+r] - (\mu_1+\mu_2)v_k(1,n_1-1,n_2+2)] \\
&\quad \le (\mu_1+\mu_2)v_k(1,n_1-1,n_2) - (\mu_1+\mu_2)v_k(1,n_1,n_2) \\
&\quad = Tv_k(1,n_1,n_2-1) - Tv_k(1,n_1+1,n_2-1),
\end{aligned}
$$

where the inequality above follows from the fact that, when $s_a = \mu_1+\mu_2$, $f_k(1,n_1,n_2+1) \le 0$. This implies, $r \le v_k(1,n_1-1,n_2+2) - v_k(1,n_1,n_2)$.

Case 3: *Let $(s_a,s_b,s_c,s_d) = (\mu_1+\mu_2, \mu_1+\mu_2, 0, 0)$. Then,*

$$
\begin{aligned}
&Tv_k(1,n_1-1,n_2+1) - Tv_k(1,n_1,n_2+1) = \\
&\quad = [(\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) - (\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) \\
&\qquad + (\mu_1+\mu_2)v_k[(1,n_1-1,n_2)+r]] - [(\mu_1+\mu_2)v_k(1,n_1-1,n_2+2) \\
&\qquad - (\mu_1+\mu_2)v_k(1,n_1-1,n_2+2) + (\mu_1+\mu_2)[v_k(1,n_1,n_2)+r]] \\
&\quad = (\mu_1+\mu_2)v_k[(1,n_1-1,n_2)+r] - (\mu_1+\mu_2)[v_k(1,n_1,n_2)+r] \\
&\quad \le (\mu_1+\mu_2)v_k(1,n_1-1,n_2) - (\mu_1+\mu_2)v_k(1,n_1,n_2) \\
&\quad = Tv_k(1,n_1,n_2-1) - Tv_k(1,n_1+1,n_2-1),
\end{aligned}
$$

Case 4: *Let $(s_a,s_b,s_c,s_d) = (\mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2, 0)$. Then,*

$$
\begin{aligned}
&Tv_k(1,n_1-1,n_2+1) - Tv_k(1,n_1,n_2+1) = \\
&\quad = [(\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) - (\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) \\
&\qquad + (\mu_1+\mu_2)v_k[(1,n_1-1,n_2)+r]] - [(\mu_1+\mu_2)v_k(1,n_1-1,n_2+2) \\
&\qquad - (\mu_1+\mu_2)v_k(1,n_1-1,n_2+2) + (\mu_1+\mu_2)[v_k(1,n_1,n_2)+r]] \\
&\quad = (\mu_1+\mu_2)v_k[(1,n_1-1,n_2)+r] - (\mu_1+\mu_2)[v_k(1,n_1,n_2)+r] \\
&\quad = (\mu_1+\mu_2)v_k(1,n_1-1,n_2) - (\mu_1+\mu_2)v_k(1,n_1,n_2) \\
&\quad \le (\mu_1+\mu_2)[v_k(1,n_1,n_2-2)+r] - (\mu_1+\mu_2)v_k(1,n_1,n_2) \\
&\quad = Tv_k(1,n_1,n_2-1) - Tv_k(1,n_1+1,n_2-1),
\end{aligned}
$$

where the inequality above follows from the fact that, when $s_c = \mu_1+\mu_2$, $f_k(1,n_1,n_2-1) \ge 0$. This implies, $r \ge v_k(1,n_1-1,n_2) - v_k(1,n_1,n_2-2)$.

<u>Case 5:</u> *Let $(s_a,s_b,s_c,s_d) = (\mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2)$. Then,*

$$
\begin{aligned}
&Tv_k(1,n_1-1,n_2+1) - Tv_k(1,n_1,n_2+1) = \\
&\quad = [(\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) - (\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) \\
&\qquad + (\mu_1+\mu_2)v_k[(1,n_1-1,n_2)+r]] - [(\mu_1+\mu_2)v_k(1,n_1-1,n_2+2) \\
&\qquad - (\mu_1+\mu_2)v_k(1,n_1-1,n_2+2) + (\mu_1+\mu_2)[v_k(1,n_1,n_2)+r]] \\
&\quad = (\mu_1+\mu_2)v_k[(1,n_1-1,n_2)+r] - (\mu_1+\mu_2)[v_k(1,n_1,n_2)+r] \\
&\quad = (\mu_1+\mu_2)v_k(1,n_1-1,n_2) - (\mu_1+\mu_2)v_k(1,n_1,n_2) \\
&\quad \le (\mu_1+\mu_2)v_k(1,n_1,n_2-2) - (\mu_1+\mu_2)v_k(1,n_1+1,n_2-2) \\
&\quad = Tv_k(1,n_1,n_2-1) - Tv_k(1,n_1+1,n_2-1),
\end{aligned}
$$

where the inequality above follows from Property 4(a).

Property 4(b)

Let $(s_a, s_b, s_c, s_d) = (s^*_{k+1}(2, n_1, n_2), s^*_{k+1}(2, n_1+1, n_2), s^*_{k+1}(1, n_1, n_2-1), s^*_{k+1}(1, n_1+1, n_2-1))$*. Note that there are only three possible combinations for this pair. Therefore,* $(s_a, s_b, s_c, s_d) \in \{(\mu_1+\mu_2, \mu_1+\mu_2, 0, 0), (\mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2, 0), (\mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2)\}$

Case 1: *Let* $(s_a, s_b, s_c, s_d) = (\mu_1+\mu_2, \mu_1+\mu_2, 0, 0)$*. Then,*

$$\begin{aligned}
&Tv_k(2, n_1, n_2) - Tv_k(2, n_1+1, n_2) = \\
&\quad = [(\mu_1+\mu_2)v_k(2, n_1, n_2) - (\mu_1+\mu_2)v_k(2, n_1, n_2) \\
&\qquad + (\mu_1+\mu_2)[v_k(1, n_1-1, n_2) + r]] - [(\mu_1+\mu_2)v_k(2, n_1+1, n_2) \\
&\qquad - (\mu_1+\mu_2)v_k(2, n_1+1, n_2) + (\mu_1+\mu_2)[v_k(1, n_1, n_2) + r]] \\
&\quad = (\mu_1+\mu_2)[v_k(1, n_1-1, n_2) + r] - (\mu_1+\mu_2)[v_k(1, n_1, n_2) + r] \\
&\quad = (\mu_1+\mu_2)v_k(1, n_1-1, n_2) - (\mu_1+\mu_2)v_k(1, n_1, n_2) \\
&\quad \le v_k(1, n_1-1, n_2) - v_k(1, n_1, n_2) \\
&\quad = Tv_k(1, n_1, n_2-1) - Tv_k(1, n_1+1, n_2-1).
\end{aligned}$$

Case 2: *Let* $(s_a, s_b, s_c, s_d) = (\mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2, 0)$*. Then,*

$$
\begin{aligned}
&Tv_k(2,n_1,n_2) - Tv_k(2,n_1+1,n_2) = \\
&\quad = [(\mu_1+\mu_2)v_k(2,n_1,n_2) - (\mu_1+\mu_2)v_k(2,n_1,n_2) \\
&\qquad + (\mu_1+\mu_2)[v_k(1,n_1-1,n_2)+r]] - [(\mu_1+\mu_2)v_k(2,n_1+1,n_2) \\
&\qquad - (\mu_1+\mu_2)v_k(2,n_1+1,n_2) + (\mu_1+\mu_2)[v_k(1,n_1,n_2)+r]] \\
&\quad = (\mu_1+\mu_2)[v_k(1,n_1-1,n_2)+r] - (\mu_1+\mu_2)[v_k(1,n_1,n_2)+r] \\
&\quad = (\mu_1+\mu_2)v_k(1,n_1-1,n_2) - (\mu_1+\mu_2)v_k(1,n_1,n_2) \\
&\quad \le [v_k(1,n_1,n_2-2)+r] - v_k(1,n_1,n_2) \\
&\quad = Tv_k(1,n_1,n_2-1) - Tv_k(1,n_1+1,n_2-1),
\end{aligned}
$$

where the inequality above follows from the fact that, when $s_c = \mu_1+\mu_2$, $f_k(1,n_1,n_2-1) \ge 0$. This implies, $r \ge v_k(1,n_1-1,n_2) - v_k(1,n_1,n_2-2)$

Case 3: *Let $(s_a,s_b,s_c,s_d) = (\mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2)$. Then,*

$$
\begin{aligned}
&Tv_k(2,n_1,n_2) - Tv_k(2,n_1+1,n_2) = \\
&\quad = [(\mu_1+\mu_2)v_k(2,n_1,n_2) - (\mu_1+\mu_2)v_k(2,n_1,n_2) \\
&\qquad + (\mu_1+\mu_2)[v_k(1,n_1-1,n_2)+r]] - [(\mu_1+\mu_2)v_k(2,n_1+1,n_2) \\
&\qquad - (\mu_1+\mu_2)v_k(2,n_1+1,n_2) + (\mu_1+\mu_2)[v_k(1,n_1,n_2)+r]] \\
&\quad = (\mu_1+\mu_2)[v_k(1,n_1-1,n_2)+r] - (\mu_1+\mu_2)[v_k(1,n_1,n_2)+r] \\
&\quad \le (\mu_1+\mu_2)[v_k(1,n_1,n_2-2)+r] - (\mu_1+\mu_2)[v_k(1,n_1+1,n_2-2)+r] \\
&\quad = Tv_k(1,n_1,n_2-1) - Tv_k(1,n_1+1,n_2-1),
\end{aligned}
$$

where the inequality above follows from Property 4(a).

Property 5(a)

Let $(s_a,s_b,s_c,s_d) = (s^*_{k+1}(1,n_1-1,n_2+2), s^*_{k+1}(1,n_1-1,n_2+1), s^*_{k+1}(1,n_1,n_2), s^*_{k+1}(1,n_1,n_2-$

1)). *Note that there five possible combinations for this pair. Therefore,* $(s_a, s_b, s_c, s_d) \in \{(0,0,0,0), (\mu_1+\mu_2, 0,0,0), (\mu_1+\mu_2, \mu_1+\mu_2, 0, 0), (\mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2, 0), (\mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2)\}$

Case 1: *Let* $(s_a, s_b, s_c, s_d) = (0,0,0,0)$. *Then,*

$$
\begin{aligned}
&Tv_k(1, n_1-1, n_2+2) - Tv_k(1, n_1-1, n_2+1) = \\
&\quad = [(\mu_1+\mu_2)v_k(1, n_1-2, n_2+3) - 0 + 0] - [(\mu_1+\mu_2)v_k(1, n_1-2, n_2+2) - 0 + 0] \\
&\quad = (\mu_1+\mu_2)v_k(1, n_1-2, n_2+3) - (\mu_1+\mu_2)v_k(1, n_1-2, n_2+2) \\
&\quad \leq (\mu_1+\mu_2)v_k(1, n_1-1, n_2+1) - (\mu_1+\mu_2)v_k(1, n_1-1, n_2) \\
&\quad = Tv_k(1, n_1, n_2) - Tv_k(1, n_1, n_2-1),
\end{aligned}
$$

where the inequality above follows from Property 5(a).

Case 2: *Let* $(s_a, s_b, s_c, s_d) = (\mu_1+\mu_2, 0, 0, 0)$. *Then,*

$$
\begin{aligned}
&Tv_k(1, n_1-1, n_2+2) - Tv_k(1, n_1-1, n_2+1) = \\
&\quad = [(\mu_1+\mu_2)v_k(1, n_1-2, n_2+2) - (\mu_1+\mu_2)v_k(1, n_1-2, n_2+2) \\
&\qquad + (\mu_1+\mu_2)[v_k(1, n_1-1, n_2+1) + r]] - [(\mu_1+\mu_2)v_k(1, n_1-2, n_2+2) - 0 + 0] \\
&\quad \leq (\mu_1+\mu_2)v_k(1, n_1-1, n_2+1) - (\mu_1+\mu_2)v_k(1, n_1-1, n_2) \\
&\quad = Tv_k(1, n_1, n_2) - Tv_k(1, n_1, n_2-1),
\end{aligned}
$$

where the inequality above follows from the fact that, when $s_a = \mu_1 + \mu_2$, $f_k \geq 0$. *This implies that,* $r \leq v_k(1, n_1-2, n_2+2) - v_k(1, n_1-1, n_2)$

.

Case 3: *Let* $(s_a, s_b, s_c, s_d) = (\mu_1+\mu_2, \mu_1+\mu_2, 0, 0)$. *Then,*

$$\begin{aligned}
&Tv_k(1,n_1-1,n_2+2) - Tv_k(1,n_1-1,n_2+1) = \\
&\quad = [(\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) - (\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) \\
&\qquad + (\mu_1+\mu_2)[v_k(1,n_1-1,n_2+1)+r]] - [(\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) \\
&\qquad - (\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) + (\mu_1+\mu_2)[v_k(1,n_1-1,n_2)+r] \\
&\quad = (\mu_1+\mu_2)[v_k(1,n_1-1,n_2+1)+r] - (\mu_1+\mu_2)[v_k(1,n_1-1,n_2)+r] \\
&\quad \le (\mu_1+\mu_2)v_k(1,n_1-1,n_2+1) - (\mu_1+\mu_2)v_k(1,n_1-1,n_2) \\
&\quad = Tv_k(1,n_1,n_2) - Tv_k(1,n_1,n_2-1).
\end{aligned}$$

<u>**Case 4:**</u> *Let* $(s_a,s_b,s_c,s_d) = (\mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2, 0)$. *Then,*

$$\begin{aligned}
&Tv_k(1,n_1-1,n_2+2) - Tv_k(1,n_1-1,n_2+1) = \\
&\quad = [(\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) - (\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) \\
&\qquad + (\mu_1+\mu_2)[v_k(1,n_1-1,n_2+1)+r]] - [(\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) \\
&\qquad - (\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) + (\mu_1+\mu_2)[v_k(1,n_1-1,n_2)+r] \\
&\quad = (\mu_1+\mu_2)[v_k(1,n_1-1,n_2+1)+r] - (\mu_1+\mu_2)[v_k(1,n_1-1,n_2)+r] \\
&\quad \le (\mu_1+\mu_2)[v_k(1,n_1,n_2-1)+r] - (\mu_1+\mu_2)v_k(1,n_1-1,n_2) \\
&\quad = Tv_k(1,n_1,n_2) - Tv_k(1,n_1,n_2-1),
\end{aligned}$$

where the inequality above follows from the fact that, when $s_c = \mu_1+\mu_2$, $f_k(1,n_1,n_2) \ge 0$. *This implies that,* $r \ge v_k(1,n_1-1,n_2+1) - v_k(1,n_1,n_2-1)$.

<u>**Case 5:**</u> *Let* $(s_a,s_b,s_c,s_d) = (\mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2, \mu_1+\mu_2)$. *Then,*

$$
\begin{aligned}
&Tv_k(1,n_1-1,n_2+2) - Tv_k(1,n_1-1,n_2+1) = \\
&\quad = [(\mu_1+\mu_2)v_k(1,n_1-2,n_2+3) - (\mu_1+\mu_2)v_k(1,n_1-2,n_2+3) \\
&\qquad + (\mu_1+\mu_2)[v_k(1,n_1-1,n_2+1)+r]] - [(\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) \\
&\qquad - (\mu_1+\mu_2)v_k(1,n_1-2,n_2+2) + (\mu_1+\mu_2)[v_k(1,n_1-1,n_2)+r] \\
&\quad = (\mu_1+\mu_2)[v_k(1,n_1-1,n_2+1)+r] - (\mu_1+\mu_2)[v_k(1,n_1-1,n_2)+r] \\
&\quad \le (\mu_1+\mu_2)[v_k(1,n_1,n_2-1)+r] - (\mu_1+\mu_2)[v_k(1,n_1,n_2-2)+r] \\
&\quad = Tv_k(1,n_1,n_2) - Tv_k(1,n_1,n_2-1),
\end{aligned}
$$

where the inequality above follows from Property 5(a)

Property 5(b)

Let $(s_a,s_b,s_c,s_d) = (s^*_{k+1}(1,n_1-1,n_2+2), s^*_{k+1}(1,n_1-1,n_2+1), s^*_{k+1}(1,n_1,n_2), s^*_{k+1}(1,n_1,n_2-1))$. *Note that there are only three possible combinations for this pair. Therefore,* $(s_a,s_b,s_c,s_d) \in \{(\mu_1+\mu_2,\mu_1+\mu_2,0,0),(\mu_1+\mu_2,\mu_1+\mu_2,\mu_1+\mu_2,0),(\mu_1+\mu_2,\mu_1+\mu_2,\mu_1+\mu_2,\mu_1+\mu_2)\}$

<u>Case 1:</u> *Let* $(s_a,s_b,s_c,s_d) = (\mu_1+\mu_2,\mu_1+\mu_2,0,0)$. *Then,*

$$
\begin{aligned}
&Tv_k(0,n_1,n_2+2) - Tv_k(0,n_1,n_2+1) = \\
&\quad = [(\mu_1+\mu_2)v_k(0,n_1,n_2+2) - (\mu_1+\mu_2)v_k(0,n_1,n_2+2) \\
&\qquad + (\mu_1+\mu_2)[v_k(0,n_1,n_2+1)+r]] - [(\mu_1+\mu_2)v_k(0,n_1,n_2+1) \\
&\qquad - (\mu_1+\mu_2)v_k(0,n_1,n_2+1) + (\mu_1+\mu_2)[v_k(0,n_1,n_2)+r]] \\
&\quad = (\mu_1+\mu_2)[v_k(0,n_1,n_2+1)+r] - (\mu_1+\mu_2)[v_k(0,n_1,n_2)+r] \\
&\quad = (\mu_1+\mu_2)v_k(0,n_1,n_2+1) - (\mu_1+\mu_2)v_k(0,n_1,n_2) \\
&\quad \le (\mu_1+\mu_2)v_k(0,n_1,n_2+1) - (\mu_1+\mu_2)v_k(0,n_1,n_2) \\
&\quad = Tv_k(1,n_1,n_2) - Tv_k(1,n_1,n_2-1).
\end{aligned}
$$

Case 2: *Let* $(s_a, s_b, s_c, s_d) = (\mu_1 + \mu_2, \mu_1 + \mu_2, \mu_1 + \mu_2, 0)$. *Then,*

$$\begin{aligned}
&Tv_k(0, n_1, n_2 + 2) - Tv_k(0, n_1, n_2 + 1) = \\
&\quad = [(\mu_1 + \mu_2)v_k(0, n_1, n_2 + 2) - (\mu_1 + \mu_2)v_k(0, n_1, n_2 + 2) \\
&\qquad + (\mu_1 + \mu_2)[v_k(0, n_1, n_2 + 1) + r]] - [(\mu_1 + \mu_2)v_k(0, n_1, n_2 + 1) \\
&\qquad - (\mu_1 + \mu_2)v_k(0, n_1, n_2 + 1) + (\mu_1 + \mu_2)[v_k(0, n_1, n_2) + r]] \\
&\quad = (\mu_1 + \mu_2)v_k(0, n_1, n_2 + 1) - (\mu_1 + \mu_2)v_k(0, n_1, n_2) \\
&\quad \leq (\mu_1 + \mu_2)[v_k(1, n_1, n_2 - 1) + r] - (\mu_1 + \mu_2)v_k(0, n_1, n_2) \\
&\quad = Tv_k(1, n_1, n_2) - Tv_k(1, n_1, n_2 - 1),
\end{aligned}$$

where the inequality above follows from the fact that, when $s_c = \mu_1 + \mu_2$, $f_k(1, n_1, n_2) \geq 0$. *This implies that,* $r \geq v_k(0, n_1, n_2 + 1) - v_k(1, n_1, n_2 - 1)$.

Case 3: *Let* $(s_a, s_b, s_c, s_d) = (\mu_1 + \mu_2, \mu_1 + \mu_2, \mu_1 + \mu_2, \mu_1 + \mu_2)$. *Then,*

$$\begin{aligned}
&Tv_k(0, n_1, n_2 + 2) - Tv_k(0, n_1, n_2 + 1) = \\
&\quad = [(\mu_1 + \mu_2)v_k(0, n_1, n_2 + 2) - (\mu_1 + \mu_2)v_k(0, n_1, n_2 + 2) \\
&\qquad + (\mu_1 + \mu_2)[v_k(0, n_1, n_2 + 1) + r]] - [(\mu_1 + \mu_2)v_k(0, n_1, n_2 + 1) \\
&\qquad - (\mu_1 + \mu_2)v_k(0, n_1, n_2 + 1) + (\mu_1 + \mu_2)[v_k(0, n_1, n_2) + r]] \\
&\quad = (\mu_1 + \mu_2)[v_k(0, n_1, n_2 + 1) + r] - (\mu_1 + \mu_2)[v_k(0, n_1, n_2) + r] \\
&\quad \leq [v_k(1, n_1, n_2 - 1) + r] - v_k[(1, n_1, n_2 - 2) + r] \\
&\quad = Tv_k(1, n_1, n_2) - Tv_k(1, n_1, n_2 - 1),
\end{aligned}$$

where the inequality above follows from Property 5(b).$\square$

Lemma 2

If $v_k \in \vartheta$, then $T_a v_k \in \vartheta$.

Proof 6 *For the properties that define set ϑ, we need to show that $T_a v_k \in \vartheta$.*

Property 1(a)

$$\begin{aligned} T_a v_k(x, n_1, n_2) + r &= v_k(x, n_1 + 1, n_2 - 1) + r \\ &\geq v_k(x, n_1 + 1, n_2) \\ &= T_a v_k(x, n_1, n_2 + 1), \end{aligned}$$

where the inequality above follows from Property 1(a).

Property 1(b)

$$\begin{aligned} T_a v_k(1, B_1, 1) &= v_k(1, B_1, 1) \\ &\geq v_k(1, B_1, 0) \\ &= T_a v_k(1, B_1, 0), \end{aligned}$$

where the inequality above follows from Property 1(b).

Property 2(a)

<u>Case 1:</u> *For $1 \leq n_1 < B_1, n_2 = 0$. Then,*

$$\begin{aligned} T_a v_k(x, n_1 - 1, n_2 + 1) &= v_k(x, n_1, n_2 + 1) \\ &\geq v_k(x, n_1 + 1, n_2) \\ &= T_a v_k(x, n_1, n_2), \end{aligned}$$

where the inequality above follows from Property 2(a).

<u>Case 2:</u> *For $n_1 = B_1, n_2 = 0$. Then,*

$$\begin{aligned} T_a v_k(x, n_1 - 1, n_2 + 1) &= v_k(x, n_1, n_2 + 1) \\ &\geq v_k(x, n_1, n_2) \\ &= T_a v_k(x, n_1, n_2), \end{aligned}$$

where the inequality above follows from Property 1(b).

Property 2(b)

$$\begin{aligned} T_a v_k(0, n_1, n_2 + 1) &= v_k(1, n_1, n_2 + 1) \\ &\geq v_k(1, n_1 + 1, n_2) \\ &= T_a v_k(1, n_1, n_2), \end{aligned}$$

where the inequality above follows from Property 2(a).

Property 3(a)

$$\begin{aligned} T_a v_k(0, n_1, n_2) + r &= v_k(1, n_1, n_2) + r \\ &\geq v_k(2, n_1 + 1, n_2) \\ &= T_a v_k(2, n_1, n_2), \end{aligned}$$

where the inequality above follows from Property 3(b).

Property 3(b)

$$\begin{aligned} T_a v_k(1, n_1 - 1, n_2) + r &= v_k(1, n_1, n_2) + r \\ &\geq v_k(2, n_1 + 1, n_2) \\ &= T_a v_k(2, n_1, n_2), \end{aligned}$$

where the inequality above follows from Property 3(b).

Property 4(a)

<u>Case 1:</u> *For* $1 \leq n_1 \leq B_1 - 1, 1 \leq n_2 < B_2 + 1$. *Then,*

$$\begin{aligned} T_a v_k(1, n_1 - 1, n_2 + 1) - T_a v_k(1, n_1, n_2 + 1) = \\ = v_k(1, n_1, n_2 + 1) - v_k(1, n_1 + 1, n_2 + 1) \\ \leq v_k(1, n_1 + 1, n_2 - 1) - v_k(1, n_1 + 2, n_2) \\ = T_a v_k(1, n_1, n_2 - 1) - T_a v_k(1, n_1 + 1, n_2), \end{aligned}$$

where the inequality above follows from Property 4(a).

<u>Case 2:</u> *For* $n_1 = B_1 - 1, 1 \leq n_2 < B_2 + 1$. *Then,*

$$\begin{aligned} T_a v_k(1, n_1 - 1, n_2 + 1) - T_a v_k(1, n_1, n_2 + 1) = \\ = v_k(1, n_1, n_2 + 1) - v_k(1, n_1 + 1, n_2 + 1) \\ \leq v_k(1, n_1 + 1, n_2 - 1) - v_k(1, n_1 + 1, n_2) \\ = T_a v_k(1, n_1, n_2 - 1) - T_a v_k(1, n_1 + 1, n_2), \end{aligned}$$

where the inequality above follows from Property 4(a).

Property 4(b)

To prove:

$T_a v_k(2, n_1, n_2) - T_a v_k(2, n_1 + 1, n_2) \leq T_a v_k(1, n_1, n_2 - 1) - T_a v_k(1, n_1 + 1, n_2 - 1)$

<u>Case 1:</u> *For* $0 \leq n_1 < B_2 - 1, n_2 = B_2 + 1$. *Then,*

$$\begin{aligned} T_a v_k(2, n_1, n_2) - T_a v_k(2, n_1 + 1, n_2) = \\ = v_k(2, n_1 + 1, n_2) - v_k(2, n_1 + 2, n_2) \\ \leq v_k(1, n_1 + 1, n_2 - 1) - v_k(1, n_1 + 2, n_2 - 1) \\ = T_a v_k(1, n_1, n_2 - 1) - T_a v_k(1, n_1 + 1, n_2 - 1), \end{aligned}$$

where the inequality above follows from Property 4(b).

Property 5(a)

<u>Case 1:</u> *For* $1 \le n_1 < B_1, 1 \le n_2 < B_2$. *Then,*

$$\begin{aligned} & T_a v_k(1, n_1-1, n_2+2) - T_a v_k(1, n_1-1, n_2+1) = \\ & \quad = v_k(1, n_1, n_2+2) - v_k(1, n_1, n_2+1) \\ & \quad \le v_k(1, n_1+1, n_2) - v_k(1, n_1+1, n_2-1) \\ & \quad = T_a v_k(1, n_1, n_2) - T_a v_k(1, n_1, n_2-1), \end{aligned}$$

where the inequality above follows from Property 5(a).

<u>Case 2:</u> *For* $n_1 = B_1, 1 \le n_2 < B_2$. *Then,*

$$\begin{aligned} & T_a v_k(1, n_1-1, n_2+2) - T_a v_k(1, n_1-1, n_2+1) = \\ & \quad = v_k(1, n_1, n_2+2) - v_k(1, n_1, n_2+1) \\ & \quad \le v_k(1, n_1, n_2) - v_k(1, n_1, n_2-1) \\ & \quad = T_a v_k(1, n_1, n_2) - T_a v_k(1, n_1, n_2-1). \end{aligned}$$

Property 5(b)

$$\begin{aligned} & T_a v_k(0, n_1, n_2+2) - T_a v_k(0, n_1, n_2+1) = \\ & \quad = v_k(1, n_1, n_2+2) - v_k(1, n_1, n_2+1) \\ & \quad \le v_k(1, n_1+1, n_2) - v_k(1, n_1+1, n_2-1) \\ & \quad = T_a v_k(1, n_1, n_2) - T_a v_k(1, n_1, n_2-1), \end{aligned}$$

where the inequality above follows from Property 5(b). $\square$

Appendix B

Heu2-a - faster server fixed at station 1

Heu2-b - slower server fixed at station 1

Heu2-c - faster server fixed at station 2

Heu2-d - slower server fixed at station 2

Table 1: Appendix - Numerical Analysis when $h_1 = 0.01, h_2 = 0.05$;

No.	λ	c	r	μ_1	μ_2	Heu1	Heu2-a	Heu2-b	Heu2-c	Heu2-d	Heu3
1	0.75	0	1	4	2	0	0.0134	0.0134	0.0015	0.0004	0.0151
2	1.5	0	1	4	2	0	0.0704	0.019	0.0036	0.0023	0.075
3	2.25	0	1	4	2	0	2.7426	0.5279	0.0068	0.0084	n/a
4	2.7	0	1	4	2	0	2.4928	0.6202	0.0096	0.0254	n/a
5	1	0	1	6	2	0	0.0238	0.0238	0.0006	0.0002	0.0249
6	2	0	1	6	2	0	1.9974	0.2597	0.0015	0.001	n/a
7	3	0	1	6	2	0	2.3299	0.661	0.0025	0.0042	n/a
8	3.6	0	1	6	2	0	2.123	0.7164	0.0033	0.0123	n/a
					avg	0	1.4742	0.3552	0.0037	0.0068	0.0383
9	0.75	0.5	1	4	2	0.3012	0.0092	0.0092	0	1	0.0092
10	1.5	0.5	1	4	2	0.2585	0.0553	0	0	0.0309	0.0553
11	2.25	0.5	1	4	2	0.1951	2.9018	0.4848	0	0.0101	n/a
12	2.7	0.5	1	4	2	0.1597	2.6926	0.5694	0.0027	0.0006	n/a
13	1	0.5	1	6	2	0.2628	0.0184	0.0184	0	1	0.0184
14	2	0.5	1	6	2	0.2031	2.0463	0.2243	0	0.0306	n/a
15	3	0.5	1	6	2	0.1434	2.4773	0.6234	0	0.0081	n/a
16	3.6	0.5	1	6	2	0.1203	2.2751	0.678	0.0009	0.0001	n/a
					avg	0.2055	1.5595	0.3259	0.0004	0.26	0.0276
17	0.75	2	1	4	2	0.6386	0.0093	0.0093	0	1	0.0093
18	1.5	2	1	4	2	0.5595	0.0553	0	0	0.0472	0.0553
19	2.25	2	1	4	2	0.4668	3.0352	0.4487	0	0.0388	n/a
20	2.7	2	1	4	2	0.3982	2.9428	0.5057	0	0.0021	n/a
21	1	2	1	6	2	0.5555	0.0183	0.0183	0	0	0.0183
22	2	2	1	6	2	0.4571	2.0728	0.2047	0	0	n/a
23	3	2	1	6	2	0.3428	2.6552	0.5781	0	0.0406	n/a
24	3.6	2	1	6	2	0.2932	2.4848	0.625	0	0.0091	n/a
					avg	0.464	1.6592	0.2987	0	0.1422	0.0276
25	0.75	0	5	4	2	0	0.0026	0.0026	0.0003	0.0001	0.003
26	1.5	0	5	4	2	0	0.0138	0.0037	0.0007	0.0004	0.0147
27	2.25	0	5	4	2	0	0.6346	0.1939	0.0013	0.0016	n/a
28	2.7	0	5	4	2	0	0.6985	0.3303	0.0019	0.0049	n/a
29	1	0	5	6	2	0	0.0047	0.0047	0.0001	0	0.0049
30	2	0	5	6	2	0	0.3986	0.0589	0.0002	0.0002	n/a
31	3	0	5	6	2	0	0.7311	0.3986	0.0005	0.0008	n/a
32	3.6	0	5	6	2	0	0.7758	0.498	0.0006	0.0024	n/a
					avg	0	0.4075	0.1863	0.0007	0.0013	0.0075

Table 2: Appendix - Numerical Analysis when $h_1 = 0.01, h_2 = 0.05$; continuation 1

No.	λ	c	r	μ_1	μ_2	Heu1	Heu2-a	Heu2-b	Heu2-c	Heu2-d	Heu3
33	0.75	2.5	5	4	2	0.143	0.0018	0.0018	0	1	0.0018
34	1.5	2.5	5	4	2	0.1238	0.0107	0	0	0.0096	0.0107
35	2.25	2.5	5	4	2	0.0929	0.6228	0.1679	0	0.008	n/a
36	2.7	2.5	5	4	2	0.0716	0.6824	0.2945	0	0.0007	n/a
37	1	2.5	5	6	2	0.1242	0.0036	0.0036	0	0.0037	0.0036
38	2	2.5	5	6	2	0.0975	0.3898	0.0448	0	0.0145	n/a
39	3	2.5	5	6	2	0.0645	0.7189	0.3711	0	0.0082	n/a
40	3.6	2.5	5	6	2	0.0519	0.763	0.4694	0	0.002	n/a
					avg	0.0962	0.3991	0.1691	0	0.1308	0.0054
41	0.75	10	5	4	2	0.2937	0.0019	0.0019	0	0.0019	0.0019
42	1.5	10	5	4	2	0.2558	0.0107	0	0	0.0108	0.0107
43	2.25	10	5	4	2	0.2019	0.6131	0.1465	0	0.0218	n/a
44	2.7	10	5	4	2	0.158	0.6654	0.2567	0	0.0082	n/a
45	1	10	5	6	2	0.2547	0.0038	0.0038	0.0001	0.0038	0.0038
46	2	10	5	6	2	0.2071	0.3852	0.0376	0	0.0281	n/a
47	3	10	5	6	2	0.1388	0.7056	0.3415	0	0.0264	n/a
48	3.6	10	5	6	2	0.1129	0.7485	0.4368	0	0.0095	n/a
					avg	0.2028	0.3918	0.1531	0	0.0138	0.0055
49	0.75	0	10	4	2	0	0.0013	0.0013	0.0001	0	0.0015
50	1.5	0	10	4	2	0	0.0068	0.0018	0.0003	0.0002	0.0072
51	2.25	0	10	4	2	0	0.3725	0.1524	0.0006	0.0008	n/a
52	2.7	0	10	4	2	0	0.4784	0.2947	0.0009	0.0024	n/a
53	1	0	10	6	2	0	0.0023	0.0023	0	0	0.0024
54	2	0	10	6	2	0	0.1989	0.0342	0.0001	0.0001	n/a
55	3	0	10	6	2	0	0.5321	0.3659	0.0002	0.0004	n/a
56	3.6	0	10	6	2	0	0.6097	0.471	0.0003	0.0012	n/a
					avg	0	0.2753	0.1654	0.0003	0.0006	0.0037
57	0.75	5	10	4	2	0.1026	0.0009	0.0009	0	0.001	0.0009
58	1.5	5	10	4	2	0.089	0.0052	0	0	0.0052	0.0052
59	2.25	5	10	4	2	0.0671	0.3592	0.1345	0	0.0065	n/a
60	2.7	5	10	4	2	0.0511	0.4594	0.2689	0	0.0015	n/a
61	1	5	10	6	2	0.089	0.0019	0.0019	0	0.0019	0.0019
62	2	5	10	6	2	0.0705	0.1918	0.0254	0	0.0101	n/a
63	3	5	10	6	2	0.0457	0.5173	0.3458	0	0.0073	n/a
64	3.6	5	10	6	2	0.0365	0.5942	0.45	0	0.0023	n/a
					avg	0.0689	0.2662	0.1534	0	0.0045	0.0027

Table 3: Appendix - Numerical Analysis when $h_1 = 0.01, h_2 = 0.05$; continuation 2

No.	λ	c	r	μ_1	μ_2	Heu1	Heu2-a	Heu2-b	Heu2-c	Heu2-d	Heu3
65	0.75	20	10	4	2	0.2092	0.0011	0.0011	0.0002	0.0011	0.0011
66	1.5	20	10	4	2	0.1818	0.0053	0	0	0.0055	0.0053
67	2.25	20	10	4	2	0.1423	0.348	0.1194	0	0.0169	n/a
68	2.7	20	10	4	2	0.1094	0.4396	0.2421	0	0.0081	n/a
69	1	20	10	6	2	0.1811	0.0022	0.0022	0.0003	0.0022	0.0022
70	2	20	10	6	2	0.1479	0.1881	0.021	0	0.0188	n/a
71	3	20	10	6	2	0.0965	0.5016	0.3246	0	0.0206	n/a
72	3.6	20	10	6	2	0.0791	0.5772	0.427	0	0.0079	n/a
					avg	0.1434	0.2579	0.1422	0	0.0101	0.0029

Table 4: Appendix - Numerical Analysis when $h_1 = 0.05, h_2 = 0.01$;

No.	λ	c	r	μ_1	μ_2	Heu1	Heu2-a	Heu2-b	Heu2-c	Heu2-d	Heu3
73	0.75	0	1	4	2	0	0.0028	0.0028	0.0003	0.0017	0.0139
74	1.5	0	1	4	2	0	0.0146	0.0714	0.0013	0.0044	0.0716
75	2.25	0	1	4	2	0	2.6274	2.1771	0.006	0.009	n/a
76	2.7	0	1	4	2	0	2.4996	2.1128	0.0221	0.015	n/a
77	1	0	1	6	2	0	0.0047	0.0047	0.0001	0.0008	0.024
78	2	0	1	6	2	0	1.0033	1.2495	0.0005	0.002	n/a
79	3	0	1	6	2	0	2.3163	1.979	0.0027	0.0036	n/a
80	3.6	0	1	6	2	0	2.1344	1.8466	0.0107	0.0055	n/a
					avg	0	1.3254	1.1805	0.0055	0.0053	0.0365
81	0.75	0.5	1	4	2	0.224	0	0	0.0092	0	0.0093
82	1.5	0.5	1	4	2	0.1737	0.0005	0.0561	0.032	0	0.0561
83	2.25	0.5	1	4	2	0.1362	2.7166	2.2416	0.0275	0	n/a
84	2.7	0.5	1	4	2	0.1113	2.631	2.2103	0.015	0.0001	n/a
85	1	0.5	1	6	2	0.195	0	0	0.0177	0	0.0183
86	2	0.5	1	6	2	0.137	1.004	1.2576	0.0382	0	n/a
87	3	0.5	1	6	2	0.1005	2.407	2.0464	0.0207	0	n/a
88	3.6	0.5	1	6	2	0.0855	2.2341	1.921	0.0102	0	n/a
					avg	0.1454	1.3741	1.2166	0.0213	0	0.0279
89	0.75	2	1	4	2	0.4763	0	0	0.0093	0	0.0093
90	1.5	2	1	4	2	0.3821	0	0.0552	0.0472	0	0.0552
91	2.25	2	1	4	2	0.3202	2.7951	2.2984	0.063	0	n/a
92	2.7	2	1	4	2	0.2813	2.779	2.3201	0.0323	0.0004	n/a
93	1	2	1	6	2	0.414	0	0	0.0183	0	0.0183
94	2	2	1	6	2	0.3056	1.0041	1.2633	0.0757	0	n/a
95	3	2	1	6	2	0.2287	2.5236	2.1331	0.0447	0	n/a
96	3.6	2	1	6	2	0.2046	2.3694	2.022	0.0199	0	n/a
					avg	0.3266	1.4339	1.2615	0.0388	0	0.0276
97	0.75	0	5	4	2	0	0.0005	0.0005	0	0.0003	0.0027
98	1.5	0	5	4	2	0	0.0027	0.0141	0.0002	0.0008	0.0141
99	2.25	0	5	4	2	0	0.6074	0.5187	0.0011	0.0017	n/a
100	2.7	0	5	4	2	0	0.6901	0.6157	0.0042	0.0028	n/a
101	1	0	5	6	2	0	0.0009	0.0009	0	0.0001	0.0047
102	2	0	5	6	2	0	0.1995	0.2559	0.0001	0.0003	n/a
103	3	0	5	6	2	0	0.726	0.6592	0.0005	0.0007	n/a
104	3.6	0	5	6	2	0	0.7726	0.7167	0.002	0.001	n/a
					avg	0	0.3749	0.3477	0.001	0.001	0.0072

Table 5: Appendix - Numerical Analysis when $h_1 = 0.05, h_2 = 0.01$; continuation 1

No.	λ	c	r	μ_1	μ_2	Heu1	Heu2-a	Heu2-b	Heu2-c	Heu2-d	Heu3
105	0.75	2.5	5	4	2	0.1067	0	0	0.0018	0	0.0018
106	1.5	2.5	5	4	2	0.0846	0	0.0107	0.0096	0	0.0107
107	2.25	2.5	5	4	2	0.0661	0.5993	0.5087	0.0129	0	n/a
108	2.7	2.5	5	4	2	0.0535	0.6792	0.6023	0.006	0.0001	n/a
109	1	2.5	5	6	2	0.0926	0	0	0.0036	0	0.0036
110	2	2.5	5	6	2	0.0659	0.1904	0.2474	0.0159	0	n/a
111	3	2.5	5	6	2	0.0452	0.7174	0.6485	0.0087	0	n/a
112	3.6	2.5	5	6	2	0.0385	0.7637	0.7056	0.0036	0	n/a
					avg	0.0691	0.3687	0.3404	0.0078	0	0.0054
113	0.75	10	5	4	2	0.2196	0	0	0.0019	0	0.0019
114	1.5	10	5	4	2	0.1764	0	0.0107	0.0107	0	0.0107
115	2.25	10	5	4	2	0.1411	0.5913	0.4989	0.0252	0	n/a
116	2.7	10	5	4	2	0.1182	0.6668	0.5869	0.0122	0.0001	n/a
117	1	10	5	6	2	0.1903	0	0	0.0038	0.0001	0.0038
118	2	10	5	6	2	0.1399	0.1846	0.2419	0.0286	0	n/a
119	3	10	5	6	2	0.0947	0.7074	0.6361	0.017	0	n/a
120	3.6	10	5	6	2	0.0859	0.7531	0.6924	0.0073	0	n/a
					avg	0.1458	0.3629	0.3334	0.0134	0	0.0055
121	0.75	0	10	4	2	0	0.0002	0.0002	0	0.0001	0.0013
122	1.5	0	10	4	2	0	0.0012	0.0069	0.0001	0.0004	0.007
123	2.25	0	10	4	2	0	0.3587	0.3145	0.0005	0.0008	n/a
124	2.7	0	10	4	2	0	0.4736	0.4367	0.0021	0.0014	n/a
125	1	0	10	6	2	0	0.0004	0.0004	0	0	0.0023
126	2	0	10	6	2	0	0.0993	0.1321	0	0.0001	n/a
127	3	0	10	6	2	0	0.5294	0.4961	0.0002	0.0003	n/a
128	3.6	0	10	6	2	0	0.6078	0.5799	0.001	0.0005	n/a
					avg	0	0.2588	0.2458	0.0005	0.0005	0.0035
129	0.75	5	10	4	2	0.0766	0	0	0.0009	0	0.0009
130	1.5	5	10	4	2	0.0611	0	0.0053	0.0052	0	0.0053
131	2.25	5	10	4	2	0.0476	0.3495	0.3047	0.0089	0	n/a
132	2.7	5	10	4	2	0.0385	0.4607	0.4227	0.0041	0	n/a
133	1	5	10	6	2	0.0664	0	0	0.0018	0	0.0018
134	2	5	10	6	2	0.0476	0.0929	0.1258	0.0106	0	n/a
135	3	5	10	6	2	0.0317	0.5188	0.4847	0.0059	0	n/a
136	3.6	5	10	6	2	0.0274	0.5969	0.5682	0.0024	0	n/a
					avg	0.0496	0.2523	0.2389	0.005	0	0.0027

Table 6: Appendix - Numerical Analysis when $h_1 = 0.05, h_2 = 0.01$; continuation 2

No.	λ	c	r	μ_1	μ_2	Heu1	Heu2-a	Heu2-b	Heu2-c	Heu2-d	Heu3
137	0.75	20	10	4	2	0.1565	0.0002	0.0002	0.0011	0.0002	0.0011
138	1.5	20	10	4	2	0.1283	0	0.0054	0.0054	0	0.0054
139	2.25	20	10	4	2	0.1028	0.3402	0.2947	0.0168	0	n/a
140	2.7	20	10	4	2	0.0868	0.4459	0.407	0.0084	0	n/a
141	1	20	10	6	2	0.1366	0.0003	0.0003	0.0022	0.0003	0.0022
142	2	20	10	6	2	0.1066	0.0888	0.1219	0.0187	0	n/a
143	3	20	10	6	2	0.0733	0.5069	0.472	0.0115	0	n/a
144	3.6	20	10	6	2	0.0669	0.5842	0.5547	0.005	0	n/a
					avg	0.1072	0.2458	0.232	0.0086	0	0.0029

Bibliography

[1] H.S. Ahn, I. Duenyas, and M.E. Lewis. Optimal control of a two-stage tandem queueing with flexible servers. *Probability in the Engineering and Informational Sciences*, 16:453–469, 2002.

[2] H.S. Ahn, I. Duenyas, and R. Zhang. Optimal stochastic scheduling of a two-stage tandem queue with parallel servers. *Advances in Applied Probability*, 31:1095–1117, 1999.

[3] S. Andradottir and H.Ayhan. Throughput maximization for tandem lines with two stations and flexible servers. *Operations Research*, 53:516–531, 2005.

[4] S. Andradottir, H.Ayhan, and D.G. Down. Server assignment policies for maximizing the steady-state throughput of finite queueing systems. *Management Science*, 47:1421–1439, 2001.

[5] S. Andradottir, H.Ayhan, and D.G. Down. Dynamic assignment of dedicated and flexible servers in tandem lines. *Masters/Ph.D Thesis*, 2006.

[6] R. Arumugam, M.E. Mayorga, and K.M. Taaffe. Inventory based allocation policies for flexible servers in serial systems. *Annals of Operations Research*, 2008.

[7] R. Batta, O. Berman, and Q. Wang. Balancing staffing and switching costs in a service center with flexible servers. *European Journal of Operations Research*, 177:924–938, 2007.

[8] B.Hajek. Optimal control of interacting service stations. *IEEE Transactions on Automatic Control*, 29(6):491–499, 1984.

[9] D.P. Bischak. Performance of a manufacturing module with moving workers. *IIE Transactions*, 28:723–733, 1996.

[10] J.A. Buzacott. Commonalities in reengineered business processes: Models and issues. *Management Science*, 42:768–782, 1996.

[11] I. Duenyas, D. Gupta, and T.L. Olsen. Control of a single-server tandem queueing system with setups. *Operations Research*, 46(2):218–230, 1998.

[12] T.M. Farrar. Optimal use of an extra server in a two station tandem queueing network. *IEEE Transactions on Automatic Control*, 38:1296–1299, 1993.

[13] K.D. Glazebrook. On stochastic scheduling with precedence relations and switching costs. *Journal of Applied Probability*, 17(4):1016–1024, December 1980.

[14] G.Yamazaki, H. Sakasegawa, and J.G. Shanthikumar. On optimal arrangement of stations in a tandem queueing system with blocking. *Management Science*, 38:137–153, 1992.

[15] J.M. Harrison and M.J. Lopez. Heavy traffic resource pooling in parallel server systems. *Queueing Systems*, 33:339–368, 1999.

[16] F.S. Hillier and K.C. So. On the simulataneous optimization of server and work allocations in production line systems with variable processing times. *Operations Research*, 44:435–443, 1996.

[17] S.M.R. Iravani, J.A. Buzacott, and M.J.M. Posner. Operations and shipment scheduling of a batch on a flexible machine. *Operations Research*, 51(4):585–601, 2003.

[18] S.M.R. Iravani, M.J.M. Posner, and J.A. Buzacott. A two-stage tandem queue attended by a moving server with holding and switching costs. *Queueing systems*, 26:203–228, April 1997.

[19] S.M.R. Irvani, M.J.M. Posner, and J.A. Buzacott. A two-stage tandem queue attended by a moving server with holding and switching costs. *QUESTA*, 26:203–228, 1997.

[20] M.Yu. Kitaev and R.F. Serfozo. M/m/1 queues with switching costs and hysteretic optimal control. *Operations Research*, 47(2):310–312, 1999.

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

[22] S.A. Lippman. Applying a new device in the optimization of exponential queueing systems. *Operations Research*, 23(4):687–710, 1975.

[23] A. Mandelbaum and A.L. Stoylar. Scheduling flexible servers with convex delay costs. *Operations Research*, 52:836–855, 2004.

[24] J.O. McClain, L.J. Thomas, and C. Sox. “on the fly” line balancing with very little wip. *International Journal of Production Economics*, 27:283–289, 1992.

[25] J. Ostolaza, J.0. McClain, and L.J. Thomas. The use of dynamic (state-dependent) assembly-line balancing to improve throughput. *Journal of Manufacturing Operations Management, Volume = 3, Number = , Pages = 105–133*, 1990.

[26] M.P. Oyen and D. Teneketzis. Optimal stochastic scheduling of forest networks with switching penalties. *Proceedings of the 31st conference on decision and control*, pages 3328–3333, December 1992.

[27] M.P. Van Oyen, E.G.S. Gel, and W.J. Hopp. Performance opportunity for workforce agility in collaborative and noncollaborative work systems. *IIE Transactions*, 33:761–777, 2001.

[28] M.P. Van Oyen and D. Teneketzis. Optimal stochastic scheduling of forest networks with switching penalties. *Advances in Applied Probability*, 26:474–497, 1994.

[29] D.G. Pandelis and D. Teneketzes. Optimal multiserver stochastic scheduling of two interconected priority queues. *Advances in Applied Probability*, 26:258–279, 1994.

[30] E.L. Porteus. Conditions for characterizing the structure of optimal strategies in infinite horizon dynamic programs. *Journal of Optimization Theory and Applications*, 36:419–432, 1982.

[31] M.I. Reiman and L.M. Wein. Dynamic scheduling of a two-class queue with setups. *Operations Research*, 46:532–547, 1998.

[32] Z. Rosberg, P.P. Varaiya, and J.C. Walrand. Optimal control of service in tandem queues. *IEEE Transactions on Automatic Control*, 27(3):600–609, 1982.

[33] Sheldon M. Ross. *Applied Probability Models with Optimization Applications.* Holden-Day, San Francisco, 1970.

[34] Sheldon M. Ross. *Introduction to Probability Models.* Academic press, 8 edition, 2003.

[35] S.L.Bell and R.J.Williams. Dynamic schduling of a system with two parallel servers in heavy traffic with complete resource pooling: Asymptotoic optimality of a continuous review threshold policy. *Annals of Applied Probability*, 11:608–649, 2001.

[36] M.S. Squillante, C.H. Xia, D.D. Yao, and L. Zhang. Threshold based priority policies for parallel-server systems with affinity scheduling. *Proceedings of 2001 American Control Conference*, pages 2992–2999, 2001.

[37] V.Oyen, D.G. Pandelis, and D. Teneketzis. Optimality of index policies for stochastic scheduling with switching penalties. *Journal of Applied Probability*, 29(4):957–966, 1992.

[38] R.J. Williams. On dynamic scheduling of a parallel server system with complete resource pooling. *Proceedings of the Fields Institute Workshop on Analysis and Simulation of Communication Networks*, 28:49–71, 2000.

[39] E. Zavadlav, J.O. McClain, and L.J. Thomas. Self-buffering, self-balancing, self-flushing production lines. *Management Science*, 42:1151–1164, 1996.